Neues Konzept zur Entwicklung robuster Multisensorsysteme gegen Störungen der Sensordaten

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

von der KIT-Fakultät für Maschinenbau
des Karlsruher Instituts für Technologie (KIT)

angenommene

Dissertation

von

M.Sc.

Felix Berens

Tag der mündlichen Prüfung: 23.05.2025
Hauptreferent: Prof. Dr. Markus Reischl
Korreferenten: Prof. Dr. Marcus Geimer
Prof. Dr. Stefan Elser

Abstract

In this dissertation, a new concept is presented that serves to increase the robustness of a multi-sensor system against disturbances in the sensor data. Given the unavoidable sensor disturbances in real-world application scenarios, this thesis focuses on how sensor fusion systems for object detection can be made more robust by integrating disturbances during the training process.

Two main steps are formulated for this purpose. In the first, the placement of the sensors is considered. Two new methods are presented for this purpose: Genetic Algorithm Optimisation for Sensor Placement (Genetische Algorithmus-Optimierung für Sensorplatzierung, GAOS) and Deep Learning Optimisation for Sensor Placement (Deep Learning-Optimierung für Sensorplatzierung, DLOS). Both methods optimise the placement of sensors on a vehicle to maximise the sensing field of the combined sensors, but at the same time the redundancy of the sensors against sensor failure is considered. In the next step, a new concept for increasing the robustness of sensor fusion methods is presented. This consists of a new training concept: concept for the integration of disturbance adaptivity (Konzept zur Integration von Störungs-Adaptivität, KISA) and evaluation methods that represent the increase in robustness: Multifactorial Performance Evaluation and Sensitivity Analysis. These concepts address the central hypothesis of this thesis. The hypothesis is that the integration of perturbations during the training process improves the robustness of fusion networks against sensor perturbations. To test this hypothesis, specific research questions are addressed that focus on the impact of perturbed training data on the robustness of fusion networks, the influence of different perturbation types on the training outcome, the comparison of low level and high level fusion techniques, and the depth of network architectures.

The newly developed KISA training concept specifically incorporates disturbances during the training process of sensor fusion systems. KISA makes it possible to systematically increase the robustness of the systems by preparing them for realistic and challenging conditions. To evaluate the robustness of a system, the multifactorial performance analysis and sensitivity analyses are newly introduced, which are specifically designed to analyse the reaction of the networks to different types of disturbances in detail.

The analysis in this thesis is carried out by using fusion object detectors on different data types. This broad data basis demonstrates the versatile applicability and relevance of the developed training and evaluation concept for various sensor-based technologies. The Complex-YOLO algorithm is used specifically for analysing the point cloud data, which is adapted and applied here for the first time for both low-level and high-level fusion. This extension enables the fusion of data from LiDAR and RADAR sensors with Complex-YOLO.

The results of this work confirm the initial hypothesis and emphasise the importance of carefully considering the amount of disturbance in the training dataset. It is recommended that in future implementations of fusion networks, perturbation should be considered as an integral part of the training process to promote the adaptive capacity of the networks and ensure lasting robustness. The dissertation also shows that different types of perturbations pose different challenges and that robustness is significantly influenced by the type of data fusion. For example, it shows that low level fusion achieves better robustness against sensor interference than high level fusion.

This research provides valuable insights into the dynamics and mechanisms that influence the robustness of sensor fusion systems to sensor perturbation. The systematic integration of disturbances into the training process with KISA represents an effective approach to increase robustness against sensory disturbances. The dissertation thus offers not only a theoretical contribution to research in sensor fusion, but also a concept for the development of more robust systems in sensor evaluation applications.

Kurzfassung

In der vorliegenden Dissertation wird ein neues Konzept vorgestellt, welches dazu dient die Robustheit eines Multisensorsystem gegenüber Störungen in den Sensordaten zu steigern. Angesichts der unvermeidbaren Sensorstörungen in realen Anwendungsszenarien konzentriert sich diese Arbeit darauf, wie Sensorfusionssysteme zur Objektdetektion durch die Integration von Störungen während des Trainingsprozesses robuster gemacht werden können.

Für die Steigerung der Robustheit werden zwei Hauptschritte formuliert. Im ersten wird die Platzierung der Sensoren betrachtet. Dafür werden zwei neue Methoden vorgestellt: Genetische Algorithmus-Optimierung für Sensorplatzierung (GAOS) und Deep Learning-Optimierung für Sensorplatzierung (DLOS). Beide Methoden optimieren die Platzierung von Sensoren auf einem Fahrzeug, um das Wahrnehmungsfeld der kombinierten Sensoren zu maximieren. Gleichzeitig wird die Redundanz der Sensoren gegenüber Sensorausfall betrachtet. Im nächsten Schritt wird ein neues Konzept zur Robustheitssteigerung von Sensorfusionsmethoden vorgestellt. Dieses besteht aus einem neuen Trainingskonzept: Konzept zur Integration von Störungs-Adaptivität (KISA) und Evaluationsmethodiken, die die Steigerung der Robustheit durch Multifaktorielle Performanz-Evaluation und Sensitivitätsanalyse darstellen. Diese Konzepte behandeln die zentrale Hypothese dieser Arbeit. Die Hypothese lautet, dass die Integration von Störungen während des Trainingsprozesses die Robustheit von Fusionsnetzwerken gegenüber Sensorstörungen verbessert. Um diese Hypothese zu überprüfen, werden spezifische Forschungsfragen adressiert, die sich auf die Auswirkungen von gestörten Trainingsdaten auf die Robustheit von Fusionsnetzwerken, den Einfluss verschiedener Störungsarten auf das Trainingsergebnis, den Vergleich von Low Level und High Level Fusionstechniken und die Tiefe der Netzwerkarchitekturen konzentrieren.

Im neu entwickelten Trainingskonzept KISA werden gezielt Störungen während des Trainingsprozesses von Sensorfusionssystemen einbezogen. KISA ermöglicht es, die Robustheit der Systeme systematisch zu steigern, indem es die Systeme auf realistische und herausfordernde Bedingungen vorbereitet. Zur Bewertung der Robustheit eines Systems wird die Multifaktorielle Performanz-Evaluation und Sensitivitätsanalyse eingeführt, die speziell darauf ausgelegt sind, die Reaktion der Netzwerke auf verschiedene Störungsarten detailliert zu analysieren.

Die Analyse in dieser Arbeit wird durch den Einsatz von Fusionsobjektdetektoren auf unterschiedlichen Datentypen durchgeführt. Diese breite Datengrundlage demonstriert die vielseitige Anwendbarkeit und Relevanz des entwickelten Trainings- und Evaluationskonzepts für verschiedene sensorbasierte Technologien. Speziell für die Analyse von Punktwolkedaten wird der Complex-YOLO Algorithmus verwendet, der hier erstmalig sowohl für Low Level als auch High Level Fusion adaptiert und angewendet wird. Diese Erweiterung ermöglicht die Fusion von Daten aus LiDAR- und RADAR-Sensoren mit Complex-YOLO.

Die Ergebnisse dieser Arbeit bestätigen die anfängliche Hypothese und heben die Bedeutung einer sorgfältigen Abwägung des Störungsanteils im Trainingsdatensatz hervor. Es wird empfohlen, in zukünftigen Implementierungen von Fusionsnetzwerken Störungen als integralen Bestandteil des Trainingsprozesses zu betrachten, um die adaptive Kapazität der Netzwerke zu fördern und dauerhafte Robustheit zu gewährleisten. Die Dissertation zeigt auch, dass verschiedene Störungsarten unterschiedliche Herausforderungen darstellen und dass die Robustheit signifikant von der Art der Datenfusion beeinflusst wird. So zeigt sich, dass Low Level Fusion eine bessere Robustheit gegenüber Sensorstörung erreicht als High Level Fusion.

Diese Forschungsarbeit liefert wertvolle Einsichten in die Dynamik und die Mechanismen, die die Robustheit von Sensorfusionssystemen gegenüber Sensorstörung beeinflussen. Die systematische Integration von Störungen in den Trainingsprozess mit KISA stellt einen effektiven Ansatz dar, um die Robustheit gegen sensorische Störungen zu erhöhen. Die Dissertation bietet somit nicht nur einen theoretischen Beitrag zur Forschung in der Sensorfusion, sondern

auch ein Konzept für die Entwicklung robusterer Systeme in Sensorauswertungs Anwendungen.

Danksagung

An erster Stelle möchte ich mich bei meinem Doktorvater, Herrn Prof. Dr. Markus Reischl, für die hervorragende Betreuung dieser Dissertation bedanken. Ich danke ihm für seine kontinuierliche Unterstützung sowie seine wertvollen Ratschläge und Anregungen während der gesamten Promotionszeit. Weiterhin danke ich Herrn Prof. Dr. Marcus Geimer, der als Koreferent und durch den konstruktiven Austausch diese Arbeit bereichert hat. Ebenso möchte ich mich bei Herrn Prof. Dr. Stefan Elser bedanken, der mir während der gesamten Zeit als Vorgesetzter zur Seite stand und dabei nie das Menschliche vergessen hat.

Des Weiteren danke ich meinen Kolleginnen und Kollegen an der Hochschule Ravensburg-Weingarten und insbesondere am Institut für Künstliche Intelligenz (Prof. Dr. Markus Schneider, Prof. Dr. Wolfgang Ertel, Benjamin Stähle, Christopher Bonenberger, Sarah Weiß, Maik Knof, Samuel Hafner, Tobias Niedermaier, Ankita Agrawal, Benjamin Kathan, Stephan Scholz) für die angenehme Arbeitsatmosphäre und die spannenden Diskussionen.

Nicht zuletzt danke ich meiner Frau Pia für ihre Geduld und ständige Ermutigung. Ihre Unterstützung hat mir die Motivation gegeben, dieses Projekt abzuschließen. Durch ihre Liebe und ihr Verständnis war sie mir während der gesamten Promotionszeit eine unverzichtbare Stütze. Auch meinen lieben Eltern Elisabeth und Stefan, meinen Geschwistern Friederike, Maximilian und Ida-Maria, meinen Schwägern Sophie und Julius sowie meinen Schwiegereltern Claudia und Alexander möchte ich meinen tief empfundenen Dank aussprechen. Durch ihre stete Unterstützung und ihr unerschütterliches Vertrauen haben sie maßgeblich dazu beigetragen, dass ich diese Arbeit realisieren konnte.

Inhaltsverzeichnis

Anhang

Nomenklatur

Römische Symbole

a	Anzahl zusätzlicher Information eines Punktes
$a_{\min}$,$a_{\max}$	Minimaler/ Maximaler Wert der die zusätzlichen Information erreichen kann
$\mathbf{B}$	mehrdimensionales Array zur Darstellung eines Bildes
B	Menge der Bounding Boxen die je Gitterzelle beim Complex-YOLO Algorithmus geschätzt werden
$\mathbf{B}_R, \mathbf{B}_G, \mathbf{B}_B$	Farbkomponenten Rot, Grün und Blau eines Bildes
b_{versch}	Verschiebefaktor bei der Störung "Punkte verschieben"
$(b_x, b_y, b_w, b_h, b_\alpha)$	Bounding Box Werte beim Complex-YOLO Algorithmus
$c_{1,2}$	Konstanten zur numerischen Stabilität bei SSMI
C	Menge aller Klassenlabels
D	Datenraum
d	Distanz zwischen zwei Punkten
f	Evaluationsmetrik
O	Objektraum
$\mathbf{ED}$	Evaluationsdaten
f	physikalische Brennweite
$F_{p,\mathbf{S}}$	Menge der Evaluationsergebnisse eines Detektors trainiert mit einem Anteil p der Störungen aus $\mathbf{S}$
G	Gridzellen beim YOLO Algorithmus

g_S	Gewicht der Störung S
K	Konfidenzwert
$\mathbf{K}$	Segmentierungkarte
$k_{1,2}$	Steilheit der doppelt logistischen Funktion
l, w, h	Länge, Weite, Höhe
L	obere Schranke der doppelt logistischen Funktion
logit	Doppelt logistische Funktion
$\mathbf{M}$	Transformationsmatrix
$M_p(\mathbf{S})$	Multifaktorielle Performanz eines Detektors trainiert mit einem Anteil p der Störungen aus $\mathbf{S}$
m_{logit}	Maximum der doppelt logistischen Funktion
$\mathbf{N}$	Menge der natürlichen Zahlen
n_c	Anzahl an Kanälen eines Bildes
n_h, n_b	Höhe und Breite eines Bildes in Pixeln
n_k	Anzahl der Klassen
n_P	Anzahl der Punkte in einer Punktwolke
n_S	Anzahl an Störungen
p	Anteil an eingeführten Störungen
p_{max}	Anteil bei der die Maximale multifaktorielle Performanz erreicht wird
$\mathscr{o}$	Detektionsobjekt
p_{satt}	Anteil bei der die Sättigung erreicht wird
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbf{R}$	Rotationsmatrix
$ROI_{x,y,z}$	Gibt die Maße der Region von Interesse an
$\mathbf{S}$	Menge von Störungen
S	Eine Störungen
s_u, s_v	optischer Mittelpunkt
$\mathbf{t}$	Translationsvektor

$(t_x, t_y, t_w, t_h, t_{im}, t_{re})$	Prior-Werte einer Bounding Box beim Complex-YOLO Algorithmus
TD	Trainingsdaten
$\mathcal{V}_{\text{unten, oben}}$	Gleichverteilte Zufallsvariable
u, v	Bild Koordinaten
v_r	relative radiale Geschwindigkeit beim RADAR-Sensor
V	Klassifikationsvektor
x, y, z	räumliche Koordinaten
$x_{1,2}$	Wendepunkt der doppelt logistischen Funktion

Griechische Symbole

α	Parameter für die Länge des Pfades bei der Mutation in OSGA
β	Parameter für die Veränderung der Rotation bei der Mutation in OSGA
Δt	Laufzeitdifferenz
κ_x, κ_y	Skalierungsfaktoren
θ	Scherungsfaktor
Φ	Punktwolke
σ	Sigmoidfunktion
$\mu_{\mathbf{B}}$,$\sigma_{\mathbf{B}}$	Mittelwert und Varianz der Bildinformation

Index Zugriff

$A[i, :]$	i-te Zeile eines Arrays
$A[:, j]$	j-te Spalte eines Arrays
$A[i, j]$	Element i,j in einem Array
$\Phi[i]$	i-ter Punkt aus einer Punktwolke Φ

$\Phi[i,j]$	j-ter Wert des i-ten Punktes aus einer Punktwolke Φ

1 Einleitung

1.1 Motivation und Zielsetzung

In einer zunehmend technologiegetriebenen Welt spielt die Erfassung der Umwelt eine entscheidende Rolle in einer Vielzahl von Anwendungen, vom autonomen Fahren über die medizinische Bildgebung bis hin zur industriellen Automatisierung. Die hierfür benötigten Sensoren sammeln kontinuierlich Daten aus ihrer Umgebung, um präzise und zuverlässige Informationen für Entscheidungsprozesse zu liefern. Allerdings sind die von Sensoren gesammelten Daten oft unvollständig, unsicher und anfällig für verschiedene Arten von Störungen, die ihre Zuverlässigkeit und Genauigkeit beeinträchtigen können. Um diese Herausforderungen zu überwinden, ist die Integration mehrerer Sensorquellen durch Techniken der Sensorfusion unabdingbar geworden. Sensorfusion strebt an, durch die Kombination von Daten aus mehreren Quellen eine genauere und robustere Wahrnehmung der Umwelt zu erzielen. Doch die Robustheit dieser Fusionssysteme gegenüber Störungen bleibt eine zentrale Herausforderung, die die Zuverlässigkeit der darauf basierenden Technologien direkt beeinflusst. Weiterhin ist bei vielen Datenmodalitäten nicht geklärt, wie sie fusioniert werden sollen, da die Datenmodalität nicht passt.

Ein besonders relevantes Beispiel für den Einsatz von Sensorfusionssystemen ist das autonome Fahren. Autonomes Fahren stellt eine der fortschrittlichsten Anwendungen der modernen Technologie dar, mit dem Ziel, Fahrzeuge in einem hochdynamischen Verkehrsumfeld völlig autonom zu steuern. Diese Technologie verspricht nicht nur eine Revolutionierung des Straßenverkehrs, sondern könnte auch signifikante Verbesserungen in den Bereichen Verkehrssicherheit, Effizienz und Komfort bringen. Der Erfolg des autonomen Fahrens hängt jedoch entscheidend von der Fähigkeit ab, präzise und zuverlässige

Umgebungswahrnehmung zu gewährleisten – ein Ziel, das durch die Fusion von Daten verschiedener Sensoren wie LiDAR, RADAR und Kameras erreicht werden soll. Trotz dieser Fortschritte ist die Wahrnehmung autonomer Fahrzeuge weiterhin mit Herausforderungen konfrontiert, insbesondere durch Umweltbedingungen wie Nebel, der die Qualität der erfassten Daten erheblich beeinträchtigen kann. Während Nebel die Genauigkeit von Kameras und LiDAR-Sensoren negativ beeinflusst, bleiben RADAR-Sensoren davon weitgehend unbeeinträchtigt. Diese Störanfälligkeiten einzelner Sensoren können die Gesamtwahrnehmung des Systems gefährden, wenn keine geeignete Maßnahmen zur Anpassung an solche Bedingungen getroffen werden.

Die vorliegende Dissertation widmet sich der kritischen Analyse von Sensorfusion und liefert ein Konzept, insbesondere zur Steigerung der Robustheit von Sensorfusionssystemen gegenüber verschiedenen Störungsarten. Angesichts der Tatsache, dass Sensorstörungen in realen Anwendungsszenarien unvermeidbar sind, befasst sich diese Arbeit mit der Frage, wie Sensorfusionssysteme entwickelt werden können, damit sie auch unter suboptimalen Bedingungen zuverlässig funktionieren. Dafür werden zwei im Rahmen der Arbeit neu entwickelte Methoden zur Sensorplatzierung vorgestellt: Genetische Algorithmus-Optimierung für Sensorplatzierung (GAOS) und Deep Learning-Optimierung für Sensorplatzierung (DLOS). Außerdem werden ein neues Trainingskonzept: Konzept zur Integration von Störungs-Adaptivität (KISA) und neue Evaluationsmethodiken eingeführt: Multifaktorielle Performanz-Evaluation und Sensitivitätsanalyse. Diese Methoden zielen darauf ab, Sensorfusionssysteme robuster zu machen und diese Robustheit zu quantifizieren.

Durch die Fokussierung auf diese Aspekte trägt diese Dissertation nicht nur zur theoretischen Forschung in der Sensorfusion bei, sondern bietet auch Konzepte für zukünftige Arbeiten, wie die Robustheit von Sensorfusionssytemen gegenüber Sensorstörung gesteigert werden kann.

1.2 Sensormodalitäten: Kategorien und Einsatzgebiete

In diesem Kapitel werden die verschiedenen Sensormodalitäten, ihre spezifischen Kategorien und die jeweiligen Einsatzgebiete vorgestellt, die für die Umfelderkennung in autonomen Systemen von zentraler Bedeutung sind.

Sensormodalität bezieht sich auf die Art des Sensors, der zur Erfassung von Daten aus der Umgebung verwendet wird. Zu den verschiedenen Sensormodalitäten gehören z.B. bildgebende (z. B. visuelle Kamera, Infrarotkamera), räumliche (z.B. LiDAR, RADAR-Sensoren), auditive (z. B. Mikrofone), chemische (z. B. olfaktorische Sensoren [Wu11, Du13, Jung19]) Sensoren. Die Wahl der Sensormodalität hängt von der Anwendung und den spezifischen Informationen ab, die erfasst werden müssen.

Zur Umfelderkennung werden in autonomen Fahrzeugen vor allem bildgebende und räumliche Sensoren eingesetzt [Saka18, Shee21, Reic21, Wang21, Varg21, Igna22]. Es gibt auch Versuche mit auditiven Sensoren [Nand16, Marc21, Sun21], in dieser Arbeit werden jedoch nur bildgebende und räumliche Sensoren betrachtet.

Eine visuelle Kamera erfasst Bilder innerhalb des sichtbaren Bereich des elektromagnetischen Spektrums, welche Wellenlängen von etwa 380 nm (entspricht der Farbe Violett) bis zu 780 nm (entspricht der Farbe Rot) umfasst. Diese Fähigkeit der Kamera ermöglicht beispielsweise die Erkennung und Klassifizierung von Fahrbahnmarkierungen [Zhen22] sowie Verkehrszeichen [Chen22], was für Anwendungen wie automatisiertes Fahren und Verkehrsanalyse von großer Bedeutung ist. Die Fähigkeit, visuelle Merkmale präzise zu identifizieren und zu interpretieren, sowie ein geringer Stückkostenpreis machen visuelle Kameras zu einem unverzichtbaren Werkzeug in modernen Sensorsystemen. Infrarotkameras nutzen die Infrarotstrahlung, mit Wellenlängen von etwa 3500 nm bis zu 15000 nm, die von allen Objekten abhängig von ihrer Temperatur ausgestrahlt wird. Diese Wellenlänge ermöglicht es, in der Dunkelheit oder bei schlechten Sichtverhältnissen wie Nebel oder starkem Regen zu sehen, ohne auf sichtbares Licht angewiesen zu sein [Lonn05, Bast20].

Zur Entfernungsmessung werden häufig Ligth Detection and Ranging (LiDAR)- und Radio Detection and Ranging (RADAR)-Sensoren eingesetzt. LiDAR-Systeme operieren primär im nahen Infrarotbereich des elektromagnetischen Spektrums, mit Wellenlängen zwischen etwa 800 nm und 1.550 nm [Royo19], während RADAR-Sensoren im Radiofrequenzbereich mit Wellenlängen von 1 mm bis zu 1 m agieren [Brud03]. LiDAR-Sensoren für Fahrerassistenzsysteme, wie der Velodyne Puck, verwenden elektromagnetische Wellen mit einer Wellenlänge von 905 nm und damit im Infrarotbereich [Rabl19]. RADAR-Sensoren für Fahrerassistenzsysteme verwenden dagegen längere elektromagnetische Wellen im Mikrowellenbereich von 12,5 mm ($\widehat{=}$ 24 GHz) für Nahbereichsanwendungen und 3,9 mm ($\widehat{=}$ 76-77 GHz) für Fernbereichsanwendungen [Wald21]. Neu entwickelte RADAR-Sensoren verwenden Wellen im Bereich von 3,7 mm ($\widehat{=}$ 81 GHz [Devi17, Comm17, Wald21]) (z.B. ZF Full-Range Radar [1], Bosch Frontradarsensor [2]). Das grundlegende Messprinzip ist bei LiDAR- und RADAR- Sensoren gleich: Der Sensor erzeugt die elektromagnetische Welle und sendet sie aus, diese wird von einem Objekt reflektiert und die reflektierte Welle wird vom Sensor wieder empfangen. Aus der Laufzeitdifferenz Δt kann die Entfernung d zum Objekt bestimmt werden:

$$d = \frac{c\Delta t}{2},$$

wobei c die Lichtgeschwindigkeitskonstante (in der Luft beträgt die Lichtgeschwindigkeit $\approx 2{,}997 \cdot 10^8 \mathrm{m/s}$ [Even72]) ist. LiDAR-Sensoren messen die Intensität des zurückgeworfenen Lichtsignals, während RADAR-Sensoren die Magnitude des reflektierten Signals erfassen. Bei RADAR-Sensoren können zudem Geschwindigkeiten relativ zum RADAR-Sensoren durch Ausnutzung des Dopplereffektes gemessen werden.

Für einen Vergleich zwischen diesen beiden Sensoren sei auf den Anhang A verwiesen.

[1] https://www.zf.com/products/de/cars/products_64255.html, aufgerufen am 31. Januar 2024
[2] https://www.bosch-mobility.com/en/solutions/sensors/front-radar-sensor/, aufgerufen am 25. Juli 2024

1.3 Sensorstörungen und ihre Auswirkungen auf Bilddaten und Punktwolken

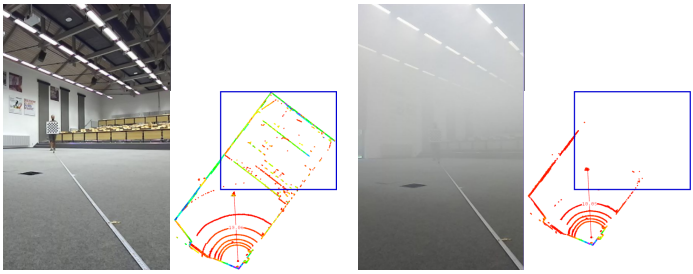
Sensorstörungen können die Qualität der Daten von bildgebenden und räumlichen Sensoren erheblich beeinträchtigen. In diesem Kapitel werden verschiedene Sensorstörungen und passende Datensätze, welche speziell bei Bilddaten von Kameras oder Punktwolken von LiDAR- oder RADAR-Sensoren vorkommen, vorgestellt.

1.3.1 Störungsquellen bei bildgebenden Sensoren

Bildgebende Sensoren sind anfällig für eine Vielzahl von Störungen, die die Qualität und Genauigkeit der erfassten Informationen beeinflussen können. Bei Kameras, die beim autonomen Fahrens eingesetzt werden, schließen diese Störungen unter anderem ein:

- Schlechte Lichtverhältnisse bedingt durch unzureichende oder übermäßige Beleuchtung [Yu20, Saka21, Sun20, Zhan15, Nugr17],
- Ungünstige Wetterbedingungen wie Nebel, Regen und Schnee [Shee21, Pitr21, Saka21],
- Kamerafehler: Defekte Photosensitive Zellen oder Linsen, Signalstörungen [Sing14, Mait18].

Für schlechte Lichtverhältnisse sind mehrere Datensätze veröffentlicht worden, die sich insbesondere auf unzureichende Beleuchtung in der Nacht konzentrieren [Yu20, Saka21]. Obwohl kein spezifischer Datensatz für Überbelichtung bekannt ist, enthalten einige Datensätze einzelne Aufnahmen, die Überbelichtung darstellen. Schlechte Lichtverhältnisse erschweren das Erkennen und Lokalisieren von z. B. Verkehrsteilnehmern, da die Kamera-Sensoren unter diesen Bedingungen weniger Licht empfangen und somit Konturen und Details der Objekte weniger deutlich sichtbar sind [Schu20, Schu21, Shee21, Wang21, Igna22, Varg21].



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

Abbildung 1.1: Visuelle Darstellung der Auswirkungen von Nebel auf die ZED2i Kamera 1.1a und 1.1c und den Velodyne-Puck-Sensor 1.1b und 1.1d. Die Intensität des Velodyne ist farbkodiert (rot = schwache Reflexionen, blau = starke Intensität). Der blaue Kasten bei den LiDAR Daten verdeutlicht, wo am meisten Punkte verloren gingen.

Auch Wetterbedingungen, wie Nebel, Regen und Schnee erschweren es auf Kamerabildern Objekte zu erkennen. Diese Wetterbedingungen streuen das Licht oder lassen Teilchen wie Schnee oder Regentropfen die Sichtbarkeit beeinträchtigen. In Abbildung 1.1 ist der Einfluss von Nebel auf das Bild einer Kamera beispielhaft dargestellt. Der Effekt durch Nebel lässt sich im besonderen auf die Mie-Streuung zurückführen [Hasi17]. Die Mie-Streuung, beschreibt die Streuung von Licht durch Partikel, die etwa in der Größenordnung der Wellenlänge des Lichts liegen. Dies ist bei Nebel und sichtbarem Licht der Fall. Nebeltröpfchen haben einen Durchmesser von etwa 5000 nm, was in der Größenordnung der Wellenlänge des sichtbaren Lichts (380 nm bis 780 nm) liegt [Hasi17]. Regentropfen und Schneekristalle sind deutlich größer als die Wellenlänge sichtbaren Lichts. Die Wechselwirkung elektromagnetischer Wellen mit Teilchen, die viel größer als die einfallende Wellenlänge sind, kann mit Hilfe der geometrischen Optik und der Beugung beschrieben werden [Hasi17].

Neben diesen Störungen, welche von außen auf die Kamera einwirken, gibt es auch solche, welche durch technische Defekte entstehen. So ist ein Problem das Vorhandensein fehlender oder verrauschter Daten aufgrund von Sensor- oder Signalstörungen sowie Datenübertragungsfehlern [Sing14, Mait18]. Für diese Art von Störung sind keine Datensätze bekannt. In [Sing14] werden

verschiedene Störungen dieser Art beschrieben und die Möglichkeit aufgezeigt, diese künstlich auf den Bildern zu simulieren.

In der medizinischen Bildgebung treten ähnliche Störungen auf, die die Qualität von Magnetresonanztomographie-Scans beeinflussen können. Ein Beispiel hierfür sind Bewegungsartefakte, die durch die Bewegung des Patienten während der Aufnahme verursacht werden [Jezz99, Zait15].

1.3.2 Störungsquellen bei LiDAR-Sensoren

LiDAR-Sensoren senden elektromagnetische Wellen mit einer Wellenlänge im Bereich von 800 nm bis 1.550 nm aus, im automotive Bereich werden häufig Sensoren mit 905 nm verwendet. Da diese Wellenlänge in der Größenordnung des sichtbaren Lichts liegt, haben LiDAR-Sensoren auch ähnliche Störungen wie Kameras:

- Starke Lichtquellen [Peti15, Park23],
- Ungünstige Wetterbedingungen: Nebel, Regen und Schnee [Peyn09, Ijaz12, Bije18, Pitr21],
- Vibration [Liu21, Schl22].

Starke externe Lichtquellen können einen LiDAR-Sensor stören, da diese den LiDAR-Sensor überlichten [Park23]. Diese Überlichtung kann zum Beispiel durch starkes Sonnenlicht, andere LiDAR-Sensoren oder durch sogenannte Spoofing-Attacken erfolgen [Peti15]. [1] Fehlendes externes Licht ist hingegen für einen LiDAR-Sensor kein Problem, da er als aktiver Sensor selbst die elektromagnetische Welle aussendet.

Störungen durch Nebel, Schnee, Regen und andere kleine Schwebeteilchen können durch die gleichen Effekte, Mie-Streuung, geometrische Optik und Beugung beschrieben werden, die auch für die Störung von Kamerabildern verantwortlich sind [Rass11, Hasi17]. Mehrere Untersuchungen haben diese

[1] Bei Spoofing-Angriffen wird gezielt versucht, LiDAR-Sensoren mittels Laser zu stören, um falsche oder irreführende Informationen zu generieren.

Effekte experimentell untersucht [Peyn09, Ijaz12, Bije18, Pitr21]. Die Ergebnisse dieser Studien zeigen, dass sowohl die maximale Reichweite als auch die Intensität der LiDAR-Punktwolke bei Nebel, Schnee und Regen abnehmen. Interessant ist bei einem LiDAR-Sensor mit einer Wellenlänge von 905 nm, dass sogar Aerosole wie Abgase detektieren werden, die im Kamerabild nur schwach sichtbar sind. Dieser Effekt wird auch zur Luftqualitätskontrolle verwendet [Rich94]. In Abbildung 1.1 ist zu sehen, dass der Velodyne-Puck eine Menge von Daten verliert, insbesondere ab einer Distanz von 10 m, was durch den blauen Kasten hervorgehoben wird. Die Anzahl der Punkte in der Punktwolke sinkt von durchschnittlich 28.169 auf 24.114 Punkte - 14,39% pro Bild. Die mittlere Intensität aller Frontpunkte verringert sich um -58,38%.

Vibrationen haben einen wesentlichen Einfluss auf die Qualität von LiDAR-Daten. Laut einer Studie von Hongchao Ma und Jianwei Wu [Ma12] führt die Plattforminstabilität, wie sie bei luftgestützten LiDAR-Systemen auftritt, zu einer Reihe von Positionierungsfehlern. Diese Fehler werden durch Plattformvibrationen verursacht und beeinträchtigen die Genauigkeit der durch LiDAR-Sensor erfassten Punktwolken. Des Weiteren zeigt die Studie, dass die durch Plattformvibration verursachten Fehler nicht durch einfache polynomische Funktionen angepasst werden können. In Schlager et al. [Schl22] wurden kontrollierte Vibrationsversuche mit einem automotiv LiDAR-Sensor durchgeführt. Schlager et al. stellten fest, dass Vibrationen einen Informationsverlust in der Punktwolke verursachen, der unabhängig von der Frequenz und Beschleunigung der Vibration ist. Die Forschung zeigte auch, dass die Interne Trägheitsmessungseinheit (IMU) des LiDAR-Sensoren bis zu einer Frequenz von 50 Hz ausreicht, um Vibrationen zu erkennen, während für höhere Frequenzen externe IMUs erforderlich waren.

1.3.3 Störungsquellen bei RADAR-Sensoren

RADAR-Sensoren senden elektromagnetische Wellen mit einer Wellenlänge im Bereich von 1 mm bis zu 100 m aus. Moderne Automobil-RADAR-Sensoren verwenden hauptsächlich zwei Frequenzbänder 24 GHz (≙12,5 mm) und 76-77

GHz (≙3,9 mm), weshalb sie allgemein als robuster gegen Wetterbedingungen gelten, aber auch diese Sensoren haben spezielle Störungen:

- Ungünstige Wetterbedingungen: Nebel, Regen und Schnee haben einen leichten Effekt auf RADAR-Sensoren [Csur10, Bala16, Hasi17].
- Interferenzen durch weitere RADAR-Quellen, wie sie durch andere RADAR-Sensoren oder durch den Multi-Path-Effekt auftreten können [Kune12, Aydo20].

Da die elektromagnetischen Wellen von RADAR-Sensoren länger als die Wellen von LiDAR-Sensoren sind, werden die Wellen bei Nebel durch die Rayleigh-Streuung und bei Regen oder Schnee durch die Mie-Streuung beeinflusst. Die Rayleigh-Streuung beschreibt die Streuung elektromagnetischer Wellen an Teilchen, die deutlich kleiner als die Wellenlänge sind [Youn81, Hasi17]. Da die Rayleigh-Streuung schwächer bei längeren Wellenlängen ist, ist auch ihr Einfluss von Nebel auf RADAR-Sensoren geringer [Hasi17]. So haben Csurgai-Horváth et al. [Csur10] reale Untersuchungen zum Einfluss von Nebel auf RADAR-Wellen gemacht. Sie konnten dabei zeigen, dass das RADAR-Signal durch Nebel abgeschwächt wird, jedoch ist dieser Effekt erst auf sehr großen Distanzen (>1 km) relevant und ist damit für die Anwendung im autonomen Fahren vernachlässigbar. Untersuchungen von Balal et al. [Bala16] und Hasirlioglu et al. [Hasi17] konnten bestätigen, dass der Einfluss von Nebel auf die Qualität von RADAR-Daten im für das autonome Fahren relevanten Bereich (< 1 km) zu vernachlässigen ist. Regen und Schnee haben auf die Radarwellen einen Einfluss und können die Qualität der Daten beeinflussen. Da Regentropfen und Schneekristalle eine Größe aufweisen, die in etwa der Wellenlänge der Radarwelle entspricht, bilden sie zusätzliche Rückstreuziele. Die Überlagerung all dieser kleinen Rückstreuungen verursacht ein Rauschen. Mit zunehmender Entfernung und damit zunehmender Rücklaufzeit der elektromagnetischen Welle nimmt die Intensität der Rückstreuung ab, weswegen dieses Rauschen vor allem im Nahbereich Wirkung zeigt [Winn11, Stei21]. Auch führt Regen zu einer nassen Straßenoberfläche, wodurch ein Teil der Radarstrahlung absorbiert wird, was die Reichweite des RADAR-Sensors beeinflusst [Viik08]. [Zang19] et al. konnten in einer Simulation zeigen, dass

ein automotiv RADAR-Sensor eine um 45 % verringerte maximale Reichweite bei starkem Regenfall hat.

Interferenz durch andere RADAR-Sensoren treten auf, wenn mehrere Fahrzeuge in der Nähe RADAR verwenden; in solchen Fällen können sich die Radarwellen gegenseitig stören und zu Messfehlern führen [Kune12, Aydo20]. Neben den unabsichtlichen Interferenzen durch andere Verkehrsteilnehmer, können Interferenzen auch absichtlich in sogenannten Spoofing Attacks auftreten [Komi21]. Der Multi-Path-Effekt setzt ein, wenn Radarwellen nicht direkt zum Sensor reflektiert werden, sondern mehrfach reflektiert werden [Emad22]. Beide Effekte können dazu führen, dass das RADAR sogenannte Geisterobjekte anzeigt - Objekte, die in der Realität nicht existieren - oder dass bestimmte Objekte nicht erkannt werden können.

1.4 Objektdetektion: Definitionen und Methoden

Objektdetektion kann definiert werden als die Aufgabe, eine Region in den Daten um das detektierte Objekt herum zu identifizieren und die Art des Objekts (Klassenlabel) zu klassifizieren (z.B. Auto, Person, Hund). Allgemein kann ein Objektdetektor folgendermaßen definiert werden:

Definition 1.1. *Sei D ein Datenraum und sei E ein Objektraum. Ein Objektdetektor O ist definiert als eine Abbildung:*

$$O : D \to E.$$

Der Datenraum D enthält die Eingangsdaten, in denen Objekte identifiziert werden sollen, dies können Daten wie Bilder, Punktwolken oder andere sensorische Daten umfassen. Der Objektraum E hängt von der spezifischen Aufgabe des Objektdetektors ab:

- Bei der 2D-Bounding-Box-Detektion werden Objekte durch Rechtecke beschrieben. Damit besteht jedes Objekt aus Paaren von Koordinaten

und Dimension für die Bounding Boxen (aus dem Raum $\mathbb{R}^4$, repräsentierend x, y, Breite, Höhe, zusätzlich kann auch die Rotierung der Bounding Box angegeben werden) und zugehörigen Klassenlabeln aus der Menge C aller Klassenlabel. Damit ist der Objektraum die Potenzmenge von dem kartesischen Produkt: $E = \mathcal{P}(\mathbb{R}^4 \times C)$. Da 2D-Bounding-Box-Detektion gewöhnlich auf Bildern durchgeführt wird, sind die Lage und Dimension der Bounding Box beschränkt auf die Dimension des Bildes.

- Bei der 3D-Bounding-Box-Detektion werden Objekte durch Quader beschrieben. Damit besteht jedes Objekt aus Tupel von Koordinaten, Dimension und Orientierung für die Bounding Boxen (aus dem Raum $\mathbb{R}^9$, repräsentierend x, y, z, Länge, Breite, Höhe, Roll, Nick, Gier) und zugehörigen Klassenlabeln aus der Menge C aller Klassenlabel. Damit ist der Objektraum die Potenzmenge vom kartesischen Produkt: $E = \mathcal{P}(\mathbb{R}^9 \times C)$. Zur Vereinfachung wird häufig auf die Roll und Nick Rotation verzichtet [Geig13, Jana20].
- Bei der semantischen Segmentierung wird jedem Bildpunkt eines Bildes eine Klasse zugeordnet. Hier ist der Objektraum $E = C^{n_h \times n_b}$, wobei n_h,n_b die Höhe bzw. Breite des Eingangsbilds und C die Menge der Klassenlabel beschreiben [Jana20].

Data Augmentation wird häufig verwendet, um die Vielfalt der Trainingsdaten zu erhöhen, die Generalisierungsfähigkeit des Modells zu verbessern und eine Überanpassung durch Veränderung des Eingaberaums zu verhindern. Bei der Data Augmentation für Bilddatensätze werden unter anderem geometrische Transformationen verwendet, wie z. B. das Spiegeln oder Beschneiden, oder es wird auch Rauschen eingefügt.

Um die Leistung von Objektdetektoren zu bewerten, werden Evaluationsmetriken herangezogen. Diese Metriken sind quantitative Maße, die die Effizienz und Genauigkeit der Detektionsmethoden objektiv beurteilen.

Definition 1.2. *Eine Evaluationsmetrik f ist eine Funktion, welche einen Objektdetektor O und einen Datenraum D auf einen metrischen Wert in $\mathbb{R}$ abbildet.*

In diesem Kontext wird die Average Precision (AP) als Evaluationsmetrik zur Beurteilung der Leistung von 2D- und 3D-Objektdetektoren verwendet. Die Average Precision misst die Genauigkeit des Detektionsalgorithmus über verschiedene Schwellenwerte für die Intersection-over-Union (IoU) und gibt an, wie gut der Algorithmus im Durchschnitt detektiert. Die Übereinstimmung zwischen den vom Algorithmus vorgeschlagenen Bounding Boxes und den tatsächlichen Bounding Boxes wird mittels der IoU bewertet. Die Intersection over Union berechnet das Verhältnis der Überlappung zwischen der vorhergesagten und der tatsächlichen Bounding Box zur Gesamtfläche der beiden Boxen, was eine quantitative Bewertung der Genauigkeit der Lokalisierung bietet. Zur Bewertung von Ergebnissen der semantischen Segmentierung wird das F1-Maß verwendet, welches das harmonische Mittel zwischen Präzision und Recall bildet. Das F1-Maß kombiniert Präzision (der Anteil der korrekt vorhergesagten positiven Instanzen an allen vorhergesagten positiven Instanzen) und Recall (der Anteil der korrekt vorhergesagten positiven Instanzen an allen tatsächlichen positiven Instanzen) zu einer einzigen Metrik, die ein ausgewogenes Bild der Klassifikationsleistung vermittelt.

Für weitere Information zu aktuellen Objektdetektoren siehe Anhang C und für Berechnungsdetails der Evaluationsmetriken siehe Anhang D.

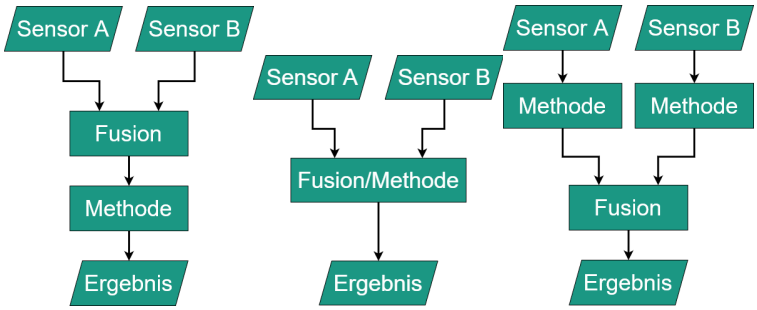
1.5 Sensorfusion: Definitionen und Methoden

Durch die Verwendung von mehreren Sensoren muss eine Strategie entwickelt werden, wie die Daten von mehreren Sensoren kombiniert werden, um ein umfassenderes Bild der Umgebung zu erhalten. Die Auswahl der Sensormodalität und der Sensorfusionsstrategie sind entscheidende Komponenten bei der Konzeption und Entwicklung effektiver Erfassungssysteme. Die Kombination von Sensordaten wird als Sensorfusion oder kurz als Fusion bezeichnet.

Ein notwendiger Schritt für Sensorfusionsmethoden ist die Kalibrierung. Methoden für die Kalibrierung von RADAR, LiDAR und Kamera werden in Anhang B dargestellt.

Im Folgenden werden die Konzepte zur Sensorfusion und beispielhafte Algorithmen für die jeweiligen Strategien dargestellt.

Die Wahl der geeigneten Sensorfusionsstrategie hängt von verschiedenen Faktoren ab, darunter die spezifische Anwendung, die vorhandenen Sensoren, die Qualität der Daten, die Leistungsanforderungen und im besonderen die Sensormodalitäten [Koci18, Feng20]. So erfordern homogene Daten, d. h. Daten, die in ihrer Modalität einheitlich und konsistent sind, eine andere Fusionsmethode als heterogene Daten mit unterschiedlichen Modalitäten.



(a) Low Level Fusion **(b)** Mid Level Fusion **(c)** High Level Fusion

Abbildung 1.2: Schaubild für die drei Fusionsstrategien und bei zwei Sensoren (Sensor A und Sensor B). Methode steht für einen beliebigen Objekterkennungsalgorithmus, der eine Objekterkennungsliste erzeugt.

Dieser Abschnitt widmet sich der Erörterung von drei Ansätzen der Sensorfusion: Low Level Fusion, Mid Level Fusion und High Level Fusion. Diese drei Ansätze sind in Abbildung 1.2 schematisch dargestellt. Die drei Ansätze der Sensorfusion unterscheiden sich in der Verarbeitungsstufe, auf der die Sensordaten kombiniert werden: Low Level Fusion integriert Rohdaten direkt von den Sensoren, Mid Level Fusion kombiniert bereits vorverarbeitete Merkmale, und High Level Fusion fusioniert die Entscheidungen oder Ergebnisse, die aus den einzelnen Sensoren abgeleitet wurden. Es ist zu beachten, dass die Kategorisierung eines Fusionsalgorithmus nicht immer eindeutig ist, da viele Strategien Aspekte mehrerer Ansätze integrieren können. Im Rahmen dieses

Abschnittes wird der theoretische Ansatz jeder Fusionsebene beleuchtet. Dabei werden die Vor- und Nachteile jedes Ansatzes dargestellt.

1.5.1 Low Level Fusion

Low Level Fusion ist ein wichtiger Ansatz, der auch als Rohdatenfusion oder Early Fusion bekannt ist. Hierbei geht es darum, die Rohdaten der verschiedenen Sensoren zu kombinieren und in den kombinierten Daten Objekte zu detektieren. In dieser Arbeit werden Punktwolken oder Bilder, die vom Sensor erfasst wurden, als Rohdaten betrachtet, auch wenn sie bereits einer gewissen Vorverarbeitung unterzogen wurden. Die Low Level Fusion eignet sich besonders gut für homogene Daten. Beispiele hierfür sind die Fusion von LiDAR-LiDAR-Daten oder LiDAR-RADAR-Daten.

Das Schaubild 1.2 stellt ein Low Level Fusionssystem dar, bei dem die rohen Daten von Sensor A und Sensor B kombiniert werden, um im Anschluss durch eine Methode ein Ergebnis zu erzeugen.

Ein Beispiel für Low Level Fusion, ist die Fusion von Punktwolken bei RADAR- und LiDAR-Daten. Hierfür müssen die jeweiligen Punktwolken in ein gemeinsames Koordinatensystem transformiert werden. Dadurch werden die jeweiligen Punktwolken zu einer einzigen, gemeinsamen Punktwolke fusioniert, die alle Informationen der ursprünglichen Sensoren enthält. Ein weiteres Beispiel für eine Low-Level-Fusion ist die Kombination eines RGB-Bildes mit einem Tiefenbild. Dabei werden die drei Kanäle des RGB-Bilds und der eine Kanal des Tiefenbildes zu einem vierkanaligen RGB-D-Bild vereint [Song17].

Nachdem die Fusion durchgeführt wurde, kann ein Detektionsalgorithmus verwendet werden, um die Objekte zu identifizieren.

Bei der Sensorfusion von RADAR- und LiDAR-Daten wird in der Regel die Low Level Fusion auf vorverarbeiteten Daten angewandt, wobei diese Vorverarbeitung oft die Umwandlung der rohen Eingangssignale der Sensoren in Punktwolken oder andere strukturierte Darstellungen umfasst, ohne dass eine weitergehende Merkmalsextraktion oder Objekterkennung stattgefunden hat [Koci18].

Die Vorteile der Low Level Fusion sind:

- Effiziente Implementierung: Durch die Transformation der Rohdaten in ein gemeinsames Datenformat können bereits bestehende Algorithmen, die auch für nicht fusionierte Daten verwendet werden, wiederverwendet werden.
- Schnelle Verarbeitung: Die Low Level Fusion arbeitet direkt auf den Rohdaten der Sensoren, wodurch alle Daten von einem Algorithmus ausgewertet werden können, was zu einer schnelleren Verarbeitung führt [Koci18, Feng20, Yeon21].
- Dichte Dateninformation: Die Fusion der Rohdaten ermöglicht eine höhere Dichte von Datenmerkmalen, wie Punkte in einer Punktwolke, wodurch bestimmte Merkmale, welche in den einzelnen Sensordaten versteckt bleiben, erkannt werden [Koci18, Feng20, Yeon21]. So kann die charakteristische geometrische Form eines Fahrzeuges erst aus der Kombination mehrerer Punktwolken ersichtlich werden.

Die Nachteile der Low Level Fusion sind:

- Geringe Robustheit: Die Low Level Fusion ist anfällig für Störungen, insbesondere wenn einer der Sensoren gestört wird. Dies kann dazu führen, dass die nachfolgende Methode, die auf der Fusion folgt, keine korrekten Ergebnisse liefert [Koci18, Feng20, Yeon21].

1.5.2 Mid Level Fusion

Mid Level Fusion, auch als Feature Fusion bekannt, konzentriert sich darauf, extrahierte Merkmale oder charakteristische Muster aus den Daten einzelner Sensoren zu vereinigen. Diese Merkmale können beispielsweise Texturinformationen, Farbmerkmale, Kanten oder andere relevante Eigenschaften sein, die für die spezifische Anwendung von Interesse sind. Im Gegensatz zur Low Level Fusion durchlaufen die Daten hier bereits eine gewisse Verarbeitung, wodurch Merkmale aus den Sensordaten fusioniert werden können. Dadurch lassen sich sowohl homogene als auch heterogene Daten effektiv zusammenführen.

Im Schaubild 1.2 wird ein Mid Level Fusionssystem dargestellt, bei dem die Daten von Sensor A und Sensor B zuerst gesammelt werden und dann durch eine kombinierte Fusion/Methodik-Einheit verarbeitet werden, um ein finales Ergebnis zu erzielen.

Ein Beispiel für Mid Level Fusion in der LiDAR-Kamera-Fusion sind die Ansätze ”Multi-Task Multi-Sensor Fusion” [Lian19] und ”Cont Fuse” [Lian18]. Bei beiden Ansätzen gibt es zwei Datenströme, den Kamera-Stream und den LiDAR-Stream. Mehrere nacheinander aufbauende Fusionsschichten werden verwendet, um die Merkmale der Kamerabilder mit den Merkmalen des LiDAR-Streams zu verschmelzen.

Die Vorteile der Mid Level Fusion sind:

- Leistungsstarker Merkmalsvektor enthält relevante Informationen, die zur Erkennung und Klassifizierung von Objekten oder Ereignissen dienen können [Koci18, Feng20, Yeon21].
- Verbesserte Erkennungsgenauigkeit: Durch die Mid Level Fusion können Merkmalsauswahlalgorithmen eingesetzt werden, um relevante Merkmale und Merkmalskombinationen zu erkennen. Dadurch kann die Erkennungsgenauigkeit verbessert werden, indem unwichtige Merkmale eliminiert und wichtige Merkmale hervorgehoben werden [Yeon21].

Die Nachteile der Mid Level Fusion sind:

- Große Trainingsdatensätze erforderlich: Die Mid Level Fusion erfordert oft umfangreiche Trainingsdatensätze, um Merkmale und Merkmalskombinationen zu identifizieren. Dies kann besonders in komplexen Szenarien oder bei der Verwendung vieler Sensoren eine Herausforderung sein, da die Beschaffung und Annotierung großer Datenmengen zeitaufwändig und kostspielig ist [Yeon21].
- Kompliziert zu skalieren: Das Netzwerk wird sehr komplex, wenn viele Sensoren miteinander fusioniert werden sollen. Zudem muss die Architektur des Netzwerks verändert werden, wenn ein Sensor hinzugefügt oder entfernt wird [Koci18, Feng20].

1.5.3 High Level Fusion

High Level Fusion, auch bekannt als Late Fusion, konzentriert sich auf die Kombination von abstrakteren Informationen und Konzepten. In dieser Arbeit sind diese Informationen detektierte Objekte, die aus den Daten der einzelnen Sensoren gewonnen werden. Hierbei durchlaufen alle Sensoren einzeln eine Objektdetektionsmethode und die Objektlisten werden anschließend fusioniert.

Im Schaubild 1.2 wird ein High Level Fusionssystem dargestellt, bei dem zuerst unabhängige Verarbeitungsmethoden auf die Daten von Sensor A und Sensor B angewandt werden, um höherstufige Informationen oder Entscheidungen zu treffen, die anschließend in einer zentralen Fusionseinheit zusammengeführt werden, um ein endgültiges Ergebnis zu erzielen.

Diese Fusion kann über klassische Algorithmen wie Non-Maximum Suppression oder Kalman-Filter oder auch neuronale Netzwerke wie bei der Methode Camera-LiDAR Object Candidates (CLOCs) [Pang20] erfolgen.

Die Vorteile der High Level Fusion sind:

- Geringere Komplexität und Ressourcenanforderungen: High Level Fusion erfordert weniger Rechenleistung, Kommunikationsressourcen und reduziert die Gesamtkomplexität im Vergleich zu anderen Ansätzen [Koci18, Yeon21].
- Einfache Optimierung der Fusion: Durch die Auswahl und Kombination bereits optimierter Methoden auf einzelnen Sensoren kann eine effiziente und leistungsstarke Fusion realisiert werden, die die besten verfügbaren Techniken nutzt [Pang20, Feng20].

Die Nachteile der High Level Fusion sind:

- Informationsverlust: Objekte oder Informationen mit niedriger Konfidenz werden häufig verworfen, was zu unvollständigen oder ungenauen Ergebnissen führen kann [Feng20, Yeon21].

1.6 Ansätze zur Sicherstellung der Robustheit in der Objekterkennung

In Anwendungen wie autonomes Fahren, Überwachungssysteme und Robotik spielt die Robustheit gegenüber Störungen bei der Objekterkennung eine wichtige Rolle. Im Mittelpunkt der vorliegenden Arbeit steht, dass Algorithmen nicht nur unter idealen Bedingungen, sondern auch bei ungültigen Eingaben oder in anspruchsvollen Umgebungsbedingungen zuverlässig funktionieren müssen. Gemäß der Definition des IEEE Standard Glossary of Software Engineering Terminology ist die Robustheit definiert als: "The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environmental conditions" [IEEE90]. Robuste Objektdetektionssysteme sind in zahlreichen Anwendungen unverzichtbar, sei es in der autonomem Fahrzeugtechnik, in der Videoüberwachung, der medizinischen Bildverarbeitung oder in Robotikanwendungen. In diesem Kapitel werden verschiedene Ansätze beschrieben, die Robustheit eines Systems zur Objekterkennung gegenüber Störungen zu erhöhen.

1.6.1 Strategien zur Verbesserung der Robustheit bei Bilddaten in der Objektdetektion

Um den Herausforderungen der Sensorstörung auf Bilddaten zu begegnen, werden in der Literatur zwei Ansätze vorgeschlagen: Daten vorverarbeiten und Fusion mehrerer Sensoren.

Ein Beispiel für die Erhöhung der Robustheit einer Methode ist die Vorverarbeitung der Daten durch die Entfernung von Rauschen aus den Bilddaten mit Hilfe von Filtern. Hierbei werden verschiedene Vorverarbeitungsschritte angewendet, um die Daten von Rauschen zu bereinigen. Hierzu gehören die Anwendung linearer, nichtlinearer oder adaptiver Filter [Soma12, Sing14, Mait18]. Lineare Filter, wie der Gauss-Filter und der Durchschnittsfilter, werden verwendet, um spezifische Arten von Rauschen zu entfernen. Diese Filter können jedoch Kanten verwischen und feine Details im Bild zerstören [Sing14]. Nichtlineare Filter, wie der Medianfilter, der Min-Filter und der Max-Filter,

wurden entwickelt, um diese Einschränkungen zu überwinden und zeigen in der Regel eine bessere Leistung bei der Rauschentfernung [Sing14]. Diese Ansätze mit einfachen Filtern sind jedoch limitiert auf einfaches Rauschen. Für komplexere Störungen, wie schlechte Lichtverhältnisse oder Dunst sind in den letzten Jahren Ansätze, welche Image-to-Image Translations Netzwerke verwenden, entwickelt worden, welche die jeweilige Störung herrausrechnen [Ki18, Schu20]. Schutera et al. [Schu20] verwenden für ihre Anwendung Unsupervised Image-to-Image Translation (UNIT) auf dem BDD100K Datensatz [Yu20]. UNIT wurde darauf trainiert, Nachtaufnahmen in Tagaufnahmen zu transformieren und umgekehrt. Evaluiert haben sie die Translation anhand der Performanz eines Objektdetektors, welcher nur auf Tagaufnahmen trainiert wurde. Die Performanz des Objektdetekors auf den transformierten Nachtaufnahmen konnte um 5,27 Prozentpunkte gesteigert werden, verglichen mit der Performanz auf den nicht transformierten Nachtaufnahmen [Schu20]. Um Dunst aus Bildern herauszurechnen haben Ki et al. [Ki18] eine Methode vorgeschlagen, welche auf der Kombination von Boundary Equilibrium Generative Adversial Network (BEGAN) und Conditional Generative Adversial Network (cGAN) basiert. Als Generator verwenden sie ein Netzwerk, welches auf U-Net [Ronn15] basiert. Das conditional BEGAN von Ki et al. [Ki18] hatte eine bessere Leistung Dunst zu entfernen als Pix2Pix, welches ein cGAN mit U-Net als Generator ist oder BEGAN mit U-Net als Generator.

Um die Störung eines Sensors zu begegnen, können mehrere Sensoren kombiniert werden. Bei der Kombination von gleichen Sensoren erzeugt man damit eine gewisse Redundanz, wodurch der Ausfall einzelner Sensoren kompensiert werden kann. Bei der Kombination von Sensoren mit unterschiedlichen Fähigkeiten können spezifische Schwächen eines Sensors gegenüber Aufnahmebedingungen ausgeglichen werden.

In einer Studie von [Sun20] wird vorgeschlagen, Daten von RGB-Kameras und Thermalkameras zu fusionieren, um damit die Schwäche der RGB-Kamera bei Nacht auszugleichen. Die vorgestellte Methode FuseSeg, verwendet eine Mid-Level Fusion, um diese Sensordaten zu fusionieren. Bijelic et al. [Bije20] betrachteten die Fusion von LiDAR, RGB-Kamera, Gated-Near-Infrared (NIR) und RADAR Sensoren bezüglich der Störung durch Nebel. Die Sensordaten

werden in das Kamerakoordinatensystem projiziert. Die Average Precision der 2D Bounding Box Detektion ist mit 81,49% auf den moderaten Objekten [1] bei dichtem Nebel 3,24% Prozentpunkte besser, als ein Netzwerk, welches nur auf den RGB Bilddaten arbeitet. Besonders interessant ist hierbei, dass es nur zu einem kleinen Performanzverlust kommt, so ist die Average Precision auf Aufnahmen ohne Störung mit 85,57% nur minimal kleiner. Dies liegt vor allem an den RGB-Bild und Gated-NIR Daten, da Netzwerke, die nur mit diesen Daten trainiert wurden, sogar eine bessere Performanz auf den Daten mit dichten Nebel aufwiesen (78,25% RGB, 66,76% NIR), als auf Daten ohne Störung (75,75% RGB, 61,95% NIR).

1.6.2 Strategien zur Verbesserung der Robustheit gegenüber Störungen in den Punktwolkedaten von Objektdetektoren

Bei 3D-Daten werden verschiedene Sensormodalitäten zu einem Gesamtsystem kombiniert, um die Robustheit zu erhöhen.

Ein Multisensorsystem kann durch Redundanz das System robuster gegen Sensorausfälle machen. Es gibt bereits einige Methoden [Mou18, Liu19, Dybe20b, Dybe20a], die das Problem der optimalen Sensoranordnung für 3D-Wahrnehmungssensoren betrachten. Diese Optimierungsmethoden müssen mehrere Herausforderungen berücksichtigen: die gleichzeitige Optimierung mehrerer Sensoren hinsichtlich Position und Rotation (Yaw, Pitch und Roll), die Minimierung toter Winkel, die Berücksichtigung von Redundanz sowie die Verdeckung durch die genaue Form des Fahrzeugs und der Sensoren. Dybedal und Hovland [Dybe20b, Dybe20a] schlugen beispielsweise eine Methode zur Optimierung der Konfiguration von 3D-Kameras vor. Diese Methode basiert auf der Unterteilung des Interessenbereichs in endliche kleine Quader. Für jeden Quader berechnet die Methode, ob er sich innerhalb des sichtbaren Bereichs der Sensoren befindet oder nicht. Hierdurch ist die Methode sehr

[1] Schwierigkeit der Objekte klassifiziert nach [Geig12a].

rechenintensiv. Weiterhing ist eine Übertragung der Methode auf LiDAR Sensoren ohne eine Erweiterung nicht möglich, da die Daten eines LiDAR-Sensors spärlicher sind als die einer 3D-Kamera und ein LiDAR-Sensor nicht vollständig durch die beiden Parameter Sichtfeld und Reichweite beschrieben wird. Mou et al. [Mou18] und Liu et al. [Liu19] haben Methoden für LiDAR Sensoren entwickelt. Bei beiden Methoden wird der interessierende Bereich auch in endliche kleine Quader unterteilt, und beide stellen die LiDAR Sensoren durch kegelförmige Wahrnehmungsbereiche dar. Diese Ansätze verwenden nichtlineare Optimierungsmethoden, und haben hohe Berechnungskosten, die mit der Anzahl der Quader und der Größe des Interessenbereichs steigen. Kim und Park [Kim19] schlugen eine Methode vor, welche auf der Simulation der LiDAR Sensoren basiert. Sie führten die LiDAR-Belegungskarte und das LiDAR-Belegungsraster ein: LiDAR Belegungskartes sind quadratische Tafeln, die um die LiDAR Sensoren herum platziert werden. Die LiDAR-Belegungskarte ist in Teilbereiche unterteilt, das LiDAR-Belegungsrasters. Die vorgeschlagene Zielfunktion besteht darin, die Anzahl der Teilbereiche zu maximieren, die von einem Punkt eines LiDAR-Sensors belegt sind, was als Total-LiDAR-Belegung bezeichnet wird. Um die optimale Lösung zu finden, verwendeten Kim und Park [Kim19] einen genetischen Algorithmus, bei dem die LiDAR Sensoren der Reihe nach vom ersten bis zum letzten LiDAR platziert werden. Der Ansatz hat den Vorteil, dass der LiDAR-Sensor simuliert wird und die Ergebnisse daher leicht auf die reale Welt übertragen werden können und nur davon abhängen, wie gut der Simulator implementiert ist. Allerdings optimiert die Methode nur die Position und nicht die Drehung des LiDAR-Sensors, und der Algorithmus platziert die Sensoren sequentiell. Für mehrere Sensoren ist die vorgeschlagene Konfiguration meist nicht optimal. Auch wird die Redundanz von Sensoren nicht berücksichtigt.

Um die Robustheit des Systems gegenüber spezifische Umwelteinflüsse zu erhöhen, werden verschiedene Sensormodalitäten in das Multisensorsystem integriert. Wie bereits in Kapitel 1.2 dargestellt, sind RADAR Sensoren besonders robust gegenüber Störungen durch Umwelteinflüsse, wie Nebel oder Schnee. Diesen Vorteil in der Robustheit von RADAR Sensoren verwenden Guan et al. [Guan20] und zeigten dies für künstlich erzeugten Nebel und Regen. Da RADAR Daten verglichen mit LiDAR Daten weniger detailliert sind, wird

in Guan et al. eine Methode namens HawkEye vorgestellt, welche aus den RADAR Daten eine detaillierte Tiefenkarte erstellt. Trainiert wurde HawkEye nur auf Daten ohne Störung. Bei der Evaluierung konnten die Autoren zeigen, dass HawkEye eine bessere Darstellung der Tiefenkarten liefern kann, als das RADAR es liefert, und auch bei Nebel genaue Tiefenkarten liefern kann.

Der RADIATE-Datensatz [Shee21] umfasst Daten von 2D-RADAR, LiDAR und Kamera, erfasst unter diversen Umweltbedingungen wie Nebel, Schnee, Regen und klarem Wetter. Sheeny et al. [Shee21] demonstrieren in ihrer Studie zum RADIATE-Datensatz zudem das Potenzial der Objektdetektion mittels RADAR-Daten unter umweltbedingten Störeinflüssen. Dafür wurde Faster R-CNN [Girs15] als Detektor verwendet, wobei er so angepasst wurde, dass Faster R-CNN auch die Orientierung der Bounding Box schätzt. Als Backbone für Faster R-CNN wurde sowohl ResNet-50 als auch ResNet-101 getestet [He16]. Um den Einfluss von umweltbedingten Störungen zu bestimmen, wurde Faster R-CNN einmal nur auf Daten ohne solche Störungen trainiert und einmal gemischt, sowohl mit Störungen, als auch ohne. Wie das Verhältnis zwischen Daten mit und Daten ohne Störung war, wurde nicht angegeben. Bei Aufnahmen im urbanen Umfeld ohne umweltbedingte Störungen hatte Faster R-CNN mit ResNet-101 eine Average Precision von 35,36%, wenn es mit den gemischten Daten trainiert wurde, und 30,36% bei den Daten ohne Störung. Die Daten mit Störung wurden alle im suburbanen Umfeld aufgenommen. Bei Nebel konnte eine Average Precision von 51,22% beim Training mit Störungen und 48,30% beim Training ohne Störungen erreicht werden. Bei Schnee konnte eine Average Precision von 8,14% beim Training mit Störungen und 11,16% beim Training ohne Störungen erreicht werden. Bei Regen konnte eine Average Precision von 31,96% beim Training mit Störungen und 29,18% beim Training ohne Störungen erreicht werden. Die Autoren führen die verminderte Leistungsfähigkeit bei Regen auf Störungen des RADARs durch Niederschlag zurück, wodurch sich die Werte der Hintergrundbildpunkte verändert haben. Als ein weiterer Grund wird die unterschiedliche Aufnahme von Szenen genannt. Die Aufnahmen, welche ohne Störung sind, wurden mit einem geparkten Auto aufgenommen. Die Autoren gehen nicht darauf ein, warum sich die Leistung des Objektdetektors bei Daten mit Schnee (11,16%) verschlechtert, wenn er mit Störung trainiert wird, im Vergleich dazu, wenn er ohne Störung trainiert wird (8,14%).

Qian et al. [Qian21] schlagen die Fusion von LiDAR und RADAR Daten vor. Sie entwickelten eine Mid-Fusion Objektdetektionsmethode namens MVDNet. MVDNet konvertiert die LiDAR Punktwolken in Bird's Eye View (BEV)-Bilder. Die RADAR Daten liegen bereits als BEV-Bilder vor. Zunächst werden räumliche Merkmalskarten von beiden Sensoren extrahiert, anschließend werden orientierte 2D-Vorschläge generiert und fusioniert, sowie regionsbasierte Merkmale von beiden Sensoren über ROI-Pooling extrahiert. Ein Fusion-Netzwerk wird verwendet, um die regionsbasierten Merkmale der beiden Sensoren zu kombinieren. Die fusionierten Merkmale werden verwendet, um Objekte gemeinsam zu erkennen und zu lokalisieren. Dadurch kann MVDNet die durch Nebel verdeckten Fahrzeuge erkennen, und auch Fehldetektionen in den verrauschten RADAR Daten zurückweisen. Dabei wurde zufällig auf 50% der Trainingsdaten Nebel simuliert. Dadurch konnte sowohl die Average Precision von MVDNet um 3,67 Prozentpunkte auf Daten ohne Nebel gesteigert werden und auf Daten mit Nebel um 9,42 Prozentpunkte [Qian21].

1.7 Offene Probleme

Zusammenfassend ergeben sich aus den vorangegangenen Abschnitten folgende offene Probleme für die Fusion mehrerer Sensordaten zur Objektdetektion:

1 *Keine effiziente Sensorplatzierungsmethode für Redundanz:*
Bisherige Methoden zur Optimierung der Sensorplatzierung sind sehr rechenintensiv und damit zeitaufwendig. Außerdem zielen sie nur auf die Optimierung des Erfassungsfeldes ab. Methoden, die auch die Redundanz der Sensoren im Hinblick auf Sensorausfälle berücksichtigen, fehlen.

2 *Fehlende Analyse des Einflusses von Störungsarten auf die Detektionsleistung:*
In der aktuellen Forschung zur Robustheit von Sensorfusionsmethoden wird nicht ausreichend zwischen den spezifischen Einflüssen verschiedener Störungsarten differenziert. Diese undifferenzierte Betrachtungsweise erschwert es, gezielte Konzepte zur Verbesserung

der Robustheit von Fusionsnetzwerken gegenüber spezifischen Störungen zu entwickeln.

3 *Fehlen von Evaluationsmethoden für die Robustheit:*
Es fehlen einheitliche und systematische Evaluationskonzepte zur zuverlässigen Beurteilung der Robustheit von Fusionsnetzwerken.

4 *Unzureichende Analyse der Adaptivität von Fusionsnetzwerken auf gestörte Trainingsdaten:*
Es fehlen systematische Untersuchungen darüber, wie Fusionsnetzwerke auf gestörte Trainingsdaten reagieren und in welchem Maße diese die Robustheit des Fusionsnetzwerk gegen diese Störung zu steigern.

5 *Fehlender Vergleich von Fusionsebenen:* In der Forschung mangelt es an einem umfassenden Vergleich von Low Level und High Level Fusionstechniken bezüglich der Robustheit gegenüber Sensorstörung.

1.8 Hypothese und Forschungsfragen

Basierend auf den Erkenntnissen der vorherigen Kapitel über den aktuellen Stand der Forschung wird folgende Hypothese aufgestellt:

Die Integration von Störungen während des Trainingsprozesses kann die Robustheit von Fusionsnetzwerken gegenüber Sensorstörungen verbessern.

Zur Überprüfung dieser Hypothese werden in dieser Arbeit die nachstehenden Forschungsfragen untersucht:

1 **Wie beeinflussen gestörte Trainingsdaten die Robustheit von Fusionsnetzwerken gegenüber der Störung?**
Diese Forschungsfrage zielt darauf ab zu untersuchen, wie sich die Einbeziehung von gestörten Daten in den Trainingsprozess von Fusionsnetzwerken auswirkt. Es wird analysiert, ob und inwieweit die Exposition gegenüber Störungen während des Trainings die Fähigkeit

des Netzwerks verbessert, unter Bedingungen, in denen Sensorstörungen auftreten können, effektiv zu funktionieren. Hierbei wird betrachtet, ob das Netzwerk nach dem Training mit gestörten Daten besser in der Lage ist, Fehler zu tolerieren und korrekte Ergebnisse zu liefern.

2 **Wie beeinflussen verschiedene Störungsarten das Trainingsergebnis?**
Im Mittelpunkt dieser Fragestellung steht, welche Störungsarten das Trainingsergebnis von Fusionsnetzwerken besonders beeinflussen. Durch die Gruppierung der betrachteten Störungen wird analysiert, inwieweit das Netzwerk fähig ist, mit einer Störungen umzugehen, wenn eine ähnlichen Störung in den Trainingsprozess integriert wurde.

3 **Welchen Einfluss haben Low Level und High Level Fusion auf das Trainingsergebnis?**
In dieser Frage wird der Fokus auf den Vergleich zwischen Low Level und High Level Fusionstechniken und deren Einfluss auf die Leistungsfähigkeit von Fusionsnetzwerken gelegt. Es wird untersucht, wie sich die unterschiedlichen Integrationsstufen von Daten aus verschiedenen Sensoren auf die Genauigkeit, Effizienz und Fehleranfälligkeit des Netzwerks auswirken. Dabei wird auch untersucht, ob eine der beiden Fusionsebenen zu einer höheren Robustheit gegenüber Störungen führt.

4 **Wie muss ein Netzwerk zur Integration gestörter Daten gestaltet werden?**
Diese Forschungsfrage widmet sich der optimalen Gestaltung der Netzwerkarchitektur, insbesondere hinsichtlich der Tiefe und Größe des Netzwerks, um gestörte Daten effizient zu integrieren. Es wird analysiert, wie tief und umfangreich ein Fusionsnetzwerk sein muss, um die Komplexität und die Herausforderungen von Störungen in den Trainingsdaten zu bewältigen.

Basierend auf der zuvor dargestellten Hypothese und den daraus abgeleiteten Forschungsfragen wird diese Arbeit mehrere innovative Ansätze zur Verbesserung der Robustheit von Sensorfusionssystemen vorstellen. Im Folgenden werden die zentralen Neuheiten dieser Arbeit aufgeführt:

1 Konzept zur Entwicklung eines Multisensorsystems mit Fokus auf Sensorstörungen wird erstellt.

2 Neue Methodiken zur effizienten Optimierung der Sensorplatzierung werden eingeführt.

3 Trainingskonzept KISA zur systematischen Verbesserung der Robustheit von Fusionsmethoden wird entwickelt.

4 Neuartiges Evaluationsmonzept zur Untersuchung der Robustheit von Sensorfusionsmethoden gegen Sensorstörungen werden neu eingeführt, darunter Sensitivitätsanalyse, Multifaktorielle Performanz-Evaluierung und die Modellierung der Robustheitsdynamik.

5 Low Level- und High Level-Fusionsstrategien für die Objektdetektion werden im Hinblick auf verschiedene Sensorstörungen verglichen.

6 Anwendung von Trainings- und Evaluationskonzept auf Methoden zur Fusion von realen Punktwolkendaten.

7 Vergleich von Auswirkungen verschiedener Störungen auf reale Punktwolkendaten hinsichtlich der Leistung von Fusionsmethoden.

8 Complex-YOLO wird erstmals für die Fusion von LiDAR- und RADAR-Daten erweitert und implementiert.

9 Der Einfluss der Tiefe der Netzwerkarchitektur auf die Robustheit gegenüber Sensorstörungen wird untersucht.

Um die gestellten Forschungsfragen zu beantworten, gliedert sich die Arbeit wie folgt: Zunächst wird in Kapitel 2 die neu entwickelten Konzepte und Methodiken vorgestellt. Kapitel 3 beschreibt die verwendeten Datensätze und die Störungen, welche auf den Daten angewendet werden. Das Kapitel 4 erläutert, die Implementation der neu entwickelten Methoden und wie die Netzwerke für die Sensorfusion erweitert werden. Anschließend werden in Kapitel 5 die

Ergebnisse der neu entwickelten Methoden zur optimalen Sensorplatzierung beschrieben. In Kapitel 6 werden die Ergebnisse der Anwendung von KISA auf die Fusion von LiDAR- und RADAR-Daten durch erweiterte Low Level und High Level Fusionsmethoden des Complex-YOLO-Algorithmus dargestellt. Die Robustheit wird in besonderem auf den Daten des Astyx Datensatzes untersucht. Weiterhin erfolgt in Kapitel 7 die Validierung des Trainingskonzepts KISA anhand weiterer Datensätze und Methoden, wie dem A2D2 (Audi autonomous driving)-Datensatz, an dem die spezielle Störung des kompletten Sensorausfalls getestet wird, und dem View-of-Delft-Datensatz, an dem die Störung durch Nebel getestet wird, sowie der heterogenen Sensordatenfusion von Kamera- und LiDAR-Daten durch den Objektdetektor Sparse Fuse Dense. Des weiteren wird KISA für die Steigerung der Robustheit von U-Net auf einem synthetisch erzeugten Datensatz und dem BraTS Datensatz getestet. Dieser Validierungsabschnit zeigt die Anwendbarkeit des entwickelten Konzepts unter variierenden Bedingungen und über verschiedene Sensortypen hinweg und demonstriert die sensor- und methodenagnostische Anwendbarkeit der vorgestellten Konzepte.

2 Konzept zur Verbesserung der Robustheit

2.1 Übersicht

Bei der Realisierung eines Projektes mit Multisensorsystemen, wie z.B. einem autonomen Fahrzeug, ist die Entwurfsphase des Multisensorsystems ein entscheidender Schritt. Bei einem Multisensorsystem müssen dabei folgende Teile beachtet werden:

- Sensoren - Welche Sensoren werden im System verwendet? Dies hat direkten Einfluss darauf, was das System wahrnehmen kann.
- Redundanz und Ausfallsicherheit - Welche Maßnahmen werden ergriffen, um den Ausfall eines Sensors zu kompensieren? Dies umfasst sowohl hardwareseitige Redundanzen als auch softwareseitige Mechanismen.
- Fusionsstrategie - Wie werden die Daten der verschiedenen Sensoren kombiniert, um ein vollständiges Bild der Umgebung zu erstellen?
- Verarbeitungsmethoden - Wie werden die generierten Daten verarbeitet? Dies umfasst die Auswahl von Objektdetektoren, Algorithmen zur Hindernisvermeidung, sowie die Implementierung von Methoden zur Echtzeitverarbeitung.

In dieser Arbeit wird ein neues Konzept für diese Entwurfsphase vorgestellt, welches genutzt werden kann, um alle genannten Teile eines Multisensorsystems zu optimieren. Ein Blockdiagramm dieses Konzepts ist in Abbildung 2.1 dargestellt.

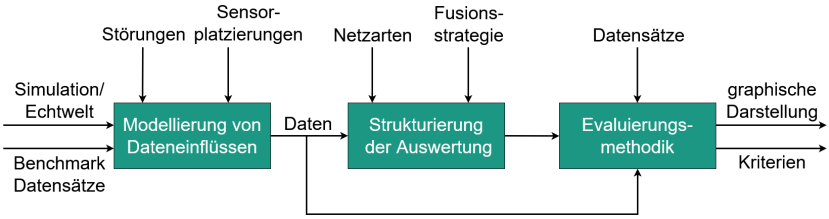


Abbildung 2.1: Neues Konzept für ein robustes Sensorsystem: Dieses Blockdiagramm zeigt die Schritte zur Konzeption eines Multisensorsystems. Es beginnt mit der *Modellierung von Dateneinflüssen*, bei der Daten, die aus einer Simulation oder *Benchmark Datensätzen* stammen, verändert werden. Die Form der Daten wird durch *Störungen* der Sensoren und die *Sensorplatzierung* beeinflusst. Für die *Strukturierung der Auswertung* der *Daten* werden *Netzarten* und *Fusionsstrategie* in Abhängigkeit von der Modalität der Daten gewählt. In der *Evaluierungsmethodik* werden die zuvor trainierten Netze auf den gleichen *Daten* oder auf neuen *Datensätzen* ausgewertet. Die Ergebnisse der Evaluierung werden in *graphischen Darstellungen* präsentiert und anhand spezifischer *Kriterien* bewertet.

Das Konzept dieser Entwurfsphase beginnt mit der *Modellierung von Dateneinflüssen* unter Verwendung von Daten, die entweder aus *Simulationen*, *Echtweltaufnahmen* oder *Benchmark-Datensätzen* stammen. Dabei können Einflüsse auf die Daten durch *Störungen* entweder direkt in einer Simulation oder in der realen Welt eingebracht werden, oder diese Modifikationen erfolgen durch nachträgliches Einbringen von *Störungen* in die Daten. Hinsichtlich der Störungen ist zu prüfen, welche im jeweiligen Anwendungsfall auftreten können und ob diese durch andere Sensoren kompensiert werden können.

In den *Benchmark-Datensätzen* ist die *Sensorplatzierung* bereits erfolgt und kann nachträglich nicht mehr verändert werden. Für neue Aufnahmen kann die *Sensorplatzierung* jedoch optimiert werden. Dazu werden zwei innovative, im Rahmen dieser Arbeit entwickelte Methoden zur Bestimmung einer optimalen Sensorplatzierung vorgestellt (*GAOS* und *DLOS*). Diese Methoden maximieren den Erfassungsbereich des Multisensorsystems und erhöhen die Ausfallsicherheit durch die erstmalige Berücksichtigung von Sensorredundanz.

Anschließend erfolgt die *Strukturierung der Auswertung* der Daten. Dabei wird die Art der Datenverarbeitung durch die Auswahl geeigneter *Netzarten* und *Fusionsstrategie* bestimmt. Diese Auswahl hängt stark von den vorliegenden

Datenmodalitäten und den Anforderungen der Anwendung ab. Ziel ist es, durch eine optimale Fusionsstrategie sowohl die Genauigkeit der Datenverarbeitung zu maximieren als auch die Robustheit des Gesamtsystems zu erhöhen. Um die Robustheit des Systems gegenüber den identifizierten Sensorstörungen zu optimieren, wird das in dieser Arbeit neu vorgestellte Trainingskonzept *Konzept zur Integration von Störungs-Adaptivität (KISA)* angewendet.

Die abschließende *Evaluierungsmethodik* dient der Bewertung der trainierten Netzwerke. Die Ergebnisse der Evaluierung werden durch graphische Darstellungen veranschaulicht und anhand definierter Kriterien quantitativ bewertet. Das Evaluationskonzept dieser Arbeit bietet eine neuartige Methode zur Bewertung der Robustheit von Sensorfusionsmethoden gegenüber Sensorstörungen. Es unterscheidet zwischen zwei neuen Evaluationsmethoden: der *Multifaktoriellen Performanz-Evaluation* und der *Sensitivitätsanalyse*. Die multifaktorielle Performanz-Evaluation analysiert die Leistungsfähigkeit des Systems über ein breites Spektrum von Störungen hinweg, während die Sensitivitätsanalyse dazu dient, die Grenzen und Schwachstellen des Systems aufzudecken. Durch gezielte Manipulation der Daten wird untersucht, wie das System auf verschiedene Arten von Störungen reagiert.

2.2 Modellierung von Dateneinflüssen

Dieser Abschnitt beschreibt, wie im Schritt *Modellierung von Dateneinflüssen* (siehe Abbildung 2.2) die Sensorplatzierung optimiert werden kann und mit welchen Metriken die Dateneinflüsse quantitativ verglichen werden können.

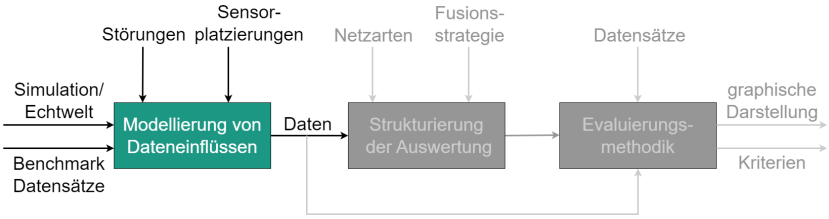


Abbildung 2.2: Erster Schritt des Konzepts für ein robustes Sensorsystem: Die *Modellierung von Dateneinflüssen*, bei der Daten, die aus einer *Simulation*, selbst aufgenommenen *Echtwelt* Daten oder *Benchmark Datensätzen* stammen, verändert werden. Die Form der Daten wird durch *Störungen* der Sensoren und die *Sensorplatzierung* beeinflusst.

2.2.1 Optimierung der Sensorplatzierung

Insbesondere bei Fahrzeugen stellt die Sensorplatzierung aufgrund ihrer komplexen Geometrie eine Herausforderung dar. Eine optimale Platzierung der Sensoren kann das Wahrnehmungsfeld erweitern und gleichzeitig die Redundanz zwischen den Sensoren optimieren. Diese Redundanz ist entscheidend für die Ausfallsicherheit, da bei einem Sensorausfall andere Sensoren weiterhin Daten liefern und so die Sicherheit des Fahrzeugs gewährleisten können.

Die erste neu entwickelte Methode *Genetische Algorithmus-Optimierung für Sensorplatzierung* (GAOS) verwendet einen genetischen Algorithmus zur Optimierung und wurde 2021 im IEEE Sensors Journal veröffentlicht [Bere21b]. Die zugehörige Implementierung ist ebenfalls öffentlich verfügbar[1]. Die zweite Methode *Deep Learning-Optimierung für Sensorplatzierung* (DLOS) nutzt innovativ Deep Deterministic Policy Gradient (DDPG) zur Optimierung und wurde 2023 im SAE International Journal of Connected and Automated Vehicles veröffentlicht [Bere24a], mit der entsprechenden öffentlicher Implementierung[2]. DLOS ist die erste Methode zur Sensorplatzierung, welche es ermöglicht nur anhand der geometrischen Form des Modells eine Platzierung zu generieren. Beide Methoden erweitern die von Kim und Park [Kim19] vorgeschlagene

[1] https://github.com/BerensRWU/GeneticOpt
[2] https://github.com/jdrew1/SensorOpt

Methode, indem sie eine exakte Platzierung der Sensoren entlang der Fahrzeugform ermöglichen, eine parallele Optimierung der Platzierung mehrerer Sensoren durchführen und potenzielle Sichtfeldverdeckungen berücksichtigen.

In einer Simulationsumgebung wird die Optimierungsumgebung durch einen Zylinder modelliert, in dessen Mitte das Fahrzeug steht, auf dem die L viele Sensoren angeordnet werden. Ein Beispiel, wie eine Optimierungsumgebung gestaltet werden kann, ist in Abbildung 2.3 dargestellt. Dieser Zylinder ist eine Erweiterung der LiDAR-Belegungskarte von Kim und Park und wird in ein LiDAR-Belegungsraster unterteilt, das aus $N_a \cdot N_h$ Teilflächen besteht. $n(i,j)$ beschreibt, wie viele verschiedene Sensoren die Teilfläche (i,j) detektieren. Ziel der Optimierung ist es, mit den vorhandenen Sensoren möglichst viele Teilflächen abzudecken. Dies wird durch die folgende Zielfunktion beschrieben, die die totale LiDAR-Belegung darstellt:

$$\%LO_k = \frac{\sum_{i=1}^{N_a} \sum_{j=1}^{N_h} lob_k(i,j)}{N_a \cdot N_h},$$

wobei lob_k eine Gewinnfunktion für jede Teilfläche (i,j) ist. Drei Gewinnfunktionen mit unterschiedlichen Eigenschaften werden vorgestellt:

$$lob_1^p(i,j) := \begin{cases} 1 & , n(i,j) \geq p \\ 0 & , n(i,j) < p \end{cases}.$$

Diese Funktion beschreibt, ob eine Teilfläche von mindestens p verschiedenen Sensoren erfasst wurde. Die Gewinnfunktion $lob_1^p(i,j)$ wurde bereits von Kim und Park definiert. Da diese Funktion jedoch keine Redundanz berücksichtigt, werden zwei alternative Gewinnfunktionen vorgeschlagen:

$$lob_2(i,j) := \frac{1}{2}(lob_1^1(i,j) + lob_1^2(i,j)),$$

$$lob_3(i,j) := \frac{2 - \frac{1}{2}^{n(i,j)-1}}{2 - \frac{1}{2}^{L-1}}.$$

lob_2 belohnt jede erkannte Teilfläche und vergibt einen zusätzlichen Bonus für jede mehrfach erkannte Teilfläche. Der Faktor $1/2$ stellt sicher, dass die maximale Belohnung für eine Teilfläche 1 beträgt. Bei lob_3 wird der zusätzliche Gewinn durch die Mehrfachbelegung einer Teilfläche mit mehreren Sensoren reduziert. lob_2 und lob_3 ermöglichen somit die Erfassung von Redundanz und eine optimale Abdeckung des Zylinders.

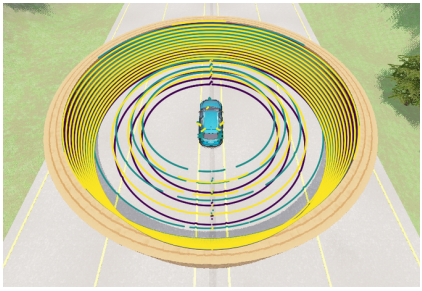


Abbildung 2.3: Beispiel aus der Simulationsumgebung CARLA, wie eine Optimierungsumgebung mit Zylinder und Fahrzeug gestaltet wird. Farblich dargestellt sind die LiDAR Punkte der drei simulierten LiDAR Sensoren.

Um die Sensoren exakt entlang der geometrischen Form des Fahrzeugs zu platzieren, wird diese, erstmals für die Optimierung der Sensorplatzierung, durch einen Graphen beschrieben. Die Knoten repräsentieren Positionen, an denen ein Sensor platziert werden kann, und zwei Knoten sind durch eine Kante verbunden, wenn der Abstand zwischen ihnen klein genug ist. Die Beschreibung des Fahrzeugs durch einen Graphen bietet den Vorteil, dass die kürzeste Distanz zwischen zwei Punkten durch Berechnung des kürzesten Pfads zwischen den entsprechenden Knotenpunkten ermittelt werden kann, wobei dieser Weg vollständig auf der Oberfläche des Fahrzeugs liegt. Zusätzlich kann durch das Löschen bestimmter Knoten verhindert werden, dass an diesen Stellen ein Sensor platziert wird. Dadurch können Designanforderungen oder bauliche Gegebenheiten berücksichtigt werden.

Da in der verwendeten Simulationsumgebung CARLA ein Sensor als infinitesimal klein modelliert wird, ein realer Sensor jedoch eine physikalische Größe besitzt, muss auch die Verdeckung des Sichtfelds eines Sensors durch einen anderen Sensor berücksichtigt werden. Dazu wird die Punktwolke eines Sensors sowie die Oberfläche des anderen Sensors vom kartesischen in das sphärische Koordinatensystem transformiert. Punkte aus der Punktwolke eines Sensors, die einen Polar- oder Azimutwinkel im Intervall der Polar- bzw. Azimutwinkel der Oberfläche eines anderen Sensors aufweisen, werden von diesem Sensor verdeckt. Indem diese Punkte aus der Punktwolke gelöscht werden, wird die Verdeckung simuliert.

Die GAOS Methode optimiert die Sensorplatzierung durch Methoden der genetischen Algorithmen. Es werden Mutation und Reproduktion verwendet, um iterativ die Sensorplatzierung zu optimieren.

Bei der Mutation wird ein Sensor aus der aktuellen Anordnung zufällig ausgewählt und seine Position auf dem Graphen verändert, indem er zufällig einen Pfad zurücklegt. Die Länge des Pfades ist eine ganzzahlige gleichverteilte Zufallszahl $X \sim \mathcal{U}_{\{0,1,\ldots,\alpha\}}$. Zusätzlich wird die Rotation des Sensors angepasst, indem zu jeder Rotationskomponente eine normalverteilte Zufallsvariable $X \sim \mathcal{N}_{0,\beta}$ addiert wird.

Für die Reproduktion werden zwei Sensoren aus unterschiedlichen Sensorplatzierungen zufällig ausgewählt. Die Positionen dieser Sensoren werden kombiniert, indem der kürzeste Pfad zwischen ihnen berechnet und der $\lfloor l/2 \rfloor$-te Knoten auf diesem Pfad gewählt wird, wobei l die Länge des Pfades ist. Die Rotationen der beiden Sensoren werden durch den Mittelwert der jeweiligen Rotationsgrößen kombiniert.

Insgesamt besteht die GAOS Methode aus den folgenden Schritten:

Schritt 1: Initialisiere eine Population der Größe P an Individuen, wobei jedes Individuum eine spezifische Anordnung von M Sensoren darstellt.

Schritt 2: Sortiere die aktuelle Population nach der totalen LiDAR-Belegung $\%LO_k$. Behalte nur die besten P-Sensorplatzierungen.

Schritt 3: Erzeuge neue Nachkommen durch Mutation und Reproduktion.

Schritt 4: Gehe zurück zu Schritt 2.

Da die Optimierung mit dem genetischen Algorithmus zeitaufwendig ist und für jedes Fahrzeugmodell separat durchgeführt werden muss, wurde im Rahmen dieser Arbeit DLOS entwickelt, die den Deep Deterministic Policy Gradient (DDPG) zur Optimierung von $\%LO_k$ anstelle des genetischen Algorithmus verwendet.

Der Deep Deterministic Policy Gradient (DDPG) ist eine Off-Policy-Erweiterung des Q-Learnings. DDPG kombiniert Elemente des Q-Learnings und der Policy-Gradient-Methoden. Beim Q-Learning lernt ein Agent eine Strategie, indem er die optimale Aktionswertfunktion (Q-Funktion) schätzt, welche die erwarteten kumulativen Belohnungen für ein gegebenes Zustands-Aktions-Paar repräsentiert [Watk92, Clif20]. Off-Policy-Algorithmen wie DDPG [Silv14] speichern Erfahrungen in einem separaten Puffer und nutzen diese zur Aktualisierung der Policy. Dies ermöglicht ein stabileres und effizienteres Lernen, da die Korrelation zwischen aufeinanderfolgenden Erfahrungen reduziert wird.

Das Ziel des DDPG-Algorithmus ist die Optimierung der Platzierung der LiDAR-Sensoren. Der Agent wählt dabei die optimale Platzierung der LiDAR-Sensoren auf einem Fahrzeug basierend auf dem aktuellen Zustand des Fahrzeugs aus. Dieser Zustand umfasst Informationen über die Form des Fahrzeugs, die durch einen Graphen beschrieben wird. Das Optimierungsziel des Netzwerks, und damit die Belohnung, ist die totale LiDAR-Belegung $\%LO_k$. Dabei wurde die Methode auf eine Vielzahl von Fahrzeugformen angewendet, um die Platzierung der LiDAR-Sensoren für jedes dieser Fahrzeuge zu optimieren.

2.2.2 Quantitative Beurteilung und Gruppierung von Störung

Um den Einfluss von Störungen auf die geometrische Form, Struktur und Dichte von Punktwolken quantitativ zu bewerten und Störungen nach geometrischer Ähnlichkeit zu gruppieren, werden Metriken benötigt. Im Rahmen dieser Arbeit wurden Metriken zur Bestimmung der Ähnlichkeit von Punktwolken untersucht. Die Ergebnisse wurden in der Zeitschrift at - Automatisierungstechnik 2021 veröffentlicht [Bere21a]. Basierend auf diesen Untersuchungen wurden die Metriken Chamfer Distanz (CD) und Average Ratio (AR) ausgewählt, die effizient berechenbare und gute Ergebnisse beim Vergleich zweier Punktwolken bezüglich Fehlern liefern.

Beide Metriken berechnen den Abstand $d(.,.)$ zwischen zwei Punkten, wobei $d(.,.)$ standardmäßig den euklidischen Abstand darstellt, sofern nicht anders angegeben. Zusätzliche Informationen jedes Punktes, wie beispielsweise die Intensität bei LiDAR-Punkten, werden in der Bewertung mit diesen Metriken nicht berücksichtigt.

Die Chamfer Distanz (CD) quantifiziert den mittleren minimalen Abstand aller Punkte einer Punktwolke Φ^1 zu einem beliebigen Punkt in einer zweiten Punktwolke Φ^2 und umgekehrt.

Definition 2.1. *Die Chamfer Distanz zwischen zwei Punktwolken Φ^1 und Φ^2 ist definiert wie folgt:*

$$\mathbf{CD}(\Phi^1,\Phi^2) := \frac{\sum_{y\in\Phi^2} \min_{x\in\Phi^1} d(x,y)}{|\Phi^2|} + \frac{\sum_{x\in\Phi^1} \min_{y\in\Phi^2} d(x,y)}{|\Phi^1|}.$$

Die erstmals in [Bere21a] veröffentlichte Average Ratio (AR) erweitert das Konzept des Ratio-Maßes, indem sie mehrere Schwellenwerte berücksichtigt und beide Punktwolken gleichermaßen einbezieht. Damit das Maß AR negativ mit der Störung korreliert, wird der Wert durch Subtraktion von 1 transformiert.

Definition 2.2. *Die AR zwischen zwei Punktwolken* Φ^1 *und* Φ^2 *ist wie folgt definiert:*

$$\mathbf{AR}(\Phi^1,\Phi^2) := 1 - \frac{\sum_{i=1}^{N} i \cdot \frac{|S_{D_i,\Phi^1,\Phi^2}|}{|\Phi^1|} + \sum_{i=1}^{N} i \cdot \frac{|S_{D_i,\Phi^2,\Phi^1}|}{|\Phi^2|}}{N^2 + N},$$

wobei S_{D_i,Φ^1,Φ^2} *die Menge aller Punkte von* Φ^1 *ist, welche eine Distanz zu einem Punkt in* Φ^2 *haben die kleiner als* D_i *ist.* N *ist die Anzahl der Schwellenwerte* D_i*, die berücksichtigt werden, wobei* $D_1 < \cdots < D_N$*.*

Das Maß CD ist bei zwei ähnlichen Punktwolken gering und nehmen zu, wenn die Punktwolken ungleicher werden; sie sind also negativ mit der Ähnlichkeit zwischen zwei Punktwolken korreliert.

Die Charakteristiken dieser zwei Metriken wurden detailliert in der Arbeit [Bere21a] untersucht.

Zur Visualisierung der Gruppierungen der Störungen wird ein Dendrogramm eingesetzt. Ein Dendrogramm ist eine graphische Darstellung, die hierarchische Beziehungen und Ähnlichkeiten zwischen verschiedenen Datengruppen veranschaulicht. Zur Bildung der Gruppen wird die Ward-Linkage-Methode verwendet. Durch die Analyse des Dendrogramms lassen sich klar abgegrenzte Hauptgruppen der untersuchten Störungen identifizieren, was eine tiefere Einordnung der Störungscharakteristika ermöglicht.

2.3 Strukturierung der Auswertung

Bei Multisensorsystemen ist die *Strukturierung der Auswertung* (siehe Abbildung 2.4) mit verschiedenen Dateneinflüssen entscheidend, da die Qualität und Verlässlichkeit der fusionierten Daten stark von der Art der Datenverarbeitung abhängt. Unterschiedliche Sensoren liefern Daten mit variierenden Modalitäten, Genauigkeiten und Störungen, die durch geeignete *Fusionsstrategien* und *Netzarten* verarbeitet werden müssen. Eine sorgfältige Strukturierung der

Auswertung ermöglicht es, diese Einflüsse zu berücksichtigen und die Sensorinformationen so zu kombinieren, sodass die Gesamtgenauigkeit maximiert und die Auswirkungen von fehlerhaften oder gestörten Sensordaten minimiert werden. Deswegen wird, das im Rahmen dieser Arbeit entwickelte, *Konzept zur Integration von Störungs-Adaptivität (KISA)* vorgestellt.

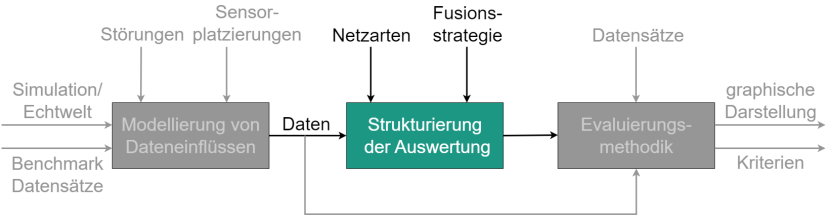


Abbildung 2.4: Für die *Strukturierung der Auswertung* der *Daten* werden *Netzarten* und *Fusionsstrategie* in Abhängigkeit von der Modalität der Daten gewählt.

Das hier vorgestellte KISA ist ein Verfahren zur systematischen Erhöhung der Robustheit von Sensorfusionsmethoden. Im Gegensatz zu herkömmlichen Ansätzen, bei denen oft willkürlich entschieden wird, wie viele und welche gestörten Daten während des Trainings verwendet werden, werden bei diesem Konzept gezielt unterschiedliche Anteile von Störungen in den Trainingsprozess integriert. Der Ablauf des Trainingskonzept ist im Schaubild 2.5 dargestellt, das den Auswahlprozess zur Integration von Sensorstörungen im Training veranschaulicht.

Da das Konzept zur Steigerung der Robustheit aus zwei Teilen besteht, werden aus der Menge aller Dateninstanzen zufällig zwei disjunkte Mengen gebildet. Damit liegen für die Evaluation Daten vor, welche im Training noch nicht verwendet wurden:

Definition 2.3. *Der Trainingsdatensatz sei **TD** und der Evaluationsdatensatz sei **ED**. Dabei wird mittels Index angemerkt, von welchem Sensor eine bestimmte Dateninstanz kommt und welchem Frame die Instanz zugeordnet wird.*

So ist zum Beispiel $\mathbf{TD}_{\text{LiDAR},10}$ die 10. Punktwolke der LiDAR-Sensordaten. Im Folgenden wird davon ausgegangen, dass die Daten, bis auf das gewöhnliche Datenrauschen, ohne weitere Störungen vorliegen.

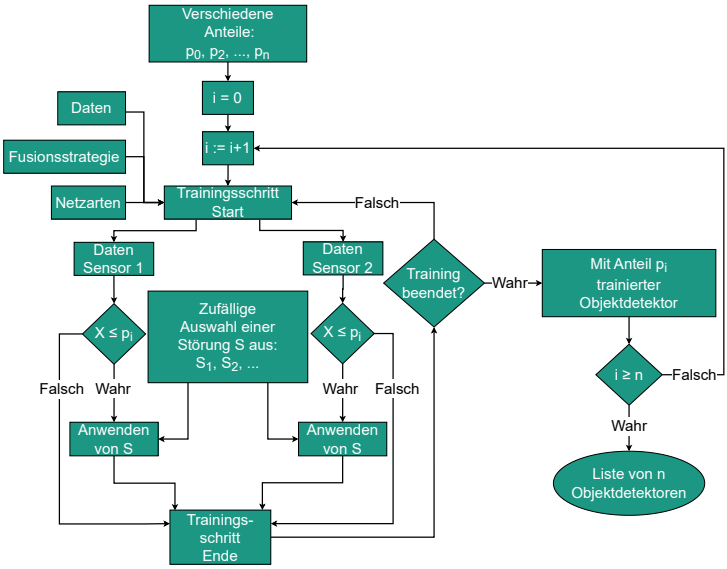


Abbildung 2.5: Das vorliegende Schaubild präsentiert das in dieser Arbeit dargestellte Trainingskonzept KISA für eine Sensorfusion mit zwei Modalitäten, es kann aber auf beliebig viele Sensoren erweitert werden. Beim vorgestellten Konzept werden in den Sensordaten Störungen integriert, wenn eine gleichverteilte Zufallszahl $X \leq p_i$ ist.

Definition 2.4. *Die Menge aller Störungen, die auf den Daten angewendet wird, ist definiert als* $\mathbf{S} = \{S_1, \ldots, S_{n_S}\}$. *Dabei ist* n_S *die Anzahl aller betrachteten Störungen.*

Definition 2.5. *Wird auf allen Evaluationsinstanzen aus den Evaluationsdaten* ***ED*** *eine spezielle Störung* $S_k \in \mathbf{S}$ *angewendet, so wird diese Datenmenge als* $\boldsymbol{ED}^{S_k}$ *notiert.*

Um eine Fusionsmethode O auf eine Menge an Störungen $\mathbf{S}$ vorzubereiten, werden Daten mit Störungen bereits in den Trainingsprozess integriert. Hierbei werden mögliche Prozentsätze $p_i \in \{0\%, \dots, 100\%\}$ für $i = 1, \dots, n_p$ festgelegt. Bei einem Anteil von p_i der Trainingsdaten $\mathbf{TD}$ werden Störungen aus $\mathbf{S}$ in den Trainingsinstanzen eines oder mehrerer Sensoren hinzugefügt, während zum restlichen Anteil $1 - p_i$ der Trainingsinstanzen keine Störungen hinzugefügt werden. Um dies zu erreichen, wird für jede Trainingsinstanz $\mathbf{TD}_{\text{Sensor},j}$ eine gleichverteilte Zufallszahl $X \sim \mathcal{U}_{0\%,100\%}$ gezogen:

- Im Fall $X \leq p_i$ wird die Trainingsinstanz $\mathbf{TD}_{\text{Sensor},j}$ gestört. Befindet sich in $\mathbf{S}$ nur eine Störung, so wird diese angewendet. Befinden sich in $\mathbf{S}$ mehrere Störungen, so wird bei jeder Trainingsinstanz zufällig eine Störung $\tilde{S} \in \mathbf{S}$ gewählt. Dabei ist die Wahrscheinlichkeit für jede Störung in $\mathbf{S}$ gleich.
- Im Fall $X > p_i$ wird die Trainingsinstanz $\mathbf{TD}_{\text{Sensor},j}$ nicht gestört.

Die auf diese Weise trainierte Fusionsmethode wird als $O_{p_i,\mathbf{S}}$ bezeichnet. Dies gibt an, in welchen Anteil p_i der Trainingsdaten eine Störung aus $\mathbf{S}$ integriert wurde.

Diese Strategie erweitert die Vielfalt der Trainingsdaten[1,2]. Das vorgestellte Konzept kann auch als Data Augmentation interpretiert werden. Bei Data Augmentation wird jedoch nicht explizit auf die Herausforderung der Robustheit gegenüber Sensorstörungen eingegangen, und es wird nur Rauschen geringen Ausmaßes hinzugefügt, das den gesamten visuellen Inhalt nicht

[1] Wenn nur eine Störung auf die Daten eines Sensors angewendet wird, führt dies zu einer Verdopplung der Datenmenge, zu einer Verdreifachung, wenn die Daten von zwei Sensoren getrennt verändert werden, oder zu einer Erhöhung um den Faktor $2^n - 1$ für Setups mit n veränderbaren Sensoren.

[2] Theoretisch ist es möglich, dass alle oder keine Sensordaten gestört werden, aber dies ist sehr unwahrscheinlich. Zum Beispiel ist bei $p_i = 10\%$ und 500 Trainingsdaten die Wahrscheinlichkeit, dass 25 oder weniger Daten nicht geändert werden oder 75 oder mehr Daten geändert werden, insgesamt 0,03%. Die Wahrscheinlichkeit, dass 45 bis 55 Daten geändert werden ist 59% und die Wahrscheinlichkeit für 40 bis 60 geänderten Daten beträgt 89%.

wesentlich verändert. In dieser Arbeit werden hingegen Störungen hinzugefügt, die das Erscheinungsbild der Daten erheblich beeinträchtigen, um die Robustheit zu erhöhen.

2.4 Evaluierungsmethodik

Es gibt keine einfache Möglichkeit, Störungseinflüsse zu quantifizieren und zu visualisieren. Jedoch ist der Vergleich von Fusionsmethoden hinsichtlich ihrer Robustheit gegenüber Störung wichtig, weswegen objektive *Evaluierungsmethodik* (siehe 2.6) benötigt werden. Dafür werden in dieser Arbeit zwei neue Evaluationsmöglichkeiten vorgestellt: Multifaktorielle Performanz-Evaluation und Sensitivitätsanalyse.

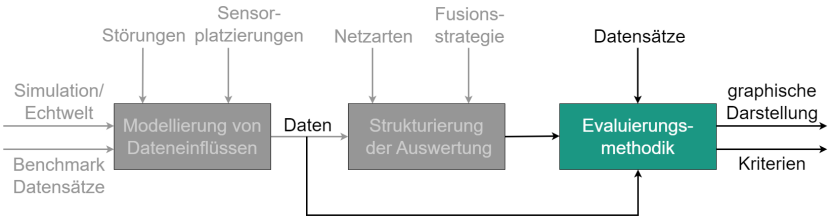


Abbildung 2.6: In der *Evaluierungsmethodik* Evaluierungsmethodik werden die zuvor trainierten Netze auf den gleichen *Daten* oder auf neuen *Datensätzen* ausgewertet. Die Ergebnisse der Evaluierung werden in *graphischen Darstellungen* präsentiert und anhand spezifischer *Kriterien* bewertet.

Zur Bewertung der Robustheit der trainierten Fusionsmethode $O_{p_i,\mathbf{S}}$ gegenüber den Störungen $\{S_1, \ldots, S_{n_S}\}$, wird für jede Störung $S_k \in \{S_1, \ldots, S_{n_S}\}$ ein Evaluationsdatensatz $\mathbf{ED}^{S_k}$ erzeugt, so dass jede Evaluationsinstanz durch S_k gestört ist. Die Performanz der Fusionsmethode $O_{p_i,\mathbf{S}}$ auf den gestörten Daten $\mathbf{ED}^{S_k}$ und ungestörten Daten $\mathbf{ED}$ wird von einer Evaluationsmetrik f bewertet.

Definition 2.6. *Sei $O_{p_i,\mathbf{S}}$ eine Fusionsmethode, die auf einen Prozentsatz p_i der Trainingsdaten **TD** unter Einbeziehung von Störungen $\mathbf{S}$ trainiert wurde. Die*

Evaluationsfunktion $\mathbf{F}_{p_i,\mathbf{S}}$ *ist definiert als:*

$$\mathbf{F}_{p_i,\mathbf{S}} = \left\{ f(O_{p_i,\mathbf{S}}, \boldsymbol{ED}^{S_k}); \forall S_k \in \{S_1, \ldots, S_{n_S}\} \right\} \cup \left\{ f(O_{p_i,\mathbf{S}}, \boldsymbol{ED}) \right\},$$

wobei $f(O_{p_i,\mathbf{S}}, \boldsymbol{ED})$ *das Ergebnis der Evaluationsmetrik* f *auf den ungestörten Daten* $\boldsymbol{ED}$ *darstellt, und* $f(O_{p_i,\mathbf{S}}, \boldsymbol{ED}^{S_k})$ *das Ergebnis auf den Daten* $\boldsymbol{ED}^{S_k}$ *bezeichnet, welche durch die Störung* S_k *modifiziert wurden.*

Anhand dieser Daten wird die Robustheit der trainierten Fusionsmethoden durch das folgende Konzept bewertet:

- *Multifaktorielle Performanz-Evaluation*: konzentriert sich darauf, die Leistungsfähigkeit des Systems über ein Spektrum verschiedener Störungsarten hinweg zu analysieren und zu bewerten.
- *Sensitivitätsanalyse*: setzt gezielt manipulierte Daten ein, um die Grenzen und Schwachstellen des Systems herauszufinden. Dies wird verwendet, um die Robustheit der Fusionsmethode gegenüber bestimmten Störungen zu bewerten.

2.4.1 Multifaktoriellen Performanz-Evaluation

Das Hauptanwendungsgebiet der Multifaktoriellen Performanz-Evaluation ist die Identifizierung des Anteils p_i, bei dem die Fusionsmethode die besten Leistungen sowohl bei gestörten als auch bei ungestörten Daten aufweist.

Definition 2.7. *Bei der Multifaktoriellen Performanz-Evaluation wird für jeden Anteil* p_i *ein gewichtetes Mittel der Evaluationsmetriken über alle bewerteten Störungen* S_k *berechnet. Diese multifaktoriellen Performanz* $M_{p_i}(\hat{\mathbf{S}})$ *ist definiert als:*

$$M_{p_i}(\hat{\mathbf{S}}) = \sum_{S_K \in \hat{\mathbf{S}}} g_{S_k} \cdot f(O_{p_i,\mathbf{S}}, \boldsymbol{ED}^{S_k}),$$

wobei $\hat{\mathbf{S}}$ *die Menge der Störungen repräsentiert, über die gemittelt wird, und* g_{S_k} *die Gewichtung der jeweiligen Störung* S_k *darstellt.*

Die Gewichtungen g_{S_k} werden auf Grundlage der Auftretenswahrscheinlichkeit jeder Störung und ihrer Bedeutung für die Aufrechterhaltung robuster Detektionsleistungen festgelegt.

Die Ergebnisse dieser Evaluierung für sämtliche Prozentsätze p_i lassen sich gemeinsam graphisch darstellen. Wie in der Beispielgraphik der Abbildung 2.7 stellen die blaue und rote Linien die Evaluationsergebnisse $f(O_{p_i,\mathbf{S}}, \mathbf{ED}_{\text{Störung mit Grad 0}})$ beziehungsweise $f(O_{p_i,\mathbf{S}}, \mathbf{ED}_{\text{Störung mit Grad 5}})$ für die verschiedenen p_i dar. Die graue Linie stellt den multifaktoriellen Performanz Wert $M_{p_i}(\hat{\mathbf{S}})$ dar, wobei hier $\hat{\mathbf{S}}$ die Störung mit den verschiedenen Graden für die verschiedenen p_i zusammenfasst. Eine solche Visualisierung erleichtert es, das p_i zu identifizieren, das die Robustheit des Systems am stärksten verbessert. Wird etwa jede Störung als gleich bedeutsam angesehen, so zeigt die Abbildung 2.7, dass ein Anteil von $p_8 = 80\%$ optimal ist, da $M_{p_8}(\hat{\mathbf{S}})$ den höchsten Wert für das gewichtete Mittel in der Leistungsbewertung erzielt.

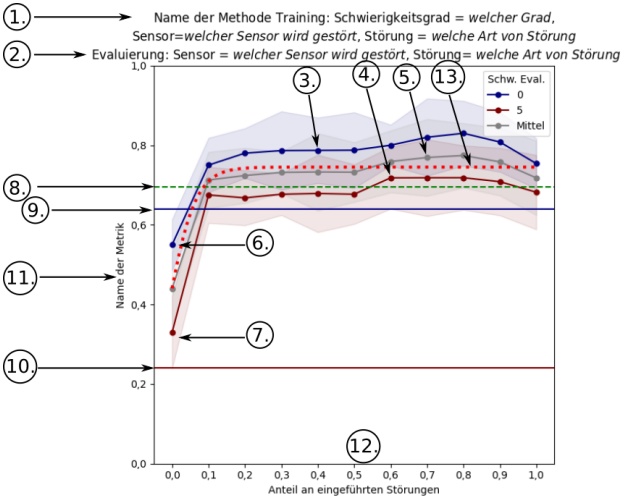


Abbildung 2.7: Beispiel für die im Rahmen der Arbeit neu entwickelte graphische Darstellung der *Multifaktoriellen Performanz-Evaluation*. Für die Erklärung der einzelnen Bestandteile (1.)-(13.) sei auf den Text verwiesen.

(1.) und (2.) geben die für das Training $\mathbf{S}$ und die Auswertung $\hat{\mathbf{S}}$ verwendeten Sensorstörungen an. Beispielsweise kann (1.) anzeigen, dass der Detektor auf alle Störungen und allen Graden trainiert wurde und (2.) könnte beschreiben, dass während der Evaluation Sensor 1 gestört wurde und Sensor 2 nicht gestört wurde.
(3.) und (4.) (Linie) zeigen die durchschnittlichen Auswertungsergebnisse bei Störungsgrad 0 (dunkelblau) beziehungsweise Störungsgrad 5 (dunkelrot), wobei der schattierte Bereich $\pm$ 1 Standardabweichung darstellt. Beachte, dass das gleiche Netz auf Stufe 0 wie auf Stufe 5 bewertet wurde. Der schattierte Bereich ist nur vorhanden, wenn der Test mehrfach durchgeführt wurde.
(5.) (graue Linie) zeigt die Mittelwerte $\boldsymbol{M}_{p_i}(\hat{\mathbf{S}})$, wobei hier $\hat{\mathbf{S}}$ die Störung mit den verschiedenen Graden zusammenfasst, für die verschiedenen p_i.
(6.) und (7.) sind Vergleichspunkte für die Methode: Die Methode wurde mit Sensorfusion und $p = 0\%$ trainiert, daher wurden beim Training keine Daten verändert. Das Netz wurde bei den entsprechenden Störungsgrad bewertet ((6.) auf Grad 0 und (7.) auf Grad 5).
(8.) (gestrichelte grüne horizontale Linie) ist ein weiterer Vergleichswert für die Methode: Die Methode wurde ohne Fusion nur auf den Daten des Sensors trainiert, welcher bei der Evaluation nicht gestört ist und auch auf diesen ausgewertet. Der Vergleich mit diesem Wert verdeutlicht, ob die Fusionsmethode mindestens eine genauso gute Leistung hat, wie eine vergleichbare Methode ohne Fusion auf ungestörten Daten.
(9.) und (10.) (horizontale Linie) sind die letzten zwei Vergleichswerte für die Methode: Die Methode wurde ohne Fusion nur auf Daten des gestörten Sensors mit den gleichen Störungen und Grad trainiert, wie es ausgewertet wurde. Für den Wert der dunkelroten Linie (10.) wurde die Methode ausschließlich auf gestörten Daten der Stufe 5 trainiert und auf gestörten Daten der Stufe 5 ausgewertet. Entsprechend wurde das Netz für die dunkelblaue Linie (9.) auf Grad 0 Daten trainiert und ausgewertet. Diese Werte verdeutlichen, wie die Fusion der gestörten Daten mit ungestörten Daten die Methode robuster macht.
(11.) Auf der y-Achse ist die Evaluationsmetrik f aufgetragen.
(12.) Die x-Achse stellt den Parameter p dar, der angibt, welcher Anteil der Trainingsdaten gestört ist.

⑬ (rote gepunktete Linie) Doppelt logistische Funktion, angepasst an die Werte $(p_i, \boldsymbol{M}_{p_i}(\hat{\mathbf{S}}))$. Da mit dieser Linie die Sättigung an Robustheit bestimmt wird, ist diese nur vorhanden, wenn die Störungsart, welche im Training verwendet wird, die selbe ist, wie im Evaluationsprozess. Anhand der Asymptote der logistischen Funktion wird die Sättigung der Robustheit bestimmt. Für weitere Informationen siehe Abschnitt 2.4.3.

2.4.2 Sensitivitätsanalyse

Im Zentrum der Sensitivitätsanalyse steht die Aufdeckung von Grenzen und Schwachstellen des Detektors. Durch die detaillierte Betrachtung, wie der Detektor auf unterschiedliche Arten und Grad von Störungen reagiert, werden Störungen identifiziert, für die die Methode nicht robust ist. Diese Erkenntnisse sind entscheidend, um die Bereiche zu verstehen, die einer Verbesserung bedürfen. Dafür werden die Ergebnisse graphisch dargestellt, wie in Abbildung 2.8.

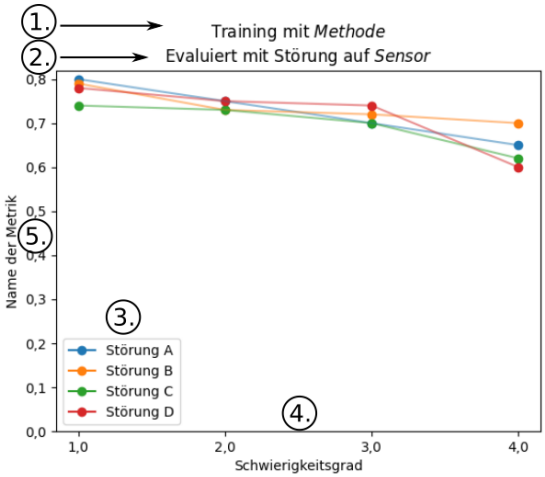


Abbildung 2.8: Beispiel für die im Rahmen der Arbeit neu entwickelte graphische Darstellung der *Sensitivitätsanalyse*. Für die Erklärung der einzelnen Bestandteile ①-⑤ sei auf den Text verwiesen.

① Gibt an, mit welcher *Methode*, also Netzart und Fusionstratigie, das Training durchgeführt wurde.
② Gibt an, auf welchem *Sensor* das Netzwerk evaluiert wurde.
③ Zeigt an, bei welchen Störungsarten das Netzwerk evaluiert wurde (hier: Störung A, Störung B, Störung C und Störung D).
④ Zeigt an, auf welchen Schwierigkeitsgrad das Netzwerk evaluiert wurde (hier: Stufe 1, 2, 3 und 4).
⑤ Gibt an, mit welchem Maß evaluiert wurde und welche Metrikwert das Netzwerk hat.

2.4.3 Neues Konzept zur Analyse der Sättigungsdynamik der Robustheit

Die Sättigungsdynamik beschreibt das Verhalten eines Systems, insbesondere dessen Robustheit, wenn es auf zunehmende Störungsanteile trainiert wird. Diese Dynamik ist von besonderem Interesse, da sie Aufschluss darüber gibt, ab welchem Punkt zusätzliche Störungen im Training keinen signifikanten Zugewinn an Robustheit mehr bieten, und sogar kontraproduktiv wirken können. Die Untersuchung dieser Dynamik stellt einen wichtigen Beitrag zur Optimierung von Trainingsstrategien in der robusten Objektdetektion dar.

Im Rahmen der Multifaktoriellen Performanz-Evaluation wird für verschiedene Anteile p_i der Trainingsdaten, denen Störungen $\hat{S}$ hinzugefügt werden, ein multifaktorieller Performanz Wert $M_{p_i}(\tilde{S})$ berechnet. Dabei bezieht sich $\tilde{S}$ auf eine Menge von Störungen, die für die Evaluation verwendet wird. Da die folgende Methode die Sättigungsdynamik der Robustheit untersucht wird, ist $\hat{S} \subseteq \tilde{S}$.

Empirische Beobachtungen, zeigen, dass $M_{p_i}(\tilde{S})$ mit zunehmendem Anteil p_i anfangs ansteigt, bis eine Sättigung erreicht wird und das System maximale Robustheit aufweist. Oftmals fällt $M_{p_i}(\tilde{S})$ für hohe Werte von p_i wieder ab, wenn $\hat{S} \neq \tilde{S}$.

Um den Anteil $\hat{p}$ zu bestimmen, bei dem diese Sättigung eintritt, werden die Daten $(p_i, M_{p_i}(\tilde{S}))$ durch eine doppelt logistische Funktion approximiert:

Definition 2.8. *Eine doppelt logistische Funktion ist definiert durch:*

$$logit(p) = \frac{L}{1 + e^{-k_1 \cdot (p - x_1)}} - \frac{L}{1 + e^{-k_2 \cdot (p - x_2)}},$$

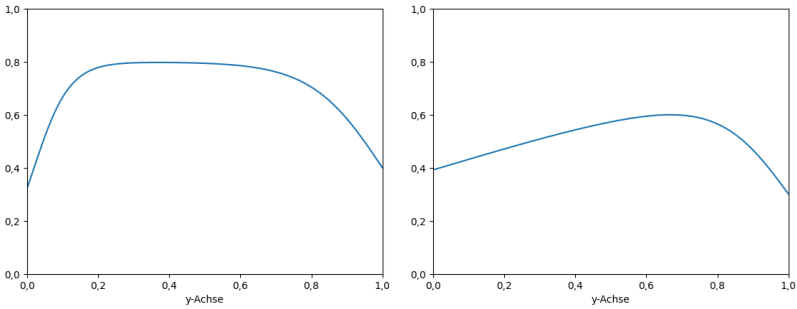
wobei $L \in \mathbb{R}^{\geq 0}$ die obere Schranke der Funktion, $k_{1,2} \in \mathbb{R}^{\geq 0}$ die Steilheit der Kurve beschreiben und $x_{1,2} \in \mathbb{R}$ die Wendepunkte der S-Kurve sind.

In Abbildung 2.9 sind zwei Beispiele für die doppelt logistische Funktion dargestellt.

Die Sättigung beschreibt den Anteil, ab welchem zusätzliche Integration von Störung im Training keinen Zugewinn mehr bietet. Zur Bestimmung der Sättigung wird das Maximum der doppelt logistischen Funktion bestimmt:

$$m_{\text{logit}} := \max_{p \in [0,1]} \text{logit}(p).$$

Das Maximum m_{logit} wird im Bereich $p \in [0,1]$ bestimmt, anstatt die obere Schranke L für die Sättigungsbestimmung zu verwenden, da die doppelt logistische Funktion nicht immer die obere Schranke erreicht, siehe Abbildung 2.9. In der Abbildung 2.9a wird die obere Schranke erreicht, während mit der Änderung eines Parameters die Funktion deutlich unter der oberen Schranke bleibt.



(a) $L = 0{,}8$, $k_1 = 20$, $k_2 = 10$, $x_1 = 0{,}02$ und $x_2 = 1$

(b) $L = 0{,}8$, $k_1 = 2$, $k_2 = 10$, $x_1 = 0{,}02$ und $x_2 = 1$

Abbildung 2.9: Zwei Beispiele für die doppelt logistische Funktion, mit verschiedenen Parametern.

Im Folgenden wird ein Modell mit einem Störungsanteil p_i als gesättigt angesehen, wenn seine multifaktorielle Performanz $M_{p_i}(\tilde{S}) > (1-\alpha) \cdot m_{\text{logit}}$. Als p_{satt} wird das kleinste p_i gewählt, welches $M_{p_i}(\tilde{S}) > (1-\alpha) \cdot m_{\text{logit}}$ erfüllt. Wird das α zu groß gewählt ($\alpha >> 0$), dann kann ein zu kleines p_{satt} gewählt werden, wobei die volle Robustheit des Modells noch nicht erreicht wird. In dieser Arbeit wird $\alpha = 0{,}05$ gewählt. Die Festlegung von α basiert auf empirischen Beobachtungen.

Um die Zunahme der Robustheit bei p_{satt} zu quantifizieren, wird die multifaktorielle Performanz bei Sättigung $M_{p_{\text{satt}}}(\tilde{S})$ mit der multifaktoriellen Performanz des kleinsten evaluierten Anteils p_0, d.h., $M_{p_0}(\tilde{S})$, verglichen. Zusätzlich wird $M_{p_{\text{satt}}}(\tilde{S})$ mit der höchsten erreichten multifaktoriellen Performanz $M_{p_{\max}}(\tilde{S})$ verglichen. Dies dient dazu, das maximale Robustheitsniveau zu ermitteln und zu beurteilen, wie nahe $M_{p_{\text{satt}}}(\tilde{S})$ an diesem Maximum liegt.

Zur Veranschaulichung wird die angepasste logistische Funktion logit(p) in den Abbildungen, analog zur Abbildung 2.7 Punkt (13.), durch eine gepunktete rote Kurve dargestellt.

2.5 Konzept zur Robustheitssteigerung

Für ein Multisensorsystem werden das Blockdiagramm aus Abbildung 2.1 und die darin dargestellten einzelnen Konzepte zu einem umfassenden Konzept zur *Adaptive Robustheitssteigerung* überführt, wie es in Abbildung 2.10 gezeigt wird. Dieses Konzept zielt darauf ab, die Robustheit des Systems systematisch gegenüber verschiedenen Sensorstörungen zu erhöhen. Der Prozess beginnt mit einer ersten *Sensitivitätsanalyse*, bei der der Einfluss spezifischer Störungen auf das System untersucht wird, um einem Ausgangswert zu schaffen. Zu diesem Zweck wird die Sensorfusionsmethode zunächst auf ungestörten Daten trainiert, und anschließend auf gestörten Daten evaluiert, um die Auswirkungen der Störungen zu quantifizieren. Diese Analyse liefert entscheidende Erkenntnisse darüber, welche Störungen das System besonders stark beeinträchtigen und welche weniger kritisch sind.

Die Ergebnisse dieser ersten Sensitivitätsanalyse werden anschließend in das *Konzept zur Integration von Störungs-Adaptivität (KISA)* eingebettet. Auf Grundlage dieser Analyse wird entschieden, welche der identifizierten Störungen in den Trainingsprozess integriert werden, um die Robustheit des Systems gezielt zu verbessern. Dieser adaptive Ansatz gewährleistet, dass die Trainingsdaten durch eine gezielte Auswahl und Integration von Störungen optimiert werden, um das System widerstandsfähiger gegen diese spezifischen Störungen zu machen.

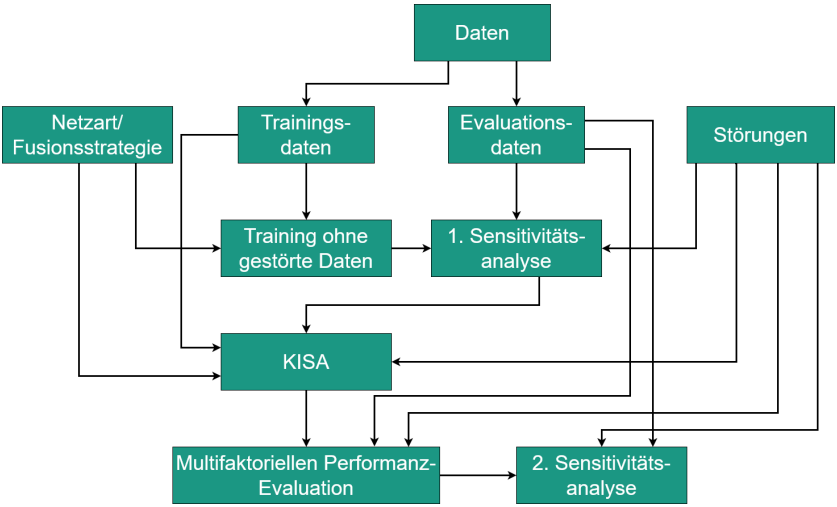


Abbildung 2.10: Schematische Darstellung des Konzeptes zur Steigerung der Robustheit. Das Diagramm zeigt den Ablauf vom initialen Training ohne gestörte Daten, gefolgt von der ersten *Sensitivitätsanalyse*, über die Anwendung von KISA, in der Störungen in den Trainingsprozess integriert werden. Die abschließende *Multifaktorielle Performanz-Evaluation* dient der Bestimmung des optimalen Anteils an Störungen und die zweite *Sensitivitätsanalyse* dient dazu das optimierte System zu überprüfen.

Nach der Integration der Störungen in die Trainingsdaten erfolgt eine *Multifaktorielle Performanz-Evaluation*, die den optimalen Anteil an Störungen bestimmt, der in das Training einfließen sollte, um ein möglichst robustes und

leistungsstarkes System zu erreichen. Diese Evaluation untersucht verschiedene Störungsgrade und analysiert die Leistung des Systems über ein Spektrum von Störungsgraden hinweg, um die Robustheit zu erhöhen.

Abschließend wird eine zweite *Sensitivitätsanalyse* durchgeführt, um zu überprüfen, wie das optimierte System auf die zuvor identifizierten Störungen reagiert. Diese abschließende Analyse zeigt, inwieweit die implementierten Anpassungen die Robustheit des Systems verbessert haben, und gibt Aufschluss darüber, ob weitere Optimierungsmaßnahmen notwendig sind. Der iterative Charakter dieses Ansatzes stellt sicher, dass das System kontinuierlich verfeinert wird, um eine maximale Leistungsfähigkeit und Robustheit in realen Einsatzszenarien zu gewährleisten.

3 Daten

3.1 Übersicht

In dieser Arbeit wird die entwickelte Methode KISA zur strukturierten Datenauswertung und die Evaluationsmethodiken Sensitivitätsanalyse und Multifaktorielle Performanz-Evaluation gemeinsam im neuen Konzept zur Robustheitssteigerung auf verschiedene Datensätze angewendet. In diesem Kapitel werden die Eigenschaften der verwendeten Datensätze und die Störungen, welche auf den Punktwolken angewendet werden, beschrieben. Für die Störungen auf Bilddaten sei auf den Anhang F.3 hingewiesen. Im Blockdiagramm von Abbildung 2.1 sind die *Benchmark Datensätze*, *Simulierte Daten* und *Störungen* in der *Modellierung von Dateneinflüssen* angelegt (siehe Abbildung 3.1).

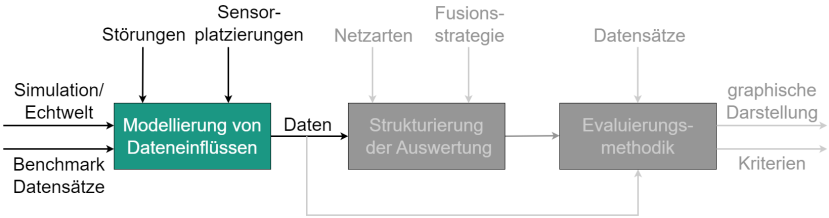
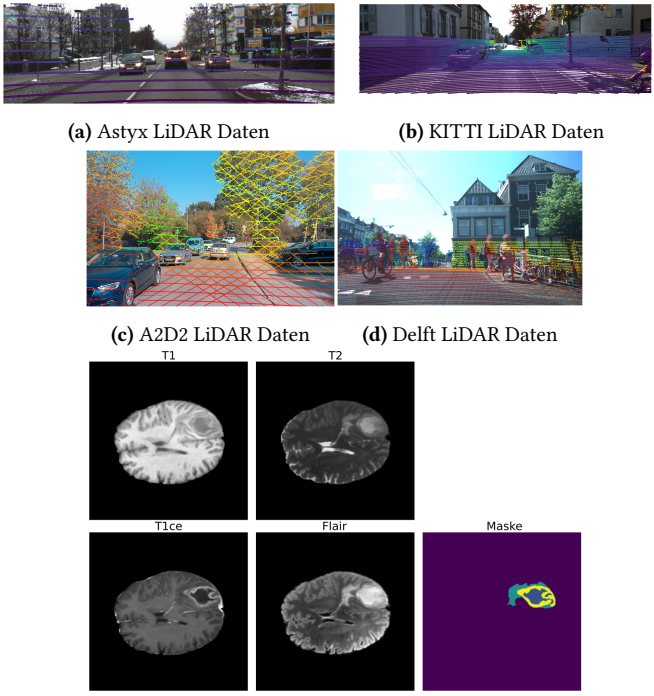


Abbildung 3.1: Dieses Kapitel beschreibt die Datensätze auf denen die neue Methode KISA und die neuen Evaluationsmethodiken Sensitivitätsanalyse und Multifaktorielle Performanz-Evaluation zusammen im neuen Konzept zur Robustheitssteigerung angewendet werden. Im Blockdiagramm von Abbildung 2.1 sind die *Benchmark Datensätze* und *Simulierte Daten* in der *Modellierung von Dateneinflüssen* angelegt.

Zu den verwendeten Datensätzen gehören die folgenden etablierte Echtwelt Datensätze: Astyx [Meye19], A2D2 [Geye20], Delft [Palf22], KIT-TI [Geig13] simulierte Daten für die semantische Segmentierung [Münk24]

und BraTS [Menz14, Baka17, Baid21, Hata21]. Die Auswahl dieser Datensätze basiert darauf, dass die Datensätze verschiedene Sensormodalitäten abdecken, wodurch die Leistungsfähigkeit von KISA in komplexen, multimodalen Szenarien demonstriert wird. Abbildung 3.2 zeigt jeweils ein Beispiel aus den Datensätze. In Tabelle 3.1 sind die Eigenschaften der Datensätze aufgelistet. Neben diesen Echtwelt Datensätzen werden simulierte Daten für die Anwendung von KISA bei semantischer Segmentierung verwendet. Die Generierung dieser simulierten Daten wird in Abschnitt 3.3 dargestellt.



(a) Astyx LiDAR Daten **(b)** KITTI LiDAR Daten

(c) A2D2 LiDAR Daten **(d)** Delft LiDAR Daten

(e) BraTS - MRT Daten und Segmentierungskarte

Abbildung 3.2: Beispiele für die Daten aus dem Astyx, A2D2, Delft, und KITTI und BraTS Datensatz.

Ein Faktor bei der Anwendung von KISA ist die Anzahl der Sensoren, die für die Fusion verwendet werden. Je mehr Sensoren verwendet werden, desto aufwendiger ist es, das System gegen Störungen robuster zu gestalten. Weitere wichtige Kriterien sind die Sensormodalität und die Sensorhomogenität. Die Sensormodalität beschreibt die Art der erfassten Daten, z.B. visuelle Bilder von Kameras, Tiefeninformationen von LiDAR-Sensoren oder Reflexionen von Radarwellen. Diese verschiedenen Modalitäten liefern unterschiedliche Perspektiven und Informationen über die Umgebung. Sensorhomogenität hingegen bezieht sich auf die Konsistenz und Einheitlichkeit der von verschiedenen Sensoren erfassten Daten. Homogene Sensoren, wie z.B. mehrere LiDAR-Sensoren, erfassen ähnliche Datentypen, was die Fusion und Verarbeitung vereinfacht. Heterogene Sensoren, die unterschiedliche Modalitäten verwenden, erfordern komplexere Fusionstechniken, um die verschiedenen Informationen zu integrieren und eine kohärente Darstellung der Umgebung zu erzeugen.

Tabelle 3.1: Datensatzeigenschaften der verwendeten Datensätze. Dabei sind bei Sensorik nur die Sensoren aufgeführt, welche auch in der vorliegenden Arbeit verwendet werden. Homogenität bezieht sich auf die Ähnlichkeit der Datenmodalität.

Datensatz	Sensorik	Sensormodalität	Homogenität
Astyx	LiDAR/RADAR	Punktwolken	+
A2D2	3 x LiDAR	Punktwolken	+
Delft	LiDAR/RADAR	Punktwolken	o
KITTI	LiDAR/Kamera	Punktwolken/Bilder	-
BraTS	MRT	Bilder	+

3.2 Echtwelt Datensätze

Die Datensätze Astyx, A2D2, Delft und KITTI konzentrieren sich auf Anwendungen im Bereich des autonomen Fahrens. Alle diese Datensätze enthalten Sensordaten von Kamera, RADAR- und LiDAR-Sensoren. Der KITTI-Datensatz verwendet nur einen LiDAR-Sensor (Velodyne-64e), während der A2D2-Datensatz Daten von fünf LiDAR-Sensoren (Velodyne Puck) enthält, von denen drei nach vorne gerichtet sind. Der Astyx-Datensatz kombiniert LiDAR-Daten (Velodyne Puck) mit einem RADAR-Sensor, ebenso wie der

Delft-Datensatz, der einen LiDAR-Sensor (Velodyne HDL-64 S3) und einen RADAR-Sensor verwendet. Mit Ausnahme des KITTI-Datensatzes werden in dieser Arbeit nur die LiDAR- und RADAR-Daten der Datensätze verwendet. Die Sensormodalität für die LiDAR- und RADAR-Daten ist in den Datensätzen eine Punktwolke, wobei je nach Sensor und Datensatz neben der Lage des einzelnen Punktes im Raum noch weitere Informationen wie Intensität bei LiDAR Punkten oder relative radiale Geschwindigkeit bzw. Magnitude bei RADAR Punkten angegeben werden. Die Kameradaten werden immer als Bild dargestellt. Da bei den A2D2 Daten dreimal der gleiche Sensor verwendet wird, ist hier die Datenhomogenität am größten. Die LiDAR und RADAR-Daten im Astyx und Delft Datensatz sind homogen. Die LiDAR Punktwolke im Delft Datensatz ist jedoch dichter als die RADAR Punktwolke. Im Astyx Datensatz ist die Punktdichte in den Punktwolken ähnlich. Die Kamera- und LiDAR-Daten im KITTI-Datensatz sind aufgrund der Sensormodalität nicht homogen.

Der Brain Tumor Segmentation (BraTS)-Datensatz ist ein Benchmark-Datensatz zur Bewertung von Algorithmen zur Segmentierung von Hirntumoren. Dies unterscheidet ihn von den oben genannten Datensätzen. Der Datensatz enthält multimodale Magnetresonanztomographie (MRT)-Daten und manuelle Tumorsegmentierungen. Die vier MRT-Modalitäten sind: T1-gewichtet (T1), kontrastverstärktes T1-gewichtet (T1ce), T2-gewichtet (T2) und Flair (Fluid Attenuated Inversion Recovery). Die Homogenität zwischen den MRT-Modalitäten ist im BraTS Datensatz gegeben.

Keiner dieser Datensätze hat den Fokus auf Sensorstörung. So sind die Daten in den Datensätzen zum autonomen Fahren bei klaren Wetter aufgenommen worden. Um diese Einschränkung zu umgehen, werden Störungen auf die Daten simuliert.

Aufgrund der unterschiedlichen verwendeten Sensortypen, bei gegebener Homogenität der Sensordaten wird in dieser Arbeit der Astyx Datensatz am häufigsten für die Analyse des Trainings- und Evaluationskonzepts verwendet.

3.3 Simulierter Datensatz

Zur Analyse des Einflusses von Sensorstörungen auf die semantische Segmentierung wird ein synthetischer Datensatz erzeugt. Der Einsatz eines synthetischen Datensatzes erlaubt eine kontrollierte Untersuchung der grundlegenden Herausforderungen der Störungen, was bei der Verwendung realer Daten nicht umsetzbar ist.

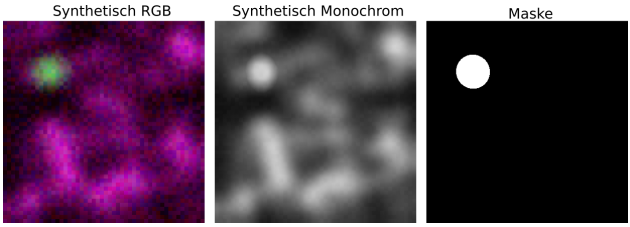


Abbildung 3.3: Beispiel für zufällig erzeugte Bilder aus dem synthetischen Datensatz ohne zusätzliche Störungen. Das linke Bild zeigt das Bild des RGB-Sensors mit einem Schwierigkeitsgrad von $D = 25\%$, das mittlere Bild zeigt das Bild des monochromen Sensors mit einem Schwierigkeitsgrad von $D = 15\%$, und das rechte Bild stellt die entsprechende Segmentierungskarte dar.

Der für die Analyse erzeugte synthetische Datensatz besteht aus insgesamt 5.120 Bildern, jedes mit einer Größe von 400×400 Pixeln. Die Bilder werden dabei in fünf Unterdatensätze mit jeweils 512 Trainingsdaten und 512 Testdaten zufällig verteilt. Damit lässt sich die Analyse auf fünf unabhängigen aber vergleichbaren Datensätzen wiederholen. Diese Aufteilung gewährleistet, dass jede Evaluierung unter vergleichbaren Bedingungen stattfindet und die Ergebnisse somit eine höhere Zuverlässigkeit und Repräsentativität aufweisen. Durch die Wiederholung der Analyse auf fünf separaten Unterdatensätzen wird zudem die Konsistenz der Methodik überprüft und die Robustheit der Ergebnisse gegenüber zufälligen Schwankungen in den Daten sichergestellt.

Der Prozess der Datensatzerstellung folgt den Richtlinien, die in [Münk24] beschrieben sind, und umfasst die folgenden Schritte:

1 **Hintergrund erstellen**: Der Hintergrund besteht aus 50-200 zufällig erzeugten gaußschen Verteilungen.

2 **Objekt einfügen**: Ein elliptisches Objekt wird an einer zufällig gewählten Position innerhalb des Bildes platziert. Die Dimensionen des Objekts variieren zufällig, und die Textur wird durch gleichförmiges Rauschen (siehe Schritt 4) erzeugt.

3 **Segmentierungskarte erzeugen**: Für das eingefügte elliptische Objekt wird eine entsprechende Segmentierungskarte erstellt. Diese Karte kennzeichnet exakt die Pixel, die das Objekt umfassen, indem sie diese von dem Rest des Bildes differenziert. Dabei werden Pixel, die zum elliptischen Objekt gehören mit 1 gekennzeichnet und solche, die zum Hintergrund gehören mit 0.

4 **Rauschen anwenden**: Dem Bild wird Rauschen hinzugefügt, um einen Grundschwierigkeitsgrad festzulegen. Der Schwierigkeitsgrad, $D \in [0, 100\%]$, bestimmt die Intensität des dem Bild hinzugefügten Rauschens. Die spezifische Rauscharten, die in diesem Schritt hinzugefügt werden, sind wie folgt:
"Unschärfe": Es wird ein normalisierter Boxfilter mit einer zufälligen Kernelgröße in $[0, D \cdot 400]$ angewendet.
"Gleichförmiges Rauschen": Für jeden Superpixel der Größe 8×8 der Bilder wird eine zufällige ganze Zahl in $[0, D \cdot 255]$ gezogen und zu dem Superpixel addiert oder subtrahiert.
"Farbverschiebung": Für jeden Kanal der Bilder wird eine zufällige ganze Zahl in $[0, D \cdot 255]$ gezogen und zu dem Kanal addiert oder subtrahiert.

Zur Analyse der Auswirkungen von Sensorstörungen auf die Datenfusion sind zwei verschiedene Sensormodalitäten erforderlich. Um dies zu realisieren, werden für jedes Objekt zwei Bilder generiert: Das erste Bild wird gemäß der Pipeline von [Münk24] mit einem Schwierigkeitsgrad von $D = 15\%$ erstellt, während das zweite ebenfalls nach der Pipeline, aber mit einem Schwierigkeitsgrad von $D = 25\%$, erzeugt wird. Weiterhin wird das Bild mit $D = 15\%$

in den Graustufenraum konvertiert, indem der maximale Wert der drei Farbkanäle genutzt wird. Das Bild mit $D = 25\%$ hingegen bleibt unverändert. Dieses Vorgehen dient der Simulation eines monochromen und eines RGB-Sensors. Beide Bilder weisen denselben Hintergrund auf, und das Objekt bleibt hinsichtlich Größe und Position identisch. Unterschiede ergeben sich jedoch in den angewandten Texturen auf das Objekt, sowie in der Variabilität des hinzugefügten Rauschens. Abbildung 3.3 veranschaulicht ein Beispiel dieser beiden Sensoren samt der zugehörigen Segmentierungskarte.

3.4 Betrachtete Störungen auf Punktwolken

In diesem Abschnitt werden die simulierten Sensorstörungen vorgestellt, die in dieser Arbeit auf die Punktwolkedaten angewendet werden.

3.4.1 Definition und Parametrisierung der Punktwolkestörungen

Die Implementation der Vorgestellten Störungen "Punkte hinzufügen", "Punkte verlieren", "Punkte verschieben", "Information verrauscht" und "Cluster" sind auf GitHub im Zuge dieser Arbeit veröffentlicht worden[1]. Für die Störungen "Nebel" und "Schnee" sei auf die original GitHub Veröffentlichung hingewisen[2,3].

1 **"Punkte hinzufügen"**: Diese Störung fügt der bestehenden Punktwolke zusätzliche, zufällig generierte Punkte hinzu. Diese neuen Punkte werden innerhalb eines definierten räumlichen Bereichs, der sogenannten Region of Interest (ROI), generiert. Bei dem für den Astyx-Datensatz beträgt die ROI [0 m, 50 m] × [-25 m, 25 m] × [-2 m, 2 m] (siehe Abbildung K.1). Je nach Schwierigkeitsgrad und Größe der

[1] https://github.com/BerensRWU/PointCloud_Distortion/tree/main
[2] https://github.com/MartinHahner/LiDAR_fog_sim
[3] https://github.com/SysCV/LiDAR_snow_sim

Originalpunktwolke n_P werden $n_P \cdot 25\%$, $n_P \cdot 50\%$, $n_P \cdot 75\%$ oder $n_P \cdot 100\%$ Datenpunkte zu der Punktwolke hinzugefügt. Die zufälligen Werte für die Intensität bei LiDAR oder für die relative Geschwindigkeit bzw. Magnitude bei RADAR werden aus einer Gleichverteilung innerhalb des Wertebereichs der Originaldaten zufällig gezogen. Mögliche Ursachen für diese Störung können Softwarefehler in der Datenverarbeitung oder auch Umweltstörungen wie Nebel sein, die zu einer Verfälschung der ursprünglichen Sensorinformationen führen. Die Auswirkungen umfassen eine deutliche Verringerung der Datenqualität und -genauigkeit.

2 **"Punkte verlieren"**: Bei dieser Störung werden zufällig ausgewählte Punkte aus der Punktwolke entfernt. Je nach Schwierigkeitsgrad können $n_P \cdot 25\%$, $n_P \cdot 50\%$, $n_P \cdot 75\%$ oder $n_P \cdot 100\%$ der Punkte verloren gehen. Dies kann durch Hardware-Fehler, Software-Bugs, Signalverarbeitungsprobleme, Kommunikationsfehler oder durch Umweltstörungen wie Nebel verursacht werden. Diese Störung führt zu einer verringerten Genauigkeit der Umgebungserfassung und kann Fehlinterpretationen hervorrufen.

3 **"Punkte verschieben"**: Bei dieser Störung werden alle Datenpunkte einer Punktwolke Φ zufällig verschoben. Die Verschiebung jedes Punktes $\Phi[i]$ erfolgt nach der folgenden Formel:

$$\Phi[i] = \Phi[i] + b_{\text{versch}} \cdot \frac{\Phi[i]}{||\Phi[i]||_2},$$

wobei der Verschiebungsfaktor b_{versch} je nach Schwierigkeitsgrad gleichmäßig aus einem der Intervalle $[-0{,}5, 0{,}5]$, $[-1, 1]$, $[-1{,}5, 1{,}5]$ oder $[-2, 2]$ gezogen wird. Der Faktor b_{versch} skaliert den normierten Vektor $\frac{\Phi[i]}{||\Phi[i]||2}$, wodurch die Verschiebung proportional zur ursprünglichen Entfernung des Punktes vom Ursprung ist. Größere Intervalle für b_{versch} bedeuten eine stärkere Verschiebung der Punkte, was größere Verzerrungen in der Punktwolke zur Folge hat. Diese Störung kann sowohl bei LiDAR- als auch bei RADAR-Sensoren auftreten, wenn es zu Kalibrierungsfehlern, mechanischen Vibrationen

oder externen Interferenzen kommt, die die präzise Erfassung der Positionen von Objekten beeinträchtigen und somit zu einer verzerrten Darstellung der Umgebung führen.

4 **"Cluster"**: Je nach Schwierigkeitsgrad der Störung wird eine Gruppe von Punktclustern zur Punktwolke hinzugefügt. Aufgrund der Art der Datenaufnahme ist ein Cluster für Daten vom LiDAR-Sensor kugelförmig, und für Daten vom RADAR-Sensor kubisch. Abhängig vom Schwierigkeitsgrad sind zufällig 1-4, 2-7, 7- 12 oder 14-19 Cluster vorhanden. Der Radius für einen Cluster auf LiDAR Daten bzw. die Breite und Länge für einen Cluster bei RADAR Daten wird gleichmäßig Verteilt aus den Intervallen [0,225 m, 1,125 m], [0,6 m, 1,5 m], [1,225 m, 2,125 m] oder [2,1 m, 3 m]. Die Höhe für einen Cluster bei RADAR Daten wird immer aus dem Intervall [0 m, 1 m] gezogen. Bei LiDAR-Sensoren können solche Cluster durch Aerosole entstehen, die die Laserstrahlen reflektieren und somit falsche Datenpunkte generieren. Im Gegensatz zu Nebel, welcher im gesamten Bereich liegt, sind Störungen durch "Cluster" lokal begrenzt. Dies simuliert Effekte wie Abgase, Staub oder Rauch, die in realen Umgebungen auftreten können. Bei RADAR-Sensoren können Störungen durch Interferenzen oder den Multipatheffekt entstehen. Der Multipatheffekt tritt auf, wenn Signale von Objekten mehrfach reflektiert werden und so unerwartete Pfade nehmen, bevor sie zum Sensor zurückkehren. Dadurch können die Signale an verschiedenen Positionen und zu unterschiedlichen Zeiten empfangen werden.

5 **"Information verrauscht"**: Die zusätzlichen Informationen $\Phi[i,3]$ für jeden Punkt, Intensität für LiDAR-Punkte, Magnitude oder relative Geschwindigkeit für RADAR-Punkte, werden um einen Faktor verschoben. Es wird ein zufälliger Verschiebungsfaktor a_{info} erzeugt. Dieser Faktor wird, je nach Schwierigkeitsgrad, gleichmäßig aus einem Intervall $[-25\%,25\%]$, $[-50\%,50\%]$, $[-75\%,75\%]$ oder $[-100\%,100\%]$ gezogen. Für jeden Punkt aus der Punktwolke $\Phi[i] \in \Phi$ wird die zusätzliche Information verrauscht:

$$\Phi[i,3] = \Phi[i,3] + a_{\text{info}} \cdot a_{\max},$$

wobei $a_{\max}$ der Maximalwert von Intensität ($a_{\max}$ = 255) bzw. Magnitude ($a_{\max}$ = 102 oder V_r = 5,20) ist.
Störungen durch zufällige Verschiebung von Datenpunkten in einer Punktwolke können durch Kalibrierungs- oder Softwarefehler, mechanische Vibrationen und externe Interferenzen. Speziell bei Sensoren, die mittels Time-Of-Flight (TOF) die Distanz messen, kann diese Art von Fehler auch durch fehlerhafte Zeitmessungen entstehen. Die Distanz zu Objekten wird auf Basis der Laufzeit von Signalen gemessen, wodurch selbst geringfügige Zeitmessfehler zu deutlichen Verschiebungen in den erfassten Punktdaten führen können.

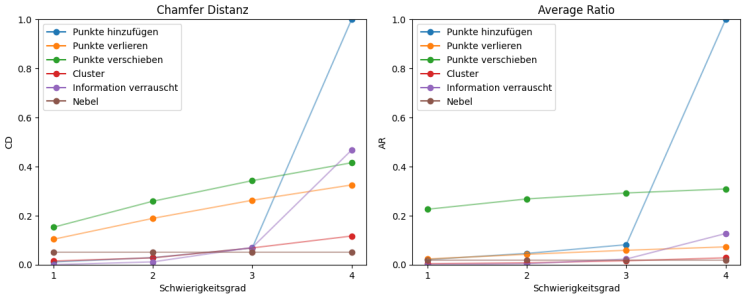
6 **”Nebel”**: In Anlehnung an [Hahn21] wird Nebel auf der LiDAR-Punktwolke simuliert. Der simulierte Nebeldichte wird durch einen Dämpfungskoeffizient (α) angepasst. Dieser ändert die Reichweite und Intensität jedes ursprünglichen Punktes bei klarem Wetter, sodass die neuen Werte den Messungen entsprechen würden, als wäre Nebel in der Szene vorhanden. Dieser Koeffizient beschreibt wie viel Energie verloren geht, während der Laser durch den Nebel geht. Für mathematische Details sei auf den Anhang I verwiesen. Abhängig vom Schwierigkeitsgrad beträgt α: Grad 0: 0, Grad 1: 0,01, Grad 2: 0,02, Grad 3: 0,04, Grad 4: 0,08. Dies reicht von nur leichtem Nebel (Grad 1) bis zu starkem Nebel (Grad 4).

7 **”Schnee”**: In Anlehnung an [Hahn22] wird Schnee auf der LiDAR-Punktwolke simuliert. Dabei wird nur eine Dichte von Schnee betrachtet, weswegen diese Störung nur einen Schwierigkeitsgrad hat. Für mathematische Details sei auf den Anhang J verwiesen.

Die zugehörigen Parameter bestimmen die Stärke der Störung. Damit ist die Menge aller betrachteten Störungen $\mathbf{S}$ = {”Punkte verlieren Grad 1”, …, ”Punkte verlieren Grad 4”, ”Punkte hinzufügen Grad 1”, …, ”Punkte hinzufügen Grad 4”, ”Punkte verschieben Grad 1”, …, ”Punkte verschieben Grad 4”, ”Information verrauscht Grad 1”, …, ”Information verrauscht Grad 4”, ”Cluster Grad 1”, …, ”Cluster Grad 4”, ”Nebel Grad 1”, …, ”Nebel Grad 4”, ”Schnee”}. So entspricht beispielsweise eine Störung ”Punkte verlieren Grad 3” dem zufälligen Verlust

von 60% der Punkte der Punktwolke. Für visuelle Beispiele aller Störungen sei auf den Anhang O verwiesen.

3.4.2 Quantitative Beurteilung und Gruppierung der Störung



(a) Chamfer Distanz **(b)** Average Ratio

Abbildung 3.4: Vergleich der Störungsarten und ihrer Schwierigkeitsgrade in Bezug auf über 125 LiDAR Punktwolken gemittelte Werte der CD und AR.

Für 125 LiDAR-Punktenwolken aus dem Astyx Datensatz wurden die Werte der Metriken CD und AR zwischen der gestörten und der ungestörten Originalpunktwolke berechnet. Die Ergebnisse dieser Analyse sind in den Abbildungen 3.4 visualisiert.

Die in den Abbildungen 3.4 dargestellten Daten zeigen deutlich die Auswirkungen der Störungen auf die Qualität der Punktwolke, mit einer klaren Tendenz zur Verschlechterung der Qualität mit steigendem Schwierigkeitsgrad. Ein auffälliges Beispiel ist die Zunahme der Chamfer Distanz bei der Störungsart "Punkte hinzufügen"von 0,10 bei Schwierigkeitsgrad 1 auf einen Wert von 0,32 bei Schwierigkeitsgrad 4. Dieser Anstieg signalisiert eine deutliche Verschlechterung der Punktwolkenqualität. Ähnliche Muster sind auch bei AR, zu beobachten. AR zeigt zunehmende geometrische Unterschiede zwischen der ursprünglichen und der gestörten Punktwolke mit steigendem Schwierigkeitsgrad.

Um die Störungen zu gruppieren ist, in Abbildung 3.5 ein Dendrogramm bezüglich der verschiedenen Störungsarten und Schwierigkeitsgrade dargestellt. Für das Dendrogramm in Abbildung 3.5 ist die Chamfer Distanzen paarweise zwischen allen Störungsarten und Schwierigkeitsgraden berechnet. Die Werte wurden über 125 LiDAR Punktwolken aus dem Astyx Datensatz gemittelt.

Auffällig ist insbesondere die Störungsart ”Punkte verlieren” bei Schwierigkeitsgrad 4, die sich signifikant von allen anderen Störungen abhebt. Dies lässt sich dadurch erklären, dass bei dieser Störung sämtliche Punkte verloren gehen. Darüber hinaus verdeutlicht das Dendrogramm der Chamfer Distanzen, dass die Störungen ”Punkte verschieben” und ”Punkte hinzufügen” jeweils eigene Gruppen bilden, die ausschließlich die verschiedenen Schwierigkeitsgrade umfassen. Außerdem werden ”Nebel” auf Grad 3 und ”Schnee” zusammen gruppiert, was darauf hindeutet, dass sie einen ähnlichen Einfluss auf die LiDAR-Daten ausüben.

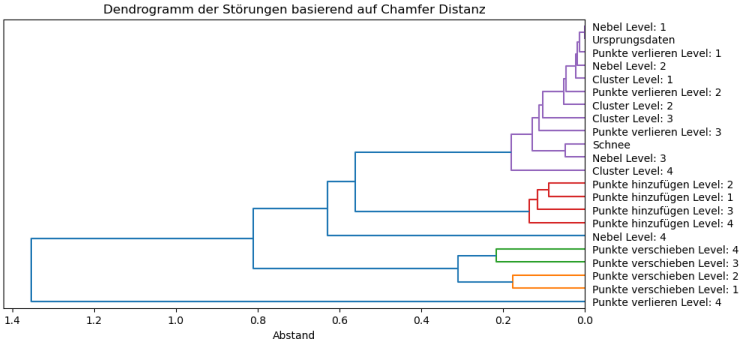


Abbildung 3.5: Dendrogramm, das die hierarchische Gruppierung aller analysierten Punktwolkestörungen darstellt. Für die Clusteranalyse wurden die Abstände zwischen den Störungen auf Basis der Metrik der Chamfer Distanz berechnet und als Mittelwert über 125 Punktwolken des Astyx Datensatzes bestimmt. Die Farben zeigen an, welche Störungen gruppiert werden.

4 Implementation

4.1 Übersicht

In diesem Kapitel wird beschrieben, wie die Implementierung der im Rahmen dieser Arbeit entwickelten Konzepte durchgeführt wurde und wo die Implementierungen veröffentlicht wurden.

4.2 Sensorplatzierung

Eine Implementierung für Genetische Algorithmus-Optimierung für Sensorplatzierung (GAOS) wurde auf GitHub veröffentlicht[1] und eine Implementierung für Deep Learning-Optimierung für Sensorplatzierung (DLOS) wurde ebenfalls auf GitHub veröffentlicht[2]. Beide Implementierungen sind vollständig in Python realisiert, wobei für die DDPG Methode in DLOS die Programmbibliothek Tensorflow verwendet wird.

Für die Simulation der Optimierungsumgebung und der Sensoren wird die Simulationsumgebung CARLA [Doso17] in der Version 0.9.10 verwendet. Die Optimierungsumgebung wird standardmäßig mit einem Zylinder von 10 m Durchmesser und einem Audi e-tron Modell erstellt, auf dem die Sensoren platziert werden. Sowohl der Durchmesser als auch das Fahrzeugmodell können gewechselt werden. Nach dem Erstellen der Optimierungsumgebung starten die jeweiligen Optimierungsalgorithmen GAOS bzw. DLOS.

[1] https://github.com/BerensRWU/GeneticOpt
[2] https://github.com/jdrew1/SensorOpt

Eine Liste der gesetzten Parameter für GAOS und DLOS sind in Anhang E angegeben.

Es wurden Zeitmessungen für beide Methoden GAOS bzw. DLOS durchgeführt, um deren Effizienz zu vergleichen, die in [Bere21b, Bere24a] veröffentlicht wurden. Dabei wurde die Platzierung eines Sensors auf einer 2,60-GHz-CPU optimiert. Für die DLOS Methode dauert ein Trainingsschritt etwa 5,4 Sekunden, was zu einer Gesamttrainingsdauer von 1 Stunde und 32 Minuten für 200 Epochen führt. Im Gegensatz dazu benötigt der Optimierungsprozess mit GAOS auf derselben Maschine 2 Stunden und 47 Minuten.

Darüber hinaus bietet das DDPG-Netzwerk den Vorteil, dass es nicht nur für ein spezifisches Fahrzeug optimiert werden kann. Einmal trainiert, ist das DDPG-Netzwerk in der Lage, optimale Sensorplatzierungen für neue Fahrzeugformen zu ermitteln, die während des Trainings nicht berücksichtigt wurden. Dieser Bewertungsprozess dauert durchschnittlich nur 4 Sekunden.

4.3 Implementation Datenauswertung und Evaluierungsmethodik

Eine Implementierung für KISA mit Complex-YOLO wurde auf GitHub veröffentlicht[1]. Diese Implementierung umfasst sowohl die Low und High Level Fusion, wie auch alle drei Complex-YOLO Modelle. Die Implementierung ist vollständig in Python realisiert und nutzt die Programmbibliothek PyTorch in Version 1.13.0+cu117. Eine detaillierte Übersicht über die Schichten der einzelnen Complex-YOLO-Architekturen ist in Anhang H dargestellt.

Die Implementierung für die Multifaktorielle Performanz-Evaluation und Sensitivitätsanalyse für Complex-YOLO ist ebenfalls auf GitHub[2] verfügbar. Die Implementierungen sind modular aufgebaut, womit eine Anpassung an andere Objektdetektoren einfach zu gestalten ist.

[1] https://github.com/BerensRWU/FusionComplexYOLO
[2] https://github.com/BerensRWU/EvaluationConcept

4.4 Verwendete Complex-YOLO Modelle

Für die Analyse der Forschungsfrage 4 (Wie muss ein Netzwerk zur Integration gestörter Daten gestaltet werden?) werden drei Complex-YOLO Modelle mit unterschiedlicher Anzahl von Merkmalsextraktionsschichten und Erkennungsköpfen betrachtet. Wenn von einem anderen Modell als dem ursprünglichen Complex-YOLO-Modell die Rede ist, wird dies durch ein Präfix gekennzeichnet:

- Kleines Complex-YOLO
- Mini Complex-YOLO

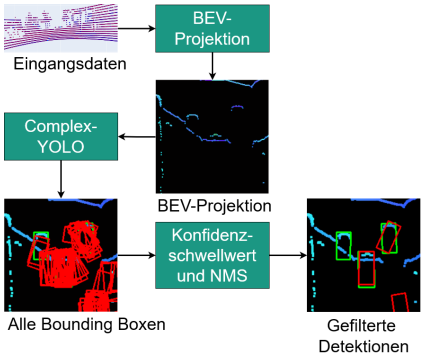


Abbildung 4.1: Schematische Darstellung des Datenflusses bei Complex-YOLO, beispielhaft mit LiDAR Daten. Für eine detaillierte Auflistung der einzelnen Schichten der Complex-YOLO Architekturen siehe Anhang H.

Die Complex-YOLO Implementation basiert auf YOLOv4 und besteht aus 162 Merkmalsextraktionsschichten und drei Erkennungsköpfen an unterschiedlichen Tiefen des Netzwerks. Kleines Complex-YOLO hat 38 Schichten und zwei Erkennungsköpfe. Mini Complex-YOLO besteht aus insgesamt 17 Schichten und einem Erkennungskopf. Die Erkennungsköpfe verwenden jeweils drei verschiedene Ankerpunkte.

Die Abbildung 4.1 zeigt schematisch den Datenfluss von Complex-YOLO. Complex-YOLO erhält als Eingabedaten eine Punktwolke. Diese Punktwolke wird in eine Bird's-Eye-View Projektion (BEV-Projektion) umgewandelt.

Aus dieser BEV-Projektion werden Merkmale extrahiert, aus denen eine Liste von Bounding Boxen erstellt wird. Durch Konfidenzschwellenwert und Non-Maximum Suppression (NMS) wird die Liste von Bounding Boxen gefiltert. Eine detaillierte Übersicht über den Aufbau der Complex-YOLO Architekturen ist in Anhang H dargestellt.

4.4.1 Erweiterung des Complex-YOLO Ansatzes durch Sensor Fusion

Wie bereits in Abschnitt 1.5 beschrieben, gibt es drei Strategien für die Sensorfusion, die sich nach der Abstraktionsebene der Fusion unterscheiden: Low Level, Mid Level und High Level Fusion. Von diesen Strategien sind die Low und High Level Fusion modellagnostisch anwendbar, was bedeutet, dass die hier gewonnenen Erkenntnisse auf eine breite Palette von Objektdetektoren übertragbar sind. Im Gegensatz dazu ist die Mid Level Fusion stark von der spezifischen Implementierung abhängig, weshalb Aussagen darüber oft nur auf das jeweilige Modell zutreffen. Aus diesen Gründen konzentriert sich diese Arbeit auf die Low und High Level Fusion.

Der ursprüngliche Complex-YOLO Algorithmus war für die Verarbeitung eines einzelnen Sensors konzipiert. In dieser Arbeit wird der Algorithmus erstmals um die Fähigkeit zur Sensorfusion erweitert, sodass er mit mehreren Sensoren gleichzeitig arbeiten kann.

Die erste untersuchte Methode zur Objektdetektion ist die Low Level Fusionsstrategie. Hierbei werden die Punktwolken zweier Sensoren – in diesem Fall LiDAR und RADAR – zu einer einzigen, gemeinsamen Punktwolke fusioniert. Diese Fusion auf niedriger Abstraktionsebene bietet eine detailreiche und umfassende Darstellung des erfassten Raumes, da die Stärken beider Sensoren kombiniert werden. Der Datenfluss dieser Low Level Fusion ist in Abbildung 4.2 schematisch dargestellt. Detaillierte Informationen zur Methode der Punktwolkenfusion sind in Anhang B zu finden. Die Low Level Fusionsstrategie wird in dieser Arbeit erstmals auf den Complex-YOLO Algorithmus angewendet.

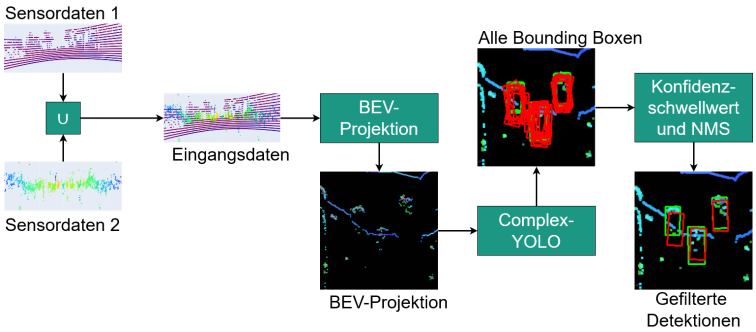


Abbildung 4.2: Schematische Darstellung des Datenflusses bei der Fusion mit Complex-YOLO Low Level.

Die zweite untersuchte Methode zur Sensorfusion ist die High Level Fusionsstrategie. Im Gegensatz zur Low Level Fusion, bei der die Rohdaten der Sensoren direkt kombiniert werden, verfolgt die High Level Fusion einen zweistufigen Ansatz. Der Datenfluss dieser Strategie ist in Abbildung 4.3 dargestellt.

Im ersten Schritt der High Level Fusion werden die Punktwolken der verschiedenen Sensoren in ein gemeinsames Koordinatensystem transformiert, wie es auch bei der Low Level Fusion der Fall ist. Anschließend werden die transformierten Daten jedoch unabhängig voneinander verarbeitet, um aus jeder Punktwolke eine spezifische Objektliste zu extrahieren. Dies führt zu separaten Objektlisten ($\mathcal{O}_1, \mathcal{O}_2$), die die erkannten Objekte in den jeweiligen Datensätzen repräsentieren.

Im zweiten Schritt werden diese individuellen Objektlisten zusammengeführt, um eine gemeinsame Objektliste $\mathcal{O}$ zu erstellen. Dieser Schritt umfasst zunächst die Vereinigung der Listen ($\mathcal{O}_1 \cup \mathcal{O}_2$), gefolgt von einer gewichteten Non-Maximum Suppression (NMS):

$$\mathcal{O} = \text{NMS}(\mathcal{O}_1 \cup \mathcal{O}_2).$$

Durch die Anwendung der gewichteten Non-Maximum Suppression (NMS) wird sichergestellt, dass jedes Objekt nur einmal und mit der höchstmöglichen Erkennungsgenauigkeit in der finalen Liste vertreten ist. In Anhang H ist

eine detaillierte Beschreibung der Schritte der gewichteten Non-Maximum Suppression zu finden.

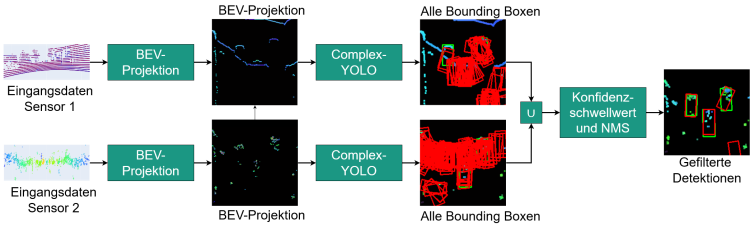


Abbildung 4.3: Schematische Darstellung des Datenflusses bei der Fusion mit Complex-YOLO High Level.

4.4.2 Laufzeitvergleich

Die Tabellen 4.1 und 4.2 zeigen die Laufzeiten für das Training einer Epoche bzw. die Auswertung einer Dateninstanz unter den Konfigurationen: Low Level-Fusion, nur LiDAR-Daten und nur RADAR-Daten. High Level-Fusion ist in den Tabellen nicht berücksichtigt, da hierbei separate Modelle für LiDAR und RADAR verwendet werden, die jeweils spezifisch trainiert werden. Über alle Modelle hinweg ist die RADAR-Konfiguration am schnellsten. Dies liegt daran, dass die Verarbeitung von RADAR-Daten weniger rechenintensiv ist, da die RADAR-Punktwolken weniger Punkte enthalten als die LiDAR-Daten. Das originale Complex-YOLO-Modell weist in allen drei Konfigurationen die höchsten Laufzeiten auf, was auf seine höhere Komplexität und Größe zurückzuführen ist. Die Modelle wurden für 300 Epochen trainiert. Das Complex-YOLO Low Level-Modell wurde unter anderem mit KISA trainiert, das 6 Störungen, 4 Schwierigkeitsgrade und 10 unterschiedliche Prozentwerte für den Anteil integrierter Störungen umfasst. Damit hat KISA das Complex-YOLO Low Level-Modells insgesamt 32 Tage lang trainiert.

Tabelle 4.1: Vergleich der Laufzeiten (in s) für das Training einer Epoche der verschiedenen Complex-YOLO-Modelle. Getestet wurde auf einer NVIDIA A40 mit CUDA Version 12.2. Die Batchgröße betrug 8, und es wurden 434 Daten verwendet.

	Low Level Fusion	LiDAR	RADAR
Complex-YOLO	37,6	37,4	27,7
Kleines Complex-YOLO	21,9	23,6	10,5
Mini Complex-YOLO	20,0	20,6	8,5

Tabelle 4.2: Vergleich der Laufzeiten (in s) für die Auswertung eines Dateninstanz der verschiedenen Complex-YOLO-Modelle. Getestet wurde auf einer NVIDIA A40 mit CUDA Version 12.2.

	Low Level Fusion	LiDAR	RADAR
Complex-YOLO	0,07	0,06	0,05
Kleines Complex-YOLO	0,01	0,01	0,01
Mini Complex-YOLO	0,01	0,01	0,01

4.5 Erweiterung der U-Net Methode durch Sensor Fusion

Für die semantische Segmentierung wird in dieser Arbeit U-Net verwendet. U-Net ist ein vollständig auf Faltungsoperationen basierendes neuronales Netzwerk. Es ist für die Segmentierung von medizinischen Bildern entwickelt und mehrfach angewendet [Ronn15, Caic19, Sche20, LeCl21, Schi22] wirden.

Die in dieser Arbeit verwendete Implementierung von U-Net und die Erweiterungen zur Sensorfusion, wurde auf GitHub veröffentlicht[1]. Eine detaillierte Übersicht über den Aufbau der U-Net Architektur ist in Anhang F dargestellt.

Ein U-Net, welches die Low Level oder High Level Fusionsstrategie verwendet, wird im Folgenden als U-Net Low Level bzw. High Level bezeichnet.

Für die Low Level Fusion werden die Bilddaten zweier verschiedener Sensoren systematisch zusammengeführt, indem die Kanäle verbunden werden. Damit ist

[1] https://github.com/BerensRWU/FusionUNet

die Kanalanzahl der resultierenden Bilddaten die Summe der Kanalanzahlen der einzelnen Bilddaten. Diese kanalweise Methode für die Low Level Fusion ist nur möglich, wenn beide Sensoren die gleiche Auflösung haben. Die fusionierten Bilddaten werden dann in das U-Net geladen. Die U-Net Architektur muss für diese Low Level Fusionsstrategie, bis auf die Anzahl der Eingangskanäle, nicht geändert werden.

Die High Level Fusionsstrategie bei semantischer Segmentierung beinhaltet einen zweistufigen Prozess. In der ersten Phase dieser Strategie werden alle verfügbaren Bilddaten unabhängig voneinander in einem eigenen U-Net verarbeitet. Dabei wird für jedes Bild eine individuelle Segmentierungskarte erstellt. In der zweiten Phase werden die erzeugten Segmentierungskarten zusammengeführt, um eine einzige Segmentierungskarte zu erstellen. Diese Fusion wird durch die Berechnung des Mittelwerts der Wahrscheinlichkeitsverteilungen über die korrespondierenden Bildpunkte der verschiedenen Segmentierungskarten hinweg realisiert. Wie auch beim ursprünglichen U-Net wird zur eindeutigen Klassenzuordnung jedes Bildpunktes der Klassenindex mit dem höchsten Wahrscheinlichkeitswert ausgewählt.

5 Simulationsgestützte Optimierung der Sensorplatzierung

5.1 Übersicht

Dieses Kapitel wendet die beiden neuen Methoden Genetische Algorithmus-Optimierung für Sensorplatzierung (GAOS) und Deep Learning-Optimierung für Sensorplatzierung (DLOS) zur Bestimmung der optimalen Sensorplatzierung für Multi-LiDAR-Systeme in Fahrzeugen an. Im Blockdiagramm von Abbildung 2.1 ist die *Sensorplatzierung* in der *Modellierung von Dateneinflüssen* angelegt (siehe Abbildung 5.1). Die Ergebnisse mit GAOS wurden 2021 im IEEE Sensors Journal veröffentlicht [Bere21b]. Die mit DLOS wurden 2023 im SAE International Journal of Connected and Automated Vehicles veröffentlicht [Bere24a].

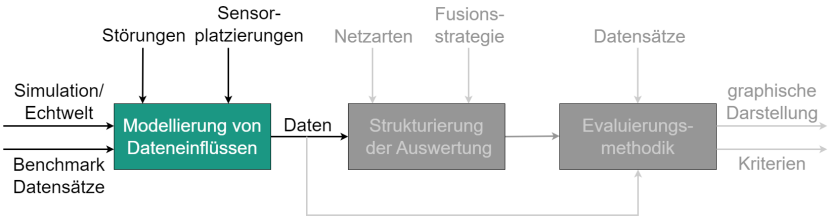
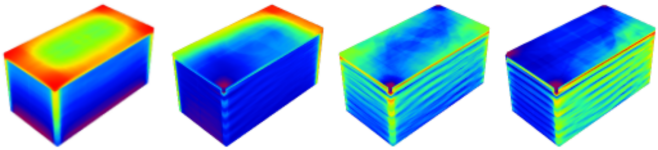


Abbildung 5.1: Dieses Kapitel beschreibt die Anwendung der beiden neuen Methoden GAOS und DLOS. Im Blockdiagramm von Abbildung 2.1 ist die *Sensorplatzierung* in der *Modellierung von Dateneinflüssen* angelegt.

Um die Wirksamkeit und Verallgemeinerbarkeit der vorgeschlagenen Methode zu evaluieren, werden im Rahmen dieser Arbeit Tests zur Optimierung der

LiDAR-Sensorplatzierung durchgeführt. Zunächst wird ein rechteckiger Quader als repräsentative Näherung für typische Fahrzeuggeometrien verwendet. Der Optimierungsprozess beinhaltet die Platzierung von LiDAR-Sensoren auf dieser vereinfachten Form, um die Leistungsfähigkeit der Methode unter kontrollierten Bedingungen zu evaluieren. Anschließend werden die Ergebnisse auf ein Modell eines Audi e-tron angewendet, das vom CARLA Simulator bereitgestellt wird.



(a) 1 Sensor **(b)** 2 Sensoren **(c)** 3 Sensoren **(d)** 4 Sensoren

Abbildung 5.2: Qualität der Platzierung für jede mögliche Position auf einem Quader. Für (a) ist kein LiDAR vorplatziert. Für (b) ist ein Sensor an der vorderen Ecke platziert. Für (c) sind zwei Sensoren fest platziert, einer an der vorderen Ecke und einer an der gegenüberliegenden Ecke. Für (d) sind drei Sensoren fest platziert, einer jeweils an den vorderen beiden Ecken und einer an der hinteren linken Ecke. Die Farbe zeigt an, wie gut die Platzierung eines neuen Sensors an dieser Stelle wäre. Die Farbskala geht von Lila schlechte Platzierung/niedriger $\%LO_1$ Wert bis rot sehr gute Platzierung/hoher $\%LO_1$ Wert.

Um den Raum zu analysieren, auf dem optimiert wird und welche Sensorplatzierung mit einem LiDAR-Sensor mit 16 Kanälen am besten ist, zeigt Abbildung 5.2 den Wert der Zielfunktion $\%LO_1$ für jede realisierbare Platzierung, wobei die Sensoren nicht rotiert werden. Dabei sind in den Fällen der Abbildung 5.2b - 5.2d bereits ein bis drei Sensoren auf den Ecken platziert. Im Fall von Abbildung 5.2a ist kein Sensor vorplatziert. Aus Abbildung 5.2a ist zu erkennen, dass die besten Standorte an den vier oberen Ecken der Box zu finden sind. An diesen Stellen werden 270° des horizontalen Sichtfeldes des Sensors nicht durch den Quader blockiert, in den restlichen 90° werden die unteren LiDAR-Kanäle durch den Quader blockiert. An den vier vertikalen Rändern werden 270° des Sensor-Sichtfeldes ebenfalls nicht blockiert. Daher haben diese Stellen einen höheren metrischen Wert als die Ebenen an den Seiten. Die restlichen 90° sind vollständig blockiert, weshalb die Zielfunktion $\%LO_1$ niedriger ist als

an den oberen Ecken des Quaders. In der Mitte des Kastens ist der Wert der Zielfunktion aufgrund der Blockierung der unteren LiDAR-Kanäle niedriger als an den Rändern. Aus Abbildung 5.2 zeigt sich, dass die Berechnung mit $\%LO_1$ zu vielen dominanten lokalen Maxima führt. Dies lässt sich dadurch erklären, dass der Wert von $\%LO_1$ nur dann durch einen neuen Laserstrahl erhöht wird, wenn dieser neue Laserstrahl einen Teilbereich belegt, der nicht bereits belegt ist. In den blauen Bereichen des Quaders in Abbildung 5.2 wird der neue LiDAR-Sensor viele Teilbereiche belegen, die bereits von einem fest platzierten LiDAR-Sensor eingenommen werden. Darüber hinaus hat $\%LO_1$ bei drei Sensoren ein Maximum unterhalb der Ecke, in der sich ein fester Sensor befindet. Dies ist darauf zurückzuführen, dass in diesem Fall die Laserstrahlen leicht versetzt sind und die beiden LiDAR-Sensoren daher unterschiedliche Teilbereiche belegen. Außerdem sinkt der Wert der Zielfunktion an der Ecke, an der sich bereits ein Sensor befindet, auf ein Minimum, da ein großer Teil des Sichtfeldes durch den feststehenden Sensor blockiert wird.

5.2 Genetische Algorithmus-Optimierung für Sensorplatzierung - Ergebnisse

GAOS wird zur Optimierung der Platzierung auf einem Quader angewendet, mit einem bis vier LiDAR-Sensoren mit jeweils 16 Kanälen. Im Fall von einem zu platzierenden Sensor, wird der Sensor vom GAOS auf eine Ecke platziert und erreicht einen Zielfunktionswert $\%LO_1$ von 0,75. Im Fall von zwei Sensoren wird ein Sensor auf einer Ecke und der zweite unterhalb der gegenüberliegenden Ecke platziert. Der Grund hierfür ist, dass der zweite Sensor unterhalb der oberen Ecke liegt und somit Teilbereiche auf der gegenüberliegenden Seite nicht erfassen kann. Dies führt zu einer Steigerung des $\%LO_1$ Zielfunktionwertes auf 0,96. Im Fall von drei und vier Sensoren platziert der Algorithmus einen Sensor an einer Ecke und die anderen Sensoren irgendwo auf der Box. Mit einem $\%LO_1$ Zielfunktionwert von 0,98 bzw. 0,99 hat der Algorithmus bereits Platzierungen gefunden, um fast alle Teilbereiche mindestens einmal zu erkennen, weswegen nicht alle Sensoren auf Ecken platziert werden.

Wird GAOS auf dem Modell des Audi e-tron angewendet, so sieht man in der oberen Reihe der Abbildung 5.5, dass die LiDAR-Sensoren mit 16 Kanälen immer auf dem Dach platziert werden. Ähnlich wie beim Quader haben die Sensoren auf dem Dach ein volles horizontales Sichtfeld von 360°, das nur von den anderen Sensoren verdeckt wird. Im Gegensatz zum Quader befinden sich die Sensoren nicht in den Ecken des Daches. Da das Dach konvex ist, platziert GAOS die Sensoren in der Nähe des höchsten Punktes des Daches. Bei zwei Sensoren platziert GAOS die Sensoren nebeneinander, rechts und links des Daches vom Modell. So nehmen die Sensoren die Vorder- und Rückseite vollständig mit beiden Sensoren wahr, während die Seiten meist nur von einem Sensor erfasst werden. Im Fall mit drei Sensoren platziert GAOS zwei Sensoren auf der Vorderseite auf beiden Seiten und einen auf der Rückseite. Die Platzierung führt daher zu einer hohen Auflösung der Vorder- und Rückseite. Die von GAOS errechnete Platzierung mit vier Sensoren ist ähnlich wie die mit drei Sensoren, wobei der vierte Sensor in der Mitte des Daches platziert wird, so dass er die anderen Sensoren nicht zu sehr verdeckt und eine geeignete Redundanz für alle drei Sensoren darstellt. Bemerkenswert ist, dass diese bestimmten Platzierungen nur gefunden werden können, wenn alle Sensoren gleichzeitig optimiert werden. Mit einem Algorithmus, der die Sensoren seriell platziert, werden diese Platzierungen nicht gefunden. Denn wenn ein Sensor aus der Platzierung eliminiert wird, muss die daraus entstehende Platzierung mit einem Sensor weniger nicht optimal sein.

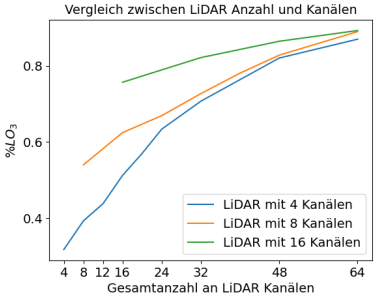


Abbildung 5.3: Ergebnisse der Zielfunktionswerte $\%LO_3$ für die Platzierung auf einem Audi e-tron Modell, berechnet durch den genetischen Optimierungsalgorithmus nach 2000 Schritten. Verglichen werden LiDAR-Sensoren mit 4, 8 und 16 Kanälen.

Mit der GAOS Methode wird des Weiteren analysiert, ob es vorteilhafter ist, eine größere Anzahl von Sensoren mit wenigen LiDAR-Kanälen zu verwenden, oder ob eine geringere Anzahl von Sensoren mit einer höheren Anzahl an Kanälen vorzuziehen ist. Abbildung 5.3 vergleicht diese beiden Ansätze und zeigt, dass die Sensortypen mit zunehmender Anzahl an Sensoren ähnliche Optimierungswerte erreichen. Für jeden der drei betrachteten Sensortypen ist die Leistung monoton steigend. Die Zielfunktionswerte $\%LO_3$ des 16-Kanal-LiDAR-Systems sind etwas besser als die der 4- und 8-Kanal-LiDAR-Systeme. Dies liegt daran, dass das Dach des Fahrzeugs leicht gewölbt ist. Wenn ein einzelner LiDAR-Sensor am höchsten Punkt des Fahrzeugs angebracht ist, ermöglicht dies eine vollständige 360° Wahrnehmung des Zylinders. Für die gleiche Anzahl von Kanälen wie das 16-Kanal-LiDAR-Sensorsystem benötigen das 4-Kanal-LiDAR-Sensorsystem und das 8-Kanal-LiDAR-Sensorsystem viermal bzw. zweimal so viele LiDAR-Sensoren, so dass sich wesentlich mehr Sensoren gegenseitig überdecken, was zu einem schlechteren Zielfunktionswert $\%LO_3$ führt.

5.3 Deep Learning-Optimierung für Sensorplatzierung - Ergebnisse

Die experimentellen Ergebnisse mit DLOS und einem Quader sind in Tabelle 5.1 zusammengefasst. Die Tabelle zeigt die Leistung von DLOS in verschiedenen Trainingsszenarien und vier zu platzierenden LiDAR-Sensoren mit 16 Kanälen. DLOS wird 20 Mal in der CARLA-Umgebung trainiert. Wenn auf dem Quader selbst trainiert wird, ergibt sich ein durchschnittlicher Zielfunktionswert $\%LO_3$ von 0,94 mit einer Standardabweichung von 0,03. Im Vergleich dazu erreicht DLOS, wenn es auf allen 33 Fahrzeugmodellen, welche in CARLA vorliegen, trainiert wird einen durchschnittlichen Zielfunktionswert $\%LO_3$ von 0,90 mit einer Standardabweichung von 0,11 für die Platzierung der Sensoren auf dem Quader. Mit zunehmend weniger Fahrzeugmodellen (22 Fahrzeuge) sinkt der durchschnittliche Zielfunktionswert $\%LO_3$ auf 0,85 mit einer Standardabweichung von 0,16 für die Platzierung auf dem Quader.

Tabelle 5.1: Durchschnittlicher Zielfunktionswert $\%LO_3$ erreicht von DLOS auf einem Quader und 4 LiDAR-Sensoren mit 16 Kanälen nach 200 Trainingsiterationen bei unterschiedlichen Trainingsfahrzeugmodellen. Die Ergebnisse stammen aus 20 Wiederholungen in der CARLA-Umgebung. Bei der Verwendung von GAOS beträgt der Zielfunktionswert $\%LO_3$ 0,92.

Trainingsszenario	Durchschnittliche $\%LO_3$	Standardabweichung
Quader	0,94	0,03
33 Fahrzeuge	0,90	0,11
22 Fahrzeuge	0,85	0,16

Zur Veranschaulichung zeigt Abbildung 5.4 die Ergebnisse eines Experiments, bei dem die 33 in CARLA verfügbaren Fahrzeugmodelle in Trainings- und Evaluierungsmodelle für DLOS aufgeteilt werden. Es zeigt sich, dass sich mit steigender Anzahl der Trainingsfahrzeuge auch der Zielfunktionswert $\%LO_3$ der vorgeschlagenen Sensorplatzierungen verbessert. Wird DLOS mit 22 Fahrzeugen trainiert, ergibt sich für die verbleibenden 11 Fahrzeugmodelle ein durchschnittlicher Zielfunktionswert $\%LO_3$ von 0,78. Dieser Wert steigt auf 0,91, wenn die Trainingsmenge auf 32 Fahrzeuge erhöht wird. Die Fähigkeit des DLOS, die Sensorplatzierung für verschiedene Fahrzeugmodelle zu optimieren, ohne für dieses Fahrzeugmodell trainiert worden zu sein, ist ein wesentlicher Vorteil des DLOS.

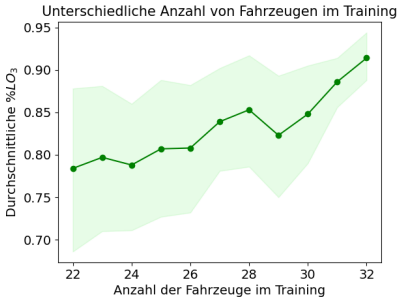


Abbildung 5.4: Ergebnisse der Zielfunktion $\%LO_3$ für die Platzierung von LiDAR Sensoren auf einem Fahrzeug mit DLOS, wenn die Anzahl der für das Training verwendeten Fahrzeuge steigt. Die restlichen Fahrzeuge werden für die Auswertung verwendet. Die Graphik stammt aus der Publikation [Bere24a].

In der unteren Reihe der Abbildung 5.5 wird DLOS verwendet für die Platzierung von 1, 2, 3 oder 4 LiDAR Sensoren mit 16 Kanälen auf einem Audi e-tron Modell. Hierfür wird DLOS nur auf dem Audi e-tron Modell trainiert. Die Platzierungen von DLOS können mit den Platzierungen, welche GAOS berechnet hat, verglichen werden. Auch DLOS bevorzugt beim Audi e-tron eine Sensorplatzierung auf dem Dach.

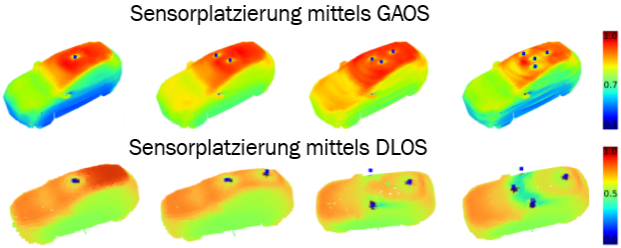


Abbildung 5.5: Die obere Reihe zeigt die Ergebnisse mittels GAOS aus [Bere21b]. Die untere Reihe zeigt die Ergebnisse mit DLOS aus [Bere24a]. Die Heatmaps veranschaulichen die $\%LO_3$-Werte, die durch die Einführung eines zusätzlichen Sensors an allen in Frage kommenden Positionen erreicht werden. Die blauen Punkte zeigen die von den Optimierungsalgorithmen hergeleiteten Platzierungsergebnisse für 1, 2, 3 oder 4 Sensoren. Angelehnt an eine Abbildung aus [Bere24a].

5.4 Bewertung

Mit den hier vorgestellten neu entwickelten Methoden GAOS und DLOS zur Bestimmung der optimalen Sensorplatzierung kann die Sensorplatzierung effizient bearbeitet werden.

Der Einsatz von Algorithmen wie dem DDPG in DLOS bietet eine präzise und schnelle Möglichkeit, Sensoren zu platzieren, wodurch die Entwicklungszeit und -kosten für autonome Fahrzeuge signifikant reduziert werden. Da DLOS auf eine Vielzahl von Fahrzeuggeometrien angewendet werden kann, ermöglicht er auch die effiziente Handhabung neuer Fahrzeugmodelle ohne erneutes Training.

Obwohl die Methoden speziell für LiDAR-Sensoren und Fahrzeuge entwickelt werden, können sie mit entsprechenden Anpassungen auch in anderen Bereichen wie der Robotik oder der Luft- und Raumfahrt, sowie für andere Sensormodalitäten wie RADAR-Sensoren eingesetzt werden. Hierfür werden entsprechende Simulationsumgebungen benötigt, in denen die Fahrzeugmodelle, auf denen die Sensoren platziert werden, generiert und die RADAR-Sensoren physikalisch modelliert werden können. Damit tragen die entwickelten Methoden zur allgemeinen Verbesserung von Technologien bei, die auf sensorbasierte Wahrnehmung angewiesen sind.

6 Steigerung der Robustheit bei fusionierten LiDAR/RADAR Daten

6.1 Übersicht

Dieses Kapitel betrachtet die Steigerung der Robustheit bei fusionierten LiDAR/RADAR Daten für die Objektdetektion. Dafür wird KISA und die Evaluationsmethodiken angewendet (siehe Abbildung 6.1). Dabei liegt der Schwerpunkt auf der Anpassung des Complex-YOLO Modells. Complex-YOLO steht stellvertretend für Punktwolke verarbeitende Methoden, die Aussagen sind jedoch allgemeingültig. Die Entscheidung für Complex-YOLO basierte auf seiner Effizienz. Complex-YOLO ermöglicht ein schnelles Training, die Bewertung vieler Modelle und zeigt außerdem, wie etablierte Modelle von dieser neuen Sensorfusionsstrategie profitieren können, wodurch die Relevanz dieser Ergebnisse erweitert wird. Erstmals wird Complex-YOLO so modifiziert (siehe Abschnitt 4.4.1), dass es Daten aus unterschiedlichen Sensortypen effizienter zusammenführen kann und wird damit erstmals für die Fusion von LiDAR und RADAR Daten verwendet. Eine systematische Untersuchung von Low Level und High Level Fusionsansätzen ermöglicht neue Erkenntnisse darüber, wie unterschiedlich diese die Genauigkeit und Störanfälligkeit des Detektionsprozesses beeinflussen. Die betrachteten Störungen auf den Punktwolken werden in Abschnitt 3.4 beschrieben.

Die Erweiterung von Complex-YOLO für die Low Level Fusion und die Ergebnisse für die Anwendung des Trainings- und Evaluationskonzepts für die Low Level Fusion mit Complex-YOLO sind 2024 in IEEE Transactions on Intelligent Vehicles veröffentlicht worden [Bere24b].

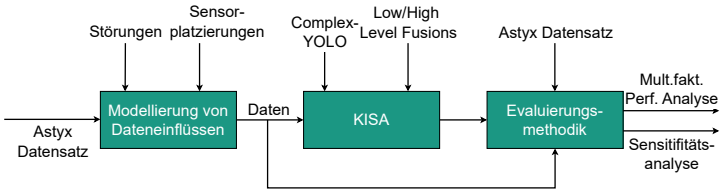


Abbildung 6.1: Blockdiagramm nach Abbildung 2.1, mit den gewählten Parametern für diese Auswertung. Die Sensorplatzierung ist dabei durch die Wahl des Astyx Datensatzes bereits gegeben.

Nach dem Konzept zur Steigerung der Robustheit wird zunächst die erste Sensitivitätsanalyse betrachtet, wenn im Trainingsprozess keine Störungen hinzugefügt werden. Anschließend wird KISA für einzelne Störungen und schließlich für die Robustheit gegenüber mehreren Störungen durchgeführt.

Im Folgenden wird als zusätzliche Information der RADAR Punkte immer die relative radiale Geschwindigkeit verwendet und die zusätzliche Information der LiDAR Punkte wird die Intensität verwendet.

6.2 Erste Sensitivitätsanalyse von Complex-YOLO

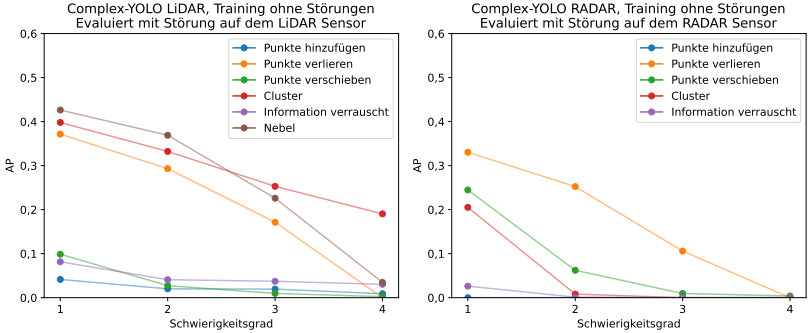
Wie im Konzept beschrieben, wird zunächst die erste Sensitivitätsanalyse durchgeführt. Dabei wird jedes der untersuchten Complex-YOLO-Modelle ausschließlich mit unveränderten Originaldaten trainiert, dies bedeutet $p_0 = 0\%$. Die Leistungsfähigkeit des Modells wird sowohl bei der Verarbeitung einzelner Sensordaten als auch bei der Fusion von LiDAR und RADAR Daten untersucht. Qualitative Ergebnisse sind in Anhang P dargestellt.

Die quantitativen Ergebnisse aus Tabelle 6.1, zeigen, dass Complex-YOLO eine bessere Leistung auf LiDAR Daten als auf RADAR Daten hat, so beträgt die Average Precision für Complex-YOLO bei Original-LiDAR-Daten ohne Störungen 0,44, für Complex-YOLO Original-RADAR-Daten: 0,36. Es zeigt sich, dass die Anwendung einer High Level Fusion die Average Precision

bei ungestörten Daten nicht verbessert, so dass diese mit einem Wert von 0,45 auf einem vergleichbaren Niveau wie die Ergebnisse des Complex-YOLO LiDAR mit 0,44 bleibt. Im Gegensatz dazu führt die Anwendung der Low Level Fusion zu einer deutlichen Steigerung der Detektionsleistung. Hier erreicht die Average Precision für Complex-YOLO Low Level einen Wert von 0,59, was eine deutliche Verbesserung um 34% gegenüber dem Modell Complex-YOLO LiDAR ohne Datenfusion darstellt.

Tabelle 6.1: Quantitative Ergebnisse von Complex-YOLO, trainiert und evaluiert auf den Originaldaten (Werte in Average Precision, AP).

LiDAR	RADAR	Low Level Fusion	High Level Fusion
0,44	0,36	0,59	0,45



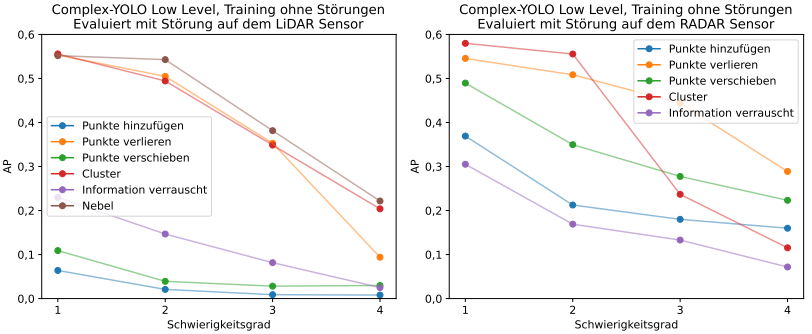
(a) Complex-YOLO LiDAR mit Störung auf den LiDAR Daten **(b)** Complex-YOLO RADAR mit Störung auf den RADAR Daten

Abbildung 6.2: Sensitivitätsanalyse von Complex-YOLO mit Original-LiDAR-Daten 6.2a trainiert beziehungsweise Complex-YOLO mit Original-RADAR-Daten 6.2b trainiert. Die ursprüngliche Average Precision beträgt für Complex-YOLO bei Original-LiDAR-Daten: 0,44 und für Complex-YOLO bei Original-RADAR-Daten: 0,36.

Die Ergebnisse der ersten Sensitivitätsanalyse für das Complex-YOLO, das mit den Original-LiDAR-Daten bzw. mit den Original-RADAR-Daten trainiert wurde, sind in Abbildungen 6.2 dargestellt. Es ist zu erkennen, dass es einen deutlichen Zusammenhang zwischen dem Schwierigkeitsgrad und der Average Precision gibt. Die optimale Leistung wird bei ungestörten Daten erreicht

(Complex-YOLO LiDAR: 0,44, Complex-YOLO RADAR: 0,36). Mit zunehmendem Schwierigkeitsgrad der Evaluationsdaten nimmt die Modellgüte deutlich ab (mittlere Average Precision über alle Störungsarten bei Schwierigkeitsgrad 4 für Complex-YOLO LiDAR: 0,04, für Complex-YOLO RADAR VR: 0,00). Leichte Störungen, wie sie durch ”Punkte verlieren” (Schwierigkeitsgrad 1 bis 3) verursacht werden, haben nur einen geringen Einfluss auf beide Modelle, während bei starken Störungen ”Punkte verlieren Grad 4” die Average Precision auf 0 sinkt, da in diesem Fall alle Punkte der Punktwolke verloren gehen und somit keine Objektdetektion mehr möglich ist. Auch die verschiedenen Schwierigkeitsgrade an Störung durch Nebel auf den LiDAR Daten führen zu einer Verschlechterung der Leistung von Complex-YOLO LiDAR, so ist die Average Precision bei ”Nebel Grad 2” 0,37, bei ”Nebel Grad 3” 0,23 und schließlich bei ”Nebel Grad 4” 0,03.

Die Sensitivitätsanalyse für die auf Datenfusion basierenden Complex-YOLO Modelle wird durch die Abbildungen 6.3 bis 6.4 dargestellt.

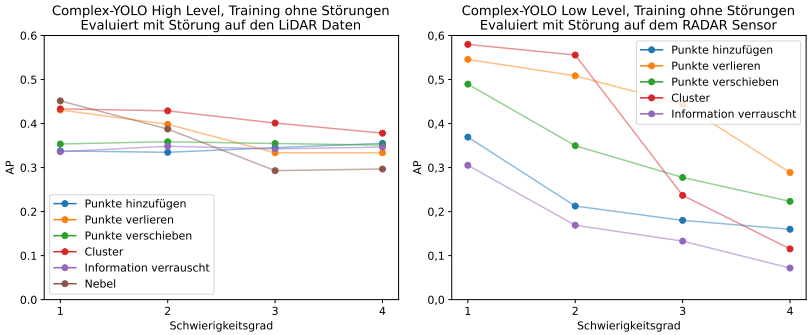


(a) Complex-YOLO Low Level mit Störung auf den LiDAR Daten

(b) Complex-YOLO Low Level mit Störung auf den RADAR Daten

Abbildung 6.3: Erste Sensitivitätsanalyse von Complex-YOLO Low Level, welches auf den Original-LiDAR und RADAR-Daten trainiert wurde, wenn die Störung in den LiDAR-Evaluationsdaten vorhanden ist 6.3a und wenn sie in den RADAR-Evaluationsdaten vorhanden ist 6.3b. Die ursprüngliche Average Precision beträgt für Complex-YOLO Low Level bei Original-LiDAR und RADAR-Daten: 0,59.

Die erste Sensitivitätsanalyse in Abbildung 6.3 verdeutlicht, dass Störungen auf LiDAR- oder RADAR Daten die Performanz des Complex-YOLO Low Level Modells in ähnlicher Weise beeinflussen wie die Modellvarianten ohne Datenfusion. Störungen wie ”Punkte verlieren” (Mittelwert über Schwierigkeitsgrade 1-4: 0,38), ”Cluster” (0,40) und ”Nebel” (0,42) auf LiDAR Daten haben einen relativ geringen Einfluss auf die Modellleistung. Im Gegensatz dazu führen Störungen wie ”Punkte hinzufügen” (0,03), ”Punkte verschieben” (0,05) und ”Information verrauscht” (0,12) zu einer stärkeren Reduktion der Leistung. Ähnliches Verhalten lässt sich für den Fall, dass die Störung auf den RADAR Daten vorliegt erkennen.



(a) Complex-YOLO High Fusion mit Störung auf den LiDAR Daten

(b) Complex-YOLO High Fusion mit Störung auf den RADAR Daten

Abbildung 6.4: Erste Sensitivitätsanalyse von Complex-YOLO High Fusion, welches auf den Original-LiDAR und RADAR-Daten trainiert wurde, wenn die Störung in den LiDAR-Evaluationsdaten vorhanden ist 6.4a und wenn sie in den RADAR-Evaluationsdaten vorhanden ist 6.4b. Die ursprüngliche Average Precision beträgt für Complex-YOLO High Fusion bei Original-LiDAR und RADAR-Daten: 0,45

Die in den Abbildungen in 6.4 dargestellte erste Sensitivitätsanalyse für Modelle mit High Level Fusion offenbart eine Verbesserung der Modellleistung über einen weiten Bereich von Störungen. Diese Verbesserung erreicht jedoch nur das Leistungsniveau des Complex-YOLO Modells, welches auf den Original-LiDAR-Daten trainiert wurde, wenn die RADAR Evaluationsdaten gestört sind, beziehungsweise entspricht der Leistung des Complex-YOLO Modells,

welches auf den Original-RADAR-Daten trainiert wurde, wenn die LiDAR Evaluationsdaten gestört sind. Die multifaktorielle Performanz von Complex-YOLO High Level beträgt $M_{p_0} = 0{,}35$.

Sowohl die Low Level als auch die High Level Fusion haben sich als wirksame Strategien zur Verbesserung der Erkennungsgenauigkeit und der Robustheit der Modelle gegenüber sensorischen Störungen erwiesen. Während die Low Level Fusion insbesondere bei spezifischen Störungsarten Vorteile bietet, ermöglicht die High Level Fusion eine allgemeine Verbesserung der Modellleistung, die jedoch an die Grenzen der ungestörten Sensordaten gebunden ist.

6.3 Anwendung von KISA für eine Störung bei Complex-YOLO Low Level Fusion

In diesem Abschnitt wird der nächste Schritt des Konzeptes zur Steigerung der Robustheit betrachtet, indem das KISA Konzept auf die Complex-YOLO-Fusionsmodelle angewendet wird. Dabei wird zusätzlich zu den ungestörten Daten mit einer spezifischen Störung trainiert.

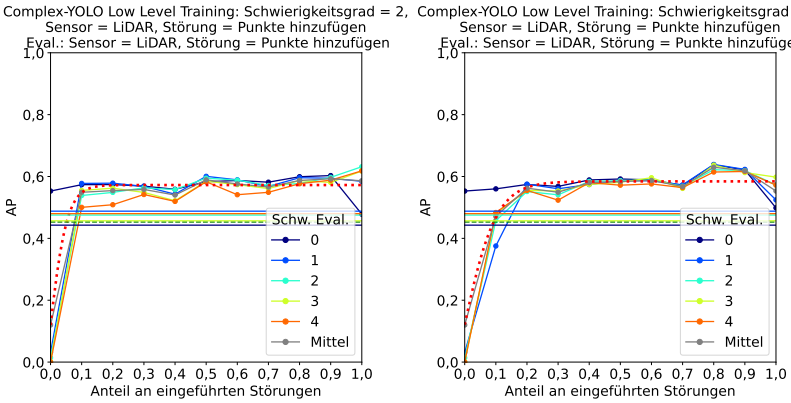


Abbildung 6.5: Multifaktorielle Performanz-Evaluation von KISA mit Complex-YOLO Low Level und Störungen auf den LiDAR Daten. Links: "Punkte hinzufügen Grad 2". Rechts: "Punkte hinzufügen Grad 4". Notation wie zu Abbildung 2.7 ausführlich beschrieben.

Wie die Ergebnisse aus den Abschnitten 3.4 und 6.2 zeigen, hat sich die Störung ”Punkte hinzufügen” als herausfordernd für Complex-YOLO-Modelle erwiesen. Daher wird hier KISA beispielhaft an dieser Störung betrachtet. Dazu wird das Complex-YOLO Low Level Modell mit den Anteilen $p \in \{0\%, 10\%, 20\%, 30\%, 40\%, 50\%, 60\%, 70\%, 80\%, 90\%, 100\%\}$ beziehungsweise $\mathbf{S}$ = {”Punkte hinzufügen Grad 2”}, $\mathbf{S}$ ={”Punkte hinzufügen Grad 4”} trainiert. Die Ergebnisse nach der Multifaktoriellen Performanz-Evaluation sind in den Abbildungen 6.5 - 6.6 dargestellt. Die Ergebnisse der Multifaktoriellen Performanz-Evaluation für KISA mit weiteren speziellen Störungen sind im Anhang Q aufgeführt.

Für die Störungen ”Punkte hinzufügen” lässt sich aus den Abbildungen 6.5 und 6.6 erkennen, dass die Leistung von Complex-YOLO Low Level nicht nur für den jeweiligen Schwierigkeitsgrad, auf das bei KISA abgezielt wird, verbessert wird, sondern auch Verbesserungen bei höheren Schwierigkeitsgraden zeigt, sowohl wenn die Störung auf den LiDAR Daten vorliegt (Abbildung 6.5) als auch wenn die Störung ”Punkte hinzufügen” auf den RADAR Daten vorliegt (Abbildung 6.6).

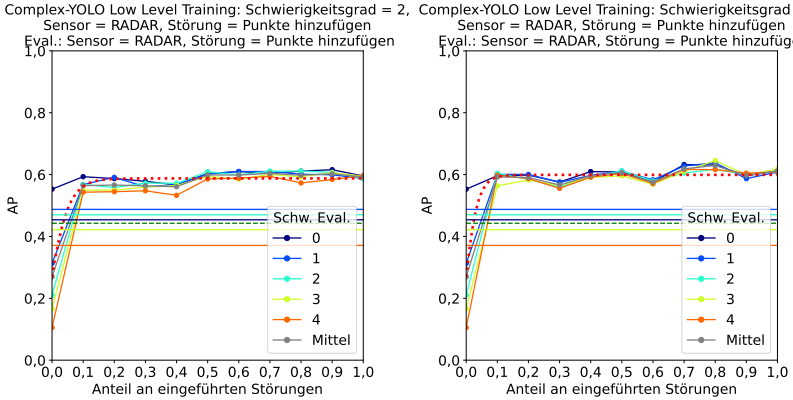


Abbildung 6.6: Multifaktorielle Performanz-Evaluation von KISA mit Complex-YOLO Low Level und Störungen auf den RADAR Daten. Links: ”Punkte hinzufügen Grad 2”. Rechts: ”Punkte hinzufügen Grad 4”.

Die Leistung bei ”Punkte hinzufügen Grad 4” auf den LiDAR Daten erreicht eine ähnliche Average Precision von 0,56 bei p = 20%, wie auf den Originaldaten

ohne zusätzliche Störung (Average Precision bei p = 0% 0,57). Die Verbesserung durch KISA ist bereits mit einem Anteil von p = 10% an eingeführten Störungen im Trainingsprozess deutlich. So wird nach Anwendung von KISA das Complex-YOLO Low Level Fusion Modell überhaupt erst einsetzbar auf LiDAR Evaluationsdaten mit der Störung "Punkte hinzufügen Grad 2" (Steigerung um 5300%, wenn mit p = 10% LiDAR Trainingsdaten mit "Punkte hinzufügen Grad 2" trainiert wird, verglichen mit Trainingsdaten ohne zusätzliche Störung, siehe Abbildung 6.5 links). Eine Plateau wird für "Punkte hinzufügen Grad 2" bei einem Anteil der in den Trainingsdaten integrierten Störung von p_{satt} = 10% erreicht, wenn die Störung in den LiDAR-, RADAR- oder beiden Sensordaten vorliegt. Bei p_{satt} = 10% ist die multifaktorielle Performanz $\boldsymbol{M}_{p_{\text{satt}}}(\hat{S}) = 0{,}55$ in allen drei Fällen. Abbildung 6.5 zeigt, dass bei einem Anteil von p = 100% an Trainingsdaten mit der hinzugefügten Störung die Leistung auf den anderen Schwierigkeitsgraden und ungestörten Daten wieder abfallen kann. Dies muss aber nicht der Fall sein, wie Abbildung 6.6 zeigt. Eine Ausnahme bildet die Evaluation auf Daten mit der spezifischen Störung, welche im Training integriert wird, diese fällt nicht ab.

Complex-YOLO-Modelle, die mit fusionierten Daten arbeiten, haben eine bessere Leistung als Modelle, die ohne fusionierte Daten, aber mit störungsfreien Daten arbeiten. So ist beispielsweise, wenn auf einen Anteil von p = 50% die Störung "Punkte hinzufügen Grad 4" auf die RADAR Trainingsdaten angewendet wird, die Average Precision 0,58, wenn "Punkte hinzufügen Grad 4" Störung auch auf den RADAR Evaluationsdaten vorliegt. Wohingegen ein Complex-YOLO LiDAR, welches nur mit den ungestörten LiDAR Daten arbeitet, eine Average Precision von nur 0,45 auf ungestörten LiDAR Daten hat (siehe Abbildung 6.6 gestrichelte Linie rechts).

In Tabelle 6.2 sind für Complex-YOLO Low Level die Mittelwerte der Sättigungswerte und der maximalen multifaktoriellen Performanzwerte für die einzelnen Störungsarten über die Schwierigkeitsgrade aufgeführt. Nach Tabelle 6.2 gibt es keinen Zusammenhang zwischen dem Einfluss der Störung auf die multifaktorielle Performanz von Complex-YOLO Low Level ohne integrierte Störung p_0 = 0% und dem Anteil des Erreichens der Sättigung. So ist $\boldsymbol{M}_{p_0}(\hat{S})$ = 0,17 bei Complex-YOLO Low Level für die Störung "Punkte hinzufügen"

relativ gering, die Sättigung p_{satt} wird jedoch im Durchschnitt bereits bei 18% integrierter Störung bei Complex-YOLO Low Level erreicht. Im Gegensatz dazu hat die Störung "Schnee" bei Complex-YOLO Low Level ebenfalls einen starken Einfluss auf die Performanz, die Sättigung p_{satt} wird jedoch erst bei 40% erreicht. Weiterhin zeigt die Tabelle 6.2, dass die maximal erreichbare multifaktorielle Performanz nur geringfügig über der multifaktoriellen Performanz am Sättigungswert liegt, zum Beispiel ist die multifaktorielle Performanz für "Punkte verschieben" bei Complex-YOLO Low Level nur 0,01 höher. Dies deutet darauf hin, dass das Netz noch über freie Kapazitäten zur Anpassung an weitere Störungen verfügt. Die Tabelle 6.2 zeigt, dass Störungen, die Informationen über die Position und Geometrie löschen, wie "Punkte verlieren" und "Punkte verschieben", nur bis zu einem gewissen Grad kompensiert werden können und die Average Precision unterhalb des Ausgangswertes von 0,59 (siehe Tabelle 6.1) bleiben. Störungen, die neue Informationen hinzufügen, wie "Punkte hinzufügen" oder "Cluster", können vollständig ausgeglichen werden und Complex-YOLO Low Level erreicht ein ähnliche Average Precision, wie auf Evaluationsdaten ohne Störung. Diese Kompensation ist jedoch nur möglich, wenn die Störungen während des Trainings integriert werden. Die Störungen "Nebel" und "Schnee" können ebenfalls gänzlich kompensiert werden, da sie nur die LiDAR Daten verändern, aber die RADAR Daten unverändert lassen.

Tabelle 6.2: Mittelwerte über alle Schwierigkeitsgrade hinweg für die multifaktorielle Performanz $M_p(\hat{S})$ von Complex-YOLO Low Level, beim Training ohne Störung p_0, bei dem Anteil an dem die Sättigung p_{satt} erreicht wird und bei dem Anteil, bei dem die maximale multifaktorielle Performanz p_{max} erreicht wird, aufgeteilt nach den einzelnen Störungsarten.

Störungsart	$(p_0, M_{p_0}(\hat{S}))$	$(p_{\text{satt}}, M_{p_{\text{satt}}}(\hat{S}))$	$(p_{\text{max}}, M_{p_{\text{max}}}(\hat{S}))$
"Punkte hinzufügen"	(0%, 0,17)	(18%, 0,57)	(86%, 0,61)
"Punkte verlieren"	(0%, 0,40)	(24%, 0,47)	(70%, 0,50)
"Punkte verschieben"	(0%, 0,25)	(27%, 0,48)	(43%, 0,49)
"Cluster"	(0%, 0,39)	(23%, 0,54)	(86%, 0,58)
"Information verrauscht"	(0%, 0,22)	(38%, 0,54)	(80%, 0,57)
"Nebel"	(0%, 0,42)	(17%, 0,52)	(60%, 0,55)
"Schnee"	(0%, 0,12)	(40%, 0,56)	(50%, 0,57)

Über alle betrachteten Punktwolkenstörungsarten und Schwierigkeitsgrade beträgt der Mittelwert des Anteils p_{satt} bei dem die Sättigung erreicht wird 26%.

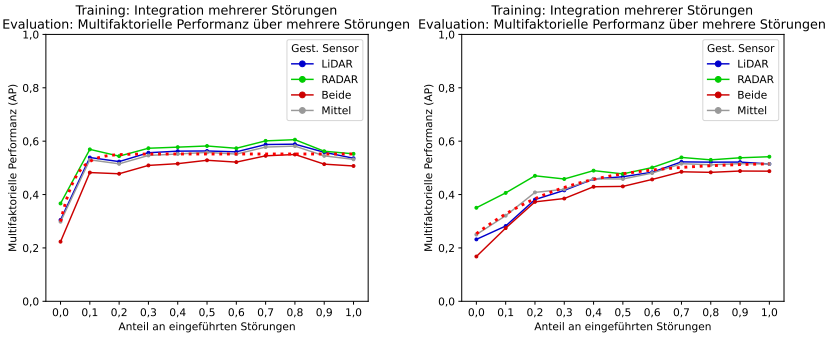
Ähnliche Aussagen können auch für die anderen hier nicht gezeigten Störungen getroffen werden. Die quantitativen Ergebnisse sind in Anhang Q aufgeführt. Insgesamt lässt sich daraus schließen, dass es mit dem vorgestellten KISA möglich ist, die Robustheit eines Objektdetektors gegen eine Störung zu erhöhen, bei gleichbleibender Leistung auf ungestörten Daten. Mit dem Evaluationskonzept kann diese Steigerung sichtbar gemacht werden.

6.4 Anwendung von KISA für mehrere Störung bei Complex-YOLO-Fusionsmodellen

Im vorherigen Abschnitt wurde untersucht, wie das vorgestellte KISA genutzt werden kann, um die Robustheit von Complex-YOLO für eine spezifische Störung zu erhöhen. In diesem Abschnitt wird untersucht, inwieweit es mit KISA möglich ist, die Robustheit von Complex-YOLO gegenüber mehreren Störungen gleichzeitig zu erhöhen. Konkret werden folgende Störungen berücksichtigt $\mathbf{S}$ = {"Punkte verlieren Grad 1", …, "Punkte verlieren Grad 4", "Punkte hinzufügen Grad 1", …, "Punkte hinzufügen Grad 4", "Punkte verschieben Grad 1", …, "Punkte verschieben Grad 4", "Information verrauscht Grad 1", …, "Information verrauscht Grad 4", "Cluster Grad 1", …, "Cluster Grad 4", "Nebel Grad 1", …, "Nebel Grad 4", "Schnee"} und es werden die Anteile $p \in \{0\%, 10\%, 20\%, 30\%, 40\%, 50\%, 60\%, 70\%, 80\%, 90\%, 100\%\}$ betrachtet.

Um die Forschungsfrage "Wie muss ein Netzwerk zur Integration gestörter Daten gestaltet werden?" zu untersuchen, wird in diesem Abschnitt verglichen, welche Fusionsabstraktionsebene für KISA besser geeignet ist und ob die Nettiefe einen Einfluss auf die Möglichkeiten der Robustheit gegenüber Störungen hat. Dazu werden die beiden Fusionsabstraktionsebenen Low Level und High Level sowie die drei Modellvarianten Complex-YOLO, Kleines Complex-YOLO und Mini Complex-YOLO miteinander verglichen.

Abbildung 6.7 stellt die Multifaktorielle Performanz-Evaluation für die Modelle Complex-YOLO Low Level (Abbildung 6.7a) und Complex-YOLO High Level (Abbildung 6.7b) dar. Diese werden über alle Störungsarten und Schwierigkeitsgrade hinweg ermittelt, basierend auf den genannten Anteilen an integrierter Störung während des Trainings. Die Auswertung für die einzelnen Störungen ist in Anhang R dargestellt.



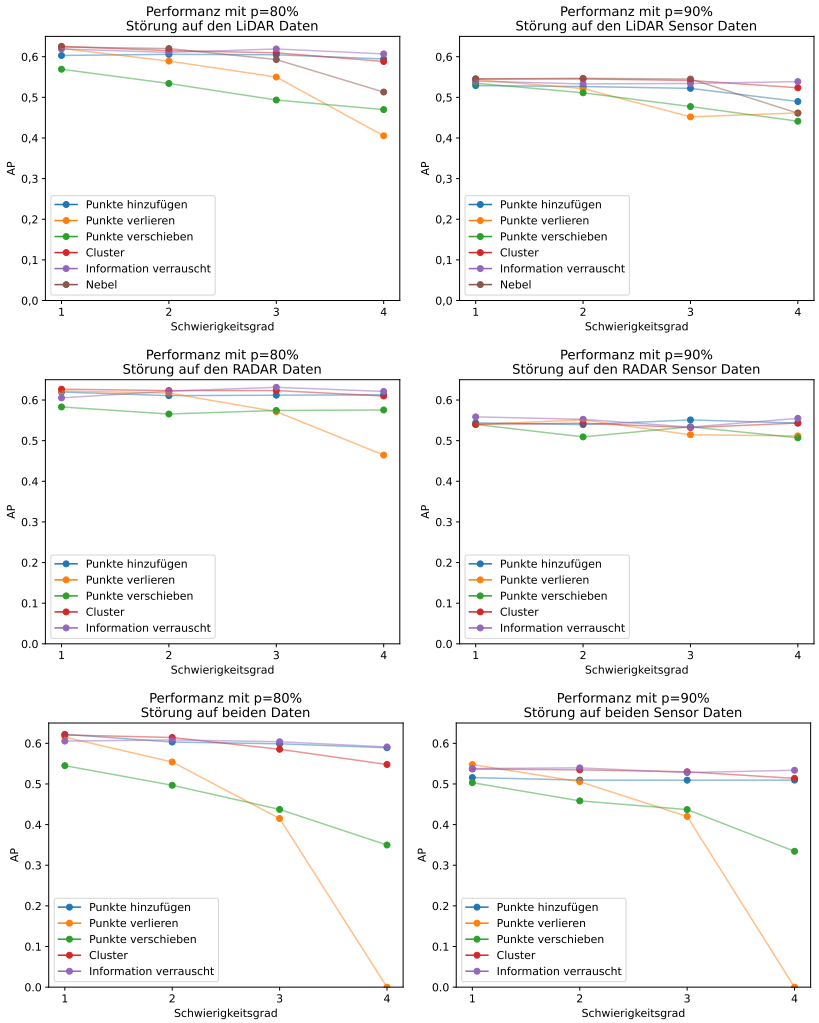
(a) Complex-YOLO Low Level

(b) Complex-YOLO High Level

Abbildung 6.7: Multifaktorielle Performanz-Evaluation von KISA mit mehreren Störungen für Complex-YOLO Low Level (a) und High Level (b) auf dem Astyx Datensatz. Es wird jeweils mit der relativen radialen Geschwindigkeit der RADAR Daten trainiert.

Mit zunehmendem Anteil an integrierten Störungen p verbessert sich die Average Precision für die Modelle Complex-YOLO Low Level und Complex-YOLO High Level.

Für das Complex-YOLO Low Level erreicht die Robustheit bei einem Störungsanteil von $p_{\max} = 80\%$ mit einer multifaktoriellen Performanz von $M_{p_{\max}}(\hat{S}) = 0{,}58$ ihr Maximum. Dies entspricht einer Steigerung von 93% gegenüber dem Modell mit ungestörten Trainingsdaten ($p_0 = 0\%$, mittlere multifaktorielle Performanz: $M_{p_0}(\hat{S}) = 0{,}30$). Eine Sättigung der Robustheit wird bereits bei $p_{\text{satt}} = 40\%$ mit einer multifaktoriellen Performanz von $M_{p_{\text{satt}}}(\hat{S}) = 0{,}55$ erreicht. Dies ist eine Verbesserung von 83% gegenüber dem mit ungestörten Daten



(a) Complex-YOLO Low Level **(b)** Complex-YOLO High Level

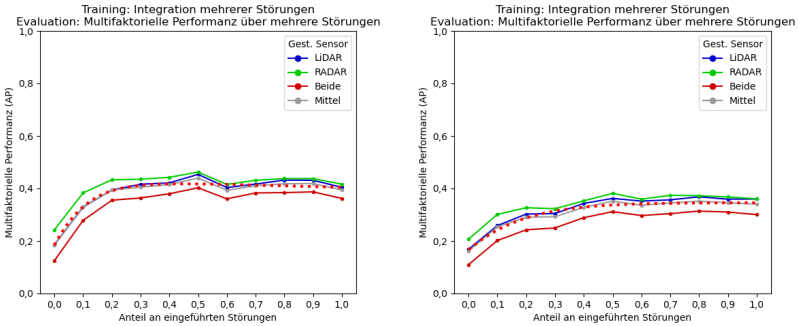
Abbildung 6.8: Zweite Sensitivitätsanalyse von Complex-YOLO Low Level (a) und Complex-YOLO High Level (b), jeweils des Modells, welches die höchste multifaktorielle Performanz hat (für Complex-YOLO Low Level p = 80% beziehungsweise Complex-YOLO High Level p = 90%).

trainierten Complex-YOLO Low Level Modell. Die größten Leistungssteigerungen werden bei gleichzeitiger Störung beider Sensoren mit einer Zunahme von 146% erzielt. Im Vergleich dazu führen Störungen auf nur einem Sensor zu einer Leistungssteigerung von 93% bei Störung der LiDAR-Sensordaten und 65% bei Störung der RADAR-Sensordaten, jeweils bei $p_{\text{max}} = 80\%$.

Das Complex-YOLO High Level Modell erreicht eine maximale multifaktorielle Performanz von $\boldsymbol{M}_{p_{\text{max}}}(\hat{S}) = 0{,}52$ bei einem Störungsanteil von $p_{\text{max}} = 90\%$. Das ist eine Leistungsverbesserung von 108% gegenüber dem Modell mit ungestörten Trainingsdaten ($p_0 = 0\%$, $\boldsymbol{M}_{p_0}(\hat{S}) = 0{,}25$). Das High Level Modell erreicht eine Sättigung bei $p_{\text{satt}} = 70\%$ mit einer multifaktoriellen Performanz von $\boldsymbol{M}_{p_{\text{satt}}}(\hat{S}) = 0{,}52$. Dies ist eine Verbesserung von 108% im Vergleich zum Complex-YOLO High Level Modell, welches ohne Störung $p_0 = 0\%$ trainiert wird. Auch hier sind die größten Steigerungen der Robustheit bei Störungen der beiden Sensordaten mit einer Steigerung von 191% zu verzeichnen. Bei Störungen der Daten des LiDAR-Sensors wird die Performanz um 125% gesteigert. Bei Störungen der RADAR Daten wird die Leistung um 53% gesteigert, jeweils bei $p_{\text{max}} = 90\%$. Die geringere Steigerung bei gestörten RADAR-Sensordaten hängt mit der bereits guten Leistung des Complex-YOLO Modells bei $p_0 = 0\%$ zusammen.

In Abbildung 6.8 ist die zweite Sensitivitätsanalyse von Complex-YOLO Low Level (Abbildung 6.8a) und Complex-YOLO High Level (Abbildung 6.8b) dargestellt. Jeweils wird das Fusionsmodell, welches nach der multifaktoriellen Performanz den höchsten Wert hat dargestellt. Dies sind die Modelle mit einem Anteil $p = 80\%$ für Complex-YOLO Low Level und einem Anteil $p = 90\%$ für Complex-YOLO High Level. Aus beiden zweiten Sensitivitätsanalysen zeigt sich, dass ”Punkte verlieren Grad 4” immer noch einen negativen Einfluss auf die Leistung beider Fusionsmodelle hat, da bei dieser Störung alle Daten von einem oder beiden Sensoren verloren gehen. Beispielsweise beträgt die Average Precision 0,40 für Complex-YOLO Low Level und 0,46 für Complex-YOLO High Level, wenn die Störung auf den LiDAR Daten auftritt. Dies ist vergleichbar mit dem Complex-YOLO Modell, das ohne Fusion nur auf RADAR Daten trainiert wurde (AP: 0,45). Auch die Störungen ”Punkte verschieben” und ”Nebel Grad 4” konnten nur bis zu einem gewissen Grad kompensiert werden, während die

restlichen Störungen fast vollständig ausgeglichen werden konnten. Insgesamt zeigt sich hier, dass mit KISA die Low Level Fusion robuster gegenüber den Störungen ist, als die High Level Fusion.



(a) Kleines Complex-YOLO Low Level **(b)** Mini Complex-YOLO Low Level

Abbildung 6.9: Ergebnisse der Verbesserung der Robustheit auf dem Astyx Datensatz. Mittlere Performanz hinsichtlich der Average Precision von Kleines Complex-YOLO Low Level (a) und Mini Complex-YOLO Low Level (b), das mit Störungen trainiert wird. Es wird jeweils mit der relativen radialen Geschwindigkeit der RADAR Daten trainiert.

Abbildung 6.9 stellt multifaktorielle Performanz von Kleines Complex-YOLO Low Level und Mini Complex-YOLO Low Level für verschiedene Anteile integrierter Störung während des Trainings dar. Tabelle 6.3 stellt die multifaktoriellen Performanz $M_p(\hat{S})$ der verschiedenen betrachteten Complex-YOLO Modelle, beim Training ohne Störung p_0, beim Anteil, an dem die Sättigung p_{satt} erreicht wird, und beim Anteil mit der maximalen multifaktoriellen Performanz p_{max}, dar. Die maximale multifaktorielle Performanz von Kleines Complex-YOLO Low Level wird bei $p_{\text{max}} = 50\%$ mit einem Wert von $M_{p_{\text{max}}}(\hat{S}) = 0{,}44$ erreicht. Die maximale multifaktorielle Performanz von Mini Complex-YOLO Low Level ist ebenfalls bei $p_{\text{max}} = 50\%$, aber mit einem geringeren Wert von $M_{p_{\text{max}}}(\hat{S}) = 0{,}35$. Prozentual gesehen ist dies ein größerer Anstieg im Vergleich zu der multifaktoriellen Performanz der Modelle mit $p_0 = 0\%$ (Complex-YOLO Low Level: 95%, Kleines Complex-YOLO Low Level: 139% und Mini Complex-YOLO Low Level: 117%). Dies liegt jedoch vor allem an der schlechten

multifaktoriellen Performanz der kleineren Modelle bei $p_0 = 0\%$ (Tabelle 6.3). Vergleicht man die erreichte maximale multifaktorielle Performanz $M_{p_{\max}}(\hat{S})$ mit der Average Precision auf den Originaldaten der Modelle, die ohne Störung ($p_0 = 0\%$) trainiert wurden, zeigt sich, dass die multifaktorielle Performanz bei Kleines Complex-YOLO Low Level um 2,3% niedriger ist als bei $p_0 = 0\%$ auf den Originaldaten. Bei Mini Complex-YOLO Low Level ist sie um 4,8% niedriger und beim ursprünglichen Complex-YOLO Low Level nur um 1,6%. Dies belegt, dass größere Modelle auf gestörten Daten näher an ihre Leistung auf ungestörten Daten herankommen. Dadurch sind sie besser in der Lage, auch bei Störungen der Sensordaten eine gute Leistung zu erbringen, wenn die Modelle mit der Störung trainiert wurden.

Tabelle 6.3: Vergleich der Mittelwerte der multifaktoriellen Performanz $M_p(\hat{S})$ von Complex-YOLO Low Level, Kleines Complex-YOLO Low Level, Mini Complex-YOLO Low Level und Complex-YOLO High Level, bei Training ohne Störung p_0, bei dem Anteil, an dem die Sättigung p_{satt} erreicht wird, und bei dem Anteil mit der maximalen multifaktoriellen Performanz $p_{\max}$.

Modell	$(p_0, M_{p_0}(\hat{S}))$	$(p_{\text{satt}}, M_{p_{\text{satt}}}(\hat{S}))$	$(p_{\max}, M_{p_{\max}}(\hat{S}))$
Complex-YOLO Low Level	(0%, 0,30)	(40%, 0,55)	(80%, 0,58)
Kleines Complex-YOLO Low Level	(0%, 0,18)	(40%, 0,41)	(50%, 0,44)
Mini Complex-YOLO Low Level	(0%, 0,16)	(50%, 0,35)	(50%, 0,35)
Complex-YOLO High Level	(0%, 0,25)	(70%, 0,52)	(90%, 0,52)

6.5 Bewertung

Die durchgeführte Analyse zeigt die Robustheit und Grenzen der verschiedenen Complex-YOLO-Modelle unter verschiedenen Bedingungen der Sensorfusion auf Punktwolken. Im Folgenden werden die wesentlichen Beobachtungen und deren Implikationen diskutiert.

Die ersten Sensitivitätsanalysen zeigen, dass die Störungen ”Punkte hinzufügen” und ”Punkte verschieben” für die Complex-YOLO Modelle besonders herausfordernd sind. Dies verdeutlicht die Sensitivität der Modelle gegenüber Störungen, die die Struktur der Punktwolken verändern.

Die Erweiterung von Low Level und High Level Fusion bei Complex-YOLO zeigt unterschiedliche Vorteile. Low Level Fusion verbessert die Detektionsleistung deutlich, insbesondere bei ungestörten Daten. High Level Fusion steigert die Robustheit der Modelle gegenüber Störungen, führt jedoch nicht zu einer Verbesserung der Leistung bei ungestörten Daten.

Die Anwendung des vorgestellten KISA auf die Complex-YOLO Modelle mit spezifischen Störungen zeigt in der Multifaktoriellen Performanz-Evaluation eine Verbesserung der Robustheit gegenüber diesen Störungen. Diese Verbesserungen deuten darauf hin, dass die Modelle durch gezieltes Training effektiv lernen können, mit Störungen umzugehen. Jedoch zeigt sich auch, dass ein zu hoher Anteil an gestörten Trainingsdaten die Leistung auf ungestörten oder anders gestörten Daten beeinträchtigen kann.

Die Integration mehrerer Störungsarten in KISA verbessert nach der Multifaktoriellen Performanz-Evaluation die Robustheit der Modelle. Die Ergebnisse zeigen, dass eine Diversifizierung der Trainingsdaten hinsichtlich der Störungen die Anpassungsfähigkeit und Robustheit der Modelle erhöht. Besonders hervorzuheben ist, dass durch KISA die Modelle gleichzeitig gegen verschiedene Störungen robust gemacht werden können, was in realen Anwendungen, in denen mehrere Störungsarten gleichzeitig auftreten können, von großer Bedeutung ist.

Der Vergleich zwischen den verschiedenen Modellvarianten (Complex-YOLO, Kleines Complex-YOLO und Mini Complex-YOLO) zeigt, dass größere Modelle eine höhere Robustheit auf gestörten Daten erreichen. Dies wird durch die Multifaktorielle Performanz-Evaluation nachgewiesen, die bei den größeren Modellen näher an die Leistung auf ungestörten Daten heranreicht. Größere Modelle sind besser in der Lage, komplexe Störungen zu verarbeiten und gleichzeitig eine hohe Leistung auf ungestörten Daten zu erzielen.

Basierend auf den Ergebnissen der durchgeführten Robustheitsanalyse wird empfohlen, für die Anwendungen von KISA, eine sorgfältige Auswahl der Trainingsdaten vorzunehmen. Wenn eine umfangreiche Analyse zur Bestimmung des optimalen Anteils nicht möglich ist, wird empfohlen, mit Anteilen von 10% bis 20% integrierter Störung zu trainieren. Dabei ist zu beachten, dass der Anteil an gestörten Daten erhöht werden sollte, wenn mehrere Störungsarten parallel in KISA berücksichtigt werden. Welche Störungen in KISA einbezogen werden, hängt von der Wahl der Sensoren und dem Anwendungsfeld ab. Mit der ersten Sensitivitätsanalyse ist ein Mittel gegeben, um die Wahl einzuschränken, indem überprüft wird, welche Störungen im System zu Leistungsabfall führen. Werden nach Anwendung KISA in der weiteren Sensitivitätsanalyse immer noch relevante Leistungsverluste festgestellt, so ist eine andere Fusionsstrategie oder ein anderes Modell mit größerer Lernfähigkeit zu wählen. Bei der Wahl der Fusionsstrategie ist die Low Level Fusion zu bevorzugen, wenn Trainingsdaten mit Störungen vorliegen oder Störungen simuliert werden können, da diese Fusion eine höhere Anpassungsfähigkeit gegenüber komplexen Störungen gezeigt hat. Wenn jedoch spezifische Störungen bekannt sind und gezielt adressiert werden sollen, ist die High Level Fusion vorzuziehen, da sie durch die separate Auswertung der Sensordaten eine integrierte Robustheit bietet.

7 Validierung der Wirksamkeit von KISA

7.1 Übersicht

In diesem Kapitel erfolgt die Darstellung der Übertragbarkeit des entwickelten KISA, indem die bisher in Kapitel 6 erzielten Ergebnisse auf weitere Datensätze und Methoden angewendet werden. Mit der Einbeziehung von Echtwelt Benchmark Datensätzen wie BraTS, A2D2, Delft oder KITTI, einem simulierten Datensatz und den Methoden U-Net und Sparse Fuse Dense, wird die modell- und modalitätsagnostische Einsetzbarkeit des Konzepts hervorgehoben. Diese Validierung testet nicht nur die Adaptivität des Konzepts unter veränderten datenspezifischen Bedingungen, sondern auch seine Effektivität in Kombination mit einer technologisch unterschiedlichen Methodik.

7.2 Anwendung von KISA für den Ausfall von Sensoren auf dem A2D2 Datensatz

Die Ergebnisse dieser Analyse wurden 2024 in IEEE Transactions on Intelligent Vehicles veröffentlicht [Bere24b].

Der Audi Autonomous Driving Dataset, bekannt als A2D2 [Geye20], konzentriert sich auf Kamera- und LiDAR-Daten, wobei fünf Velodyne Puck LiDAR-Sensoren zum Einsatz kommen. Diese LiDAR-Sensoren, die auf dem Dach des Fahrzeugs angebracht sind, haben eine bestimmte Neigung, wobei für diese Arbeit nur die drei vorderen LiDAR Sensoren verwendet werden. Die vorderen LiDAR-Sensoren sind an der linken (L) und rechten Ecke (R) bzw.

in der Mitte (M) des Fahrzeugs angebracht. Diese Auswahl basiert auf den Labels des Datensatzes, die in erster Linie Objekte erfasst, die sich vor dem Fahrzeug befinden. A2D2 enthält nicht nur Beschriftungen für Autos, sondern auch für Fußgänger und Radfahrer. Weitere Informationen zu dem Datensatz sind in Anhang L dargestellt.

Damit diese Analyse vergleichbar mit den vorherigen Analysen auf dem Astyx Datensatz ist, wird auch hier für den A2D2-Datensatz Complex-YOLO Low Level für die Objekt Detektion bei KISA verwendet. In KISA fließen 6 verschiedenen Kombinationen von LiDAR-Sensorausfällen ein, sowie der Fall, dass die Daten aller Sensoren verwendet werden. Diese Kombinationen von Ausfällen beinhalten, dass die Daten von einem oder zwei Sensoren fehlen. Von welchem Sensor die Daten verwendet werden, wird mit der Abkürzung für den Sensor (L, R, M) angegeben. LR bedeutet zum Beispiel, dass die Daten des linken und rechten Sensors verfügbar sind, während die Daten des mittleren LiDAR Sensors fehlen.

Tabelle 7.1: Evaluierungsergebnisse für Complex-YOLO Low Level, welche auf dem A2D2-Datensatz trainiert wird, unter Berücksichtigung verschiedener Prozentsätze von Sensorausfällen in den Trainingsdaten. Sieben Kombinationen von Punktwolken werden für jedes trainierte Modell bewertet. Die erste Spalte zeigt an, welche LiDAR-Sensoren bei den Evaluationsdaten vorhanden sind. Die zweite Spalte zeigt die AP für Complex-YOLO Low Level, das mit Daten von allen drei LiDAR-Sensor trainiert wird. Die dritte und vierte Spalte zeigen die AP für Complex-YOLO Low Level, bei denen 10% bzw. 20% der Trainingsdaten von einem oder zwei Sensoren fehlen. Die letzte Spalte zeigt als Vergleichswert die AP für Complex-YOLO Low Level, welches nur mit den Sensordaten der Auswertungsreihe trainiert wird.

	$p = 0\%$	$p = 10\%$	$p = 20\%$	Vergleichswert
LRM	0,76	0,76	0,81	0,76
LM	0,72	0,71	0,78	0,74
MR	0,72	0,72	0,78	0,78
LR	0,57	0,60	0,68	0,69
M	0,63	0,63	0,71	0,64
R	0,39	0,43	0,51	0,59
L	0,41	0,43	0,51	0,57

Tabelle 7.1 zeigt die Leistung von Complex-YOLO Low Level, trainiert auf dem A2D2-Datensatz mit verschiedenen Trainingseinstellungen. Es ist zu beobachten, dass die Genauigkeit des Modells bei $p = 0\%$ (keine künstlichen Sensorausfälle) am niedrigsten ist, wobei die Average Precision zwischen 0,39 und 0,76 liegt. Mit steigendem Prozentsatz der Sensorausfälle nehmen die Leistungswerte für alle sieben Kombinationen von Punktwolken zu. Durch die Einführung von $p = 20\%$ Sensorausfällen in die Trainingsdaten erreichen die Modelle eine Average Precision, welche zwischen 0,51 und 0,81 liegt. Auch wenn diese Leistung mit den Modellen vergleichen wird, die speziell mit nur einem Sensor trainiert werden, schneidet das Modellfusionsmodell mit $p = 20\%$ besser ab. Der Vergleichswert der Average Precision für ein Modell, das nur auf dem mittleren LiDAR-Sensor trainiert wird, beträgt nur 0,64, verglichen mit 0,71 für das Modell, das auf Sensorausfälle mit $p = 20\%$ trainiert wird.

7.3 Anwendung von KISA für die Störung Nebel auf dem "View of Delft"Datensatz

Die Ergebnisse dieser Analyse wurden 2024 in IEEE Transactions on Intelligent Vehicles veröffentlicht [Bere24b].

Der View of Delft Datensatz, auch bekannt als Delft Datensatz, wurde 2021 von Palffy et al. [Palf22] eingeführt. Dieser Datensatz zeichnet sich durch die Verwendung hochauflösender Sensoren aus, einschließlich eines Velodyne HDL-64 S3 LiDAR, Stereo-Kameras und eines ZF FRGen21 3+1D RADARs. Damit beinhaltet der Datensatz Daten von ähnlichen Sensoren, wie im Astyx Datensatz. Weitere Beschreibung des Datensatzes ist in Anhang M gegeben.

Auf dem View of Delft Datensatzes soll im speziellen die Störung durch Nebel untersucht werden, da diese bedeutsam für das autonome Fahren ist. Des weiteren wird hier dargelegt, dass auch bei gleicher Datenmodalität die Datenhomogenität bedeutsam ist für die Fusion.

In KISA geht die Störung ”Nebel Grad 4” mit den Anteilen p von 10% bis 100% in Schritten von 10% ein. Um die Vergleichbarkeit mit den Ergebnissen auf den homogenen Daten aus dem Astyx Datensatz zu haben, wird auch hier Complex-YOLO Low Level für die Objekt Detektion verwendet. Für das Multifaktorielle Performanz-Evaluation werden allen 4 Schwierigkeitsgrade der Störung ”Nebel” verwendet.

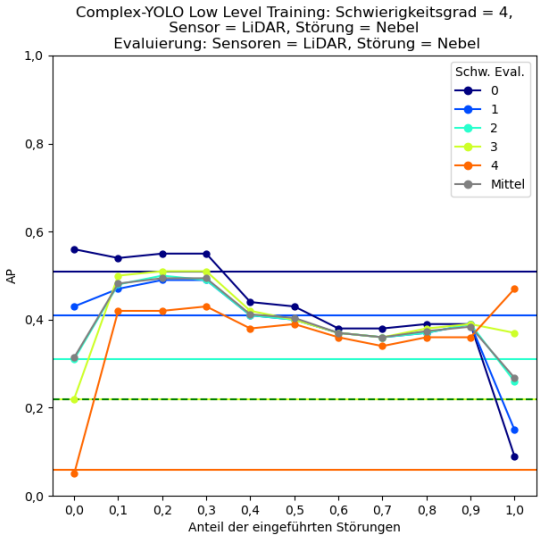


Abbildung 7.1: Multifaktorielle Performanz-Evaluation von KISA mit Complex-YOLO Low Level und mit ”Nebel Grad 4” auf dem ”View of Delft” Datensatz. In Anlehnung an eine Graphik aus [Bere24b].

Abbildung 7.1 zeigt die Multifaktorielle Performanz-Evaluation. Mit einem Teil $p = 30\%$ der Trainingsdaten, die auf ”Nebel Grad 4” gestört wurden, wird die robusteste Leistung erreicht. Die durchschnittliche Genauigkeit der Daten ohne die Störung und mit den verschiedenen Nebel Stufen verbessert sich von 0,31 ($p = 0\%$) auf 0,49 ($p = 30\%$), was einer Steigerung von 58% entspricht.

Wird jedoch der Anteil der durch "Nebel Grad 4" gestörten Trainingsdaten auf $p = 40\%$ oder mehr erhöht, spezialisiert sich das Modell auf die charakteristischen Eigenschaften von "Nebel Grad 4". Diese Überanpassung verringert die Fähigkeit des Modells, sich an geringere Nebelstärken oder nebelfreie Bedingungen anzupassen, was zu einer allgemeinen Leistungsverschlechterung führt. Während die Auswertung relativ starken Nebels (> Grad 2) immer noch eine deutliche Leistungssteigerung im Vergleich zu dem Modell zeigt, das ausschließlich auf den Originaldaten ($p = 0\%$) trainiert wurde, zeigt die Auswertung bei leichterem Nebel, dass die Leistung sogar unter das Modell $p = 0\%$ fällt. Wird nur mit Daten, auf denen "Nebel Grad 4" angewendet, trainiert, so kehrt sich die Reihenfolge der Leistungen in den verschiedenen Schwierigkeitsgraden um. Dies hängt damit zusammen, dass sich das Complex-YOLO Modell nur auf diese Störung eingestellt hat und die Daten mit "Nebel Grad 3" denen aus den Trainingsdaten noch ähneln, während die auf denen leichter Nebel vom Grad 2 dies nicht mehr tun. Die Sättigung wird bei $p_{\text{satt}} = 20\%$ erreicht, mit einer Performanz von 0,49.

Das Fusionsmodell, das nur auf den Originaldaten trainiert wird, hat eine ähnliche Leistung wie das Nicht-Fusionsmodell auf den LiDAR-Daten und verbessert sich nur um 10% vom Training ausschließlich auf LiDAR-Daten zum Training auf fusionierten LiDAR-RADAR-Daten mit $p = 0\%$. Dies ist auf die Tatsache zurückzuführen, dass das Velodyne 64 in der Ansicht von Delft viel mehr Daten als das RADAR hat, so dass die zusätzlichen Informationen des RADARs keinen großen Einfluss auf die Leistung haben.

7.4 Anwendung von KISA bei Sparse Fuse Dense

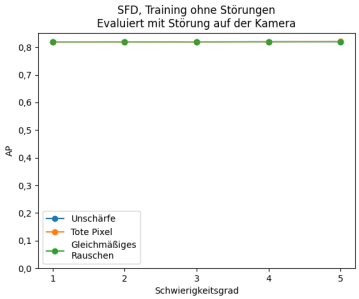
Bisher fokussierte sich die Anwendung des vorgestellten Konzepts auf die Anwendung bekannter Methoden wie Complex-YOLO auf verschiedenen Datensätzen wie A2D2 und Delft. Diese Ansätze evaluierten die Anwendbarkeit und Robustheit der Algorithmen unter Verwendung unterschiedlicher Daten bei gleichbleibender methodischer Grundlage. Im Gegensatz dazu wird

in diesem Abschnitt KISA erstmals auf dem KITTI-Datensatz angewendet, kombiniert mit einer völlig neuen Objektdetektionsmethode: Sparse Fused Dense (SFD) [Wu22]. Diese Methode, die Bilder und LiDAR-Daten fusioniert, unterscheidet sich grundlegend von den bisherigen Ansätzen. Der Fokus auf ein neues Netz und dessen Anwendung auf heterogenen Datenmodalitäten stellt eine komplett andere Art von Validierung dar, die neue Einsichten in die Flexibilität des vorgestellten Konzepts ermöglicht.

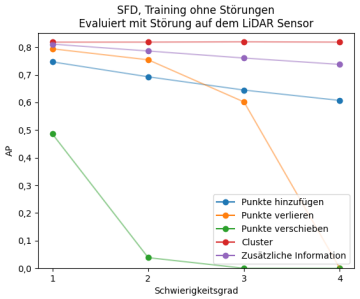
SFD ermöglicht die Fusion von spärlichen mit dichten Datenquellen. Im Gegensatz zu Complex-YOLO, das primär die Geschwindigkeit der Objektdetektion in 3D-Umgebungen durch die direkte Verarbeitung von Punktwolkendaten optimiert, verwendet Sparse Fuse Dense eine tiefer gehende Fusionsstrategie, um die Qualität der Objektdetektion zu erhöhen. SFD beginnt mit einer getrennten Vorverarbeitung der Datenströme von LiDAR-Sensor und RGB-Kamera. Die Kameradaten werden zusammen mit der LiDAR-Punktwolke zur Tiefenvervollständigung verwendet. SFD verwendet für die Tiefenvervollständigung die Methode Twin-Surface Extrapolation (TWISe) [Imra21]. Für die Fusion der Kamera- und LiDAR-Merkmale verwendet SFD Techniken wie 3D Grid-wise Attentive Fusion (3D-GAF). Damit stellt SFD einen Mid-Fusion-Algorithmus dar. Weitere Details zu SFD finden sich im Anhang N.

In KISA gehen diese Störungen $\mathbf{S}$ = {”Punkte verlieren Grad 1”, …, ”Punkte verlieren Grad 4”, ”Punkte hinzufügen Grad 1”, …, ”Punkte hinzufügen Grad 4”, ”Punkte verschieben Grad 1”, …, ”Punkte verschieben Grad 4”, ”Information verrauscht Grad 1”, …, ”Information verrauscht Grad 4”, ”Cluster Grad 1”, …, ”Cluster Grad 4”, ”Nebel Grad 1”, …, ”Nebel Grad 4”, ”Schnee”} ∪ {”Unschärfe Grad 1”, …, ”Unschärfe Grad 5”, ”Gleichmäßiges Rauschen Grad 1”,…, ”Gleichmäßiges Rauschen Grad 5”, ”Tote Pixel Grad 1”, …, ”Tote Pixel Grad 5”} auf den Punktwolkedaten beziehungsweise Bilddaten mit den Anteilen p von 10% bis 100% in Schritten von 10% ein. Die verwendeten Bildstörungen werden in Anhang F.3 beschrieben.

Die Auswertung im Evaluationskonzept beschränkt sich dabei auf die nach dem KITTI Datensatz klassifizierte Gruppe der Objekte ’hard’. Dabei wurde die Sichtbarkeit eines Objekts nach Objektivenmaßstäben, wie Abstand oder Verdeckung, eingeordnet.



(a) Sensitivitätsanalyse für Bildstörungen



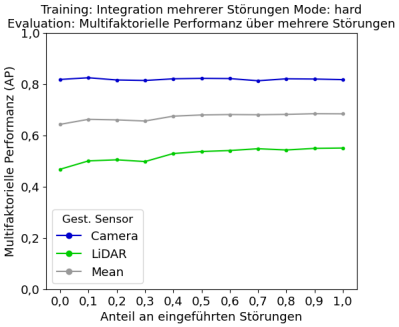
(b) Sensitivitätsanalyse für Punktwolkestörungen

Abbildung 7.2: Erste Sensitivitätsanalyse von SFD auf dem KITTI Datensatz. Die Ergebnisse beziehen sich auf Objekte der Klasse 'Car', welche der Schwierigkeit 'hard' laut KITTI eingeordnet werden. Average Precision von SFD auf klaren Daten ohne zusätzliche Störung 0,82.

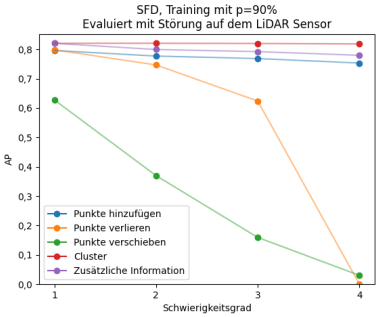
Durch die erste Sensitivitätsanalyse in Abbildung 7.2 ist erkenntlich, dass eine Störung der Bilddaten, oder die Punktwolkenstörung "Cluster" keinen Einfluss auf die Performanz von SFD hat. So ist beispielsweise die Average Precision von SFD auf "Cluster Grad 4" 0,82. Dies liegt im Besonderen an der guten Leistung von TWISe mit gestörten Bilddaten. Die erzeugte dichte Tiefenkarte ist der mit ungestörten Daten erzeugten sehr ähnlich. Die Sensitivitätsanalyse zeigt auch, dass die Störung "Punkte verschieben" einen besonders starken Einfluss auf die Leistung hat, so ist die Average Precision bei Daten mit "Punkte verschieben Grad 1" 0,49 und mit "Punkte verschieben Grad 4" 0,0. Die multifaktorielle Performanz beträgt 0,64 für SFD, welches nur auf den Originaldaten trainiert wird p=0%. Werden nur die Punktwolkestörungen betrachtet, liegt der Wert bei 0,47. Dies liegt daran, dass die Bildstörungen kaum einen Einfluss auf die Leistung haben.

Die Multifaktorielle Performanz-Evaluation in Abbildung 7.3a zeigt, dass mit steigendem Anteil an integrierter Störung in den Trainingsdaten, die multifaktorielle Performanz zunimmt. Das Maximum wird bei p_{max} = 90% erreicht, mit einer multifaktoriellen Performanz von 0,69 (Steigerung um 6,4%), wenn die multifaktorielle Performanz über alle Störungen berechnet wird, und 0,55 (Steigerung um 17,3%), wenn die multifaktorielle Performanz nur über die

Punktwolkestörung berechnet wird. In Abbildung 7.3b ist die Sensitivitätsanalyse von SFD, welches mit dem Anteil von $p = 90\%$ an integrierten Störungen trainiert wurde dargestellt. Es zeigt sich, dass besonders die Robustheit gegenüber den Störungen "Punkte hinzufügen" gesteigert wird. So steigert sich beispielsweise die Average Precision bei "Punkte hinzufügen Grad 3" um 19%. Auch die Average Precision bei der Störung "Punkte verschieben Grad 4" bleibt mit einer Average Precision von 0,03 sehr gering. Dies zeigt, dass die Methode SFD eine sehr große Schwäche gegenüber dieser Störung hat, was an der schlechten Leistung bei der Erstellung der dichten Tiefenkarte liegt.



(a) Multifaktorielle Performanz über verschiedene Anteile p an integrierten Störungen



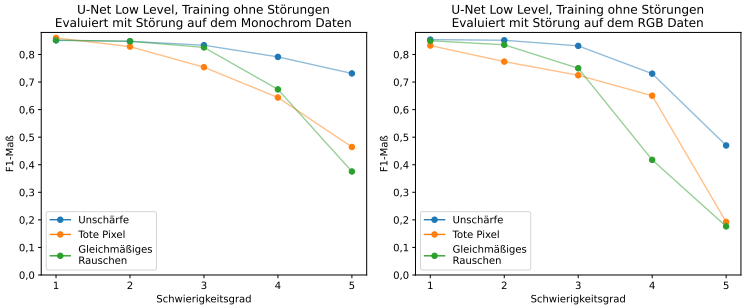
(b) 2. Sensitivitätsanalyse von SFD, welches mit einem Anteil von $p = 90\%$ an integrierten Störungen trainiert wurde

Abbildung 7.3: Multifaktoriellen Performanz-Evaluation 7.3a und zweite Sensitivitätsanalyse 7.3b von KISA mit SFD auf dem KITTI Datensatz. Die Ergebnisse beziehen sich auf Objekte der Klasse 'Car', welche der Schwierigkeit 'hard' laut KITTI eingeordnet werden.

7.5 Anwendung von KISA bei semantischer Segmentierung

In diesem Abschnitt werden die für diese Arbeit gewonnenen Ergebnisse des Konzeptes zur Robustheitssteigerung, bestehend aus KISA und den neuen Evaluationsmethodiken Sensitivitätsanalyse und Mutlifaktoriellen Performanz-Evaluation angewendet auf simulierten Daten für die semantische Segmentierung dargestellt. Der simulierte Datensatz wird in Abschnitt 3.3 beschrieben. Als Netzart zur Auswertung wird das U-Net-Modell verwendet, welches durch die High und Low Level Fusionsstrategie erweitert wird (siehe Abschnitt 4.5).

Es werden drei verschiedene Störungsarten mit fünf Schwierigkeitsgraden verwendet: ”Tote Pixel”, welche einzelne Bildpunkte auf den Wert 0 setzt, ”Unschärfe”, welche einen Boxfilter auf das Bild anwendet und ”Gleichmäßiges Rauschen”, welche den Wert einzelner Bildpunkte verändert. Die verwendeten Störungen werden in Anhang F.3 beschrieben.

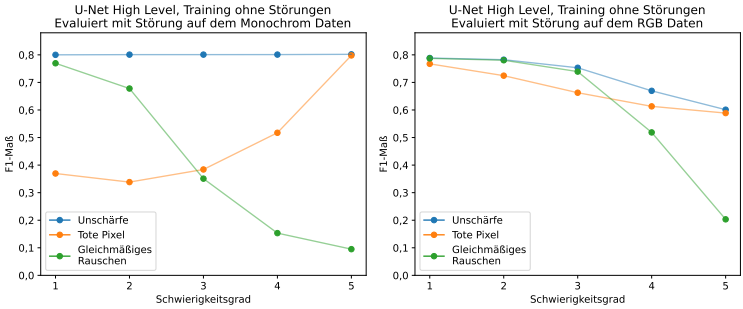


(a) Störung Monochrom Daten **(b)** Störung RGB Daten

Abbildung 7.4: Erste Sensitivitätsanalyse von U-Net Low Level.

Die erste Sensitivitätsanalyse der auf Fusion basierenden U-Net-Netzwerke ist in den Abbildungen 7.4 (Low Level) und 7.5 (High Level) dargestellt. Durch die High Level Fusion wird das F1-Maß bei Evaluationsdaten ohne zusätzliche Störung nicht verbessert. Die Leistung von U-Net High Level bleibt mit 0,80 auf demselben Niveau wie U-Net ohne Fusion und nur auf RGB Daten trainiert.

Die Low Level Fusion hingegen verbessert das Segmentierungsergebnis auf ein F1-Maß von 0,85 bei Evaluationsdaten ohne zusätzliche Störung.



(a) Störung Monochrom Daten **(b)** Störung RGB Daten

Abbildung 7.5: Erste Sensitivitätsanalyse von U-Net High Level.

Die Multifaktorielle Performanz-Evaluation von U-Net Low Level (Abbildung 7.4) ist am höchsten, wenn Störungen ausschließlich auf den Monochrom Evaluationsdaten angewendet werden, mit $M_{p_0} = 0{,}75$. Bei Anwendung der Störungen auf RGB Evaluationsdaten zeigt das U-Net Low Level System eine reduzierte multifaktorielle Performanz mit $M_{p_0} = 0{,}67$. Dieser Rückgang weist auf eine unterschiedliche Empfindlichkeit von U-Net Low Level gegenüber Störungen in den verschiedenen Sensormodalitäten hin. Die weitere Reduktion der multifaktorielle Performanz auf $M_{p_0} = 0{,}56$, wenn Störungen gleichzeitig auf beide Modalitäten angewendet werden, betont die Herausforderung, die kombinierte Störungen für U-Net Low Level darstellen.

Bei U-Net High Level (Abbildung 7.5) zeigt sich bei der Anwendung der Störungen auf den Monochrom Evaluationsdaten eine verringerte multifaktorielle Performanz $M_{p_0} = 0{,}58$ im Vergleich zur Anwendung der Störung auf RGB Evaluationsdaten $M_{p_0} = 0{,}67$. Dies liegt daran, dass U-Net Monochrom stärker von den Störungen beeinträchtigt wird als U-Net RGB. Da bei der High Level Fusion die Segmentierungskarten fusioniert werden, wirkt sich die geringere Leistung von U-Net Monochrom im Gegensatz zu U-Net Low Level direkt auf U-Net High Level aus. Bemerkenswerterweise steigt bei der Störung ”Tote

Pixel" auf den Monochrom Evaluationsdaten das F1-Maß von 0,37 bei Grad 1 auf 0,80 bei Grad 5. Der Grund dafür ist, dass bei Grad 5 alle Bildwerte auf 0 gesetzt werden, wodurch die Segmentierungskarte von U-Net Monochrom jedem Bildpunkt die Klasse Hintergrund zuweist. Da für die High Level Fusion der Mittelwert der Segmentierungskarten verwendet wird, beeinflusst das Ergebnis von U-Net Monochrom die High Level Segmentierung nicht. Andererseits erweist sich bei der High Level Fusion insbesondere "Gleichmäßiges Rauschen" als problematisch. Im Gegensatz zu "Tote Pixel" bei Grad 5, sind in der Regel bei "Gleichmäßiges Rauschen" die Bildwerte ungleich 0 sind, sodass sowohl U-Net Monochrom als auch U-Net RGB Bildpunkte fälschlicherweise der Klasse Ellipse zuordnen, obwohl keine Ellipse vorhanden ist.

Aus den Ergebnissen der ersten Sensitivitätsanalyse erwiesen sich die Störungen "Tote Pixel" und "Gleichmäßiges Rauschen" als besonders herausfordernd für das U-Net Low Level Modell. Daher wird hier beispielhaft die Anpassung durch KISA an diese Störungen betrachtet. Dafür wird das U-Net Low Level Modell mit den Anteilen p von 10% bis 100% in Schritten von 10% und den $\mathbf{S} = \{$"Tote Pixel Grad 3"$\}$ bzw. $\mathbf{S} = \{$"Gleichmäßiges Rauschen Grad 3"$\}$ trainiert.

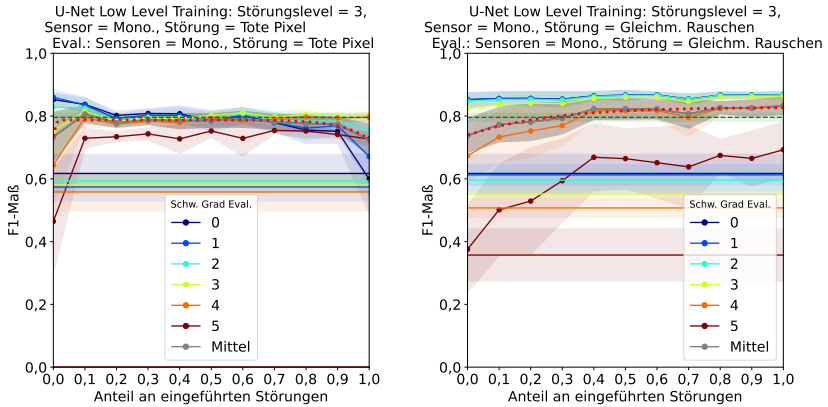


Abbildung 7.6: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung auf den Monochrom Daten. Links: "Tote Pixel Grad 3". Rechts: "Gleichmäßiges Rauschen Grad 3". Notation wie zu Abbildung 2.7 ausführlich beschrieben.

Aus den Ergebnissen der Evaluationsmethodik Multifaktorielle Performanz-Evaluation, in den Abbildungen 7.6 bis 7.8, lässt sich erkennen, dass das Training mit ”Tote Pixel Grad 3” bzw. ”Gleichmäßiges Rauschen Grad 3” die Leistung nicht nur für das jeweilige Schwierigkeitsgrad, auf das in KISA abgezielt wird, verbessert, sondern auch Verbesserungen bei höheren Graden zeigt, sowohl bei Störungen auf den Monochrom Evaluationsdaten (Abbildung 7.6) als auch bei Störungen auf beiden Evaluationsdaten (Abbildung 7.8). Diese Verbesserung wird mit steigendem Anteil p an Trainingsdaten mit Störung in KISA verstärkt. So verbessert sich die Performanz auf den Monochrom Evaluationsdaten mit der Störung ”Tote Pixel Grad 3” um 8,1%, wenn mit p =10% Trainingsdaten ”Tote Pixel Grad 3” trainiert wird (siehe Abbildung 7.6 links). Die Performanz auf den RGB Evaluationsdaten mit der Störung ”Tote Pixel Grad 3” steigt um 14,8%, wenn mit $p = 90\%$ Trainingsdaten ”Tote Pixel Grad 3” trainiert wird (siehe Abbildung 7.7). Die weiteren Schwierigkeitsgrad konnten nicht verbessert werden, wenn mit ”Tote Pixel Grad 3” auf den RGB Trainingsdaten trainiert wird.

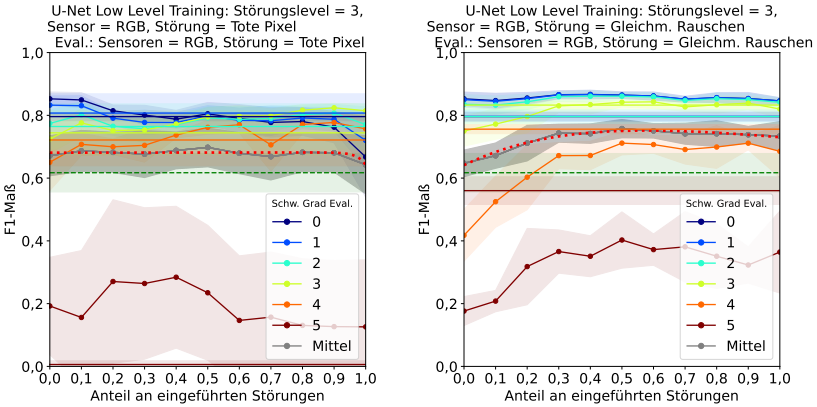


Abbildung 7.7: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung auf den RGB Daten. Links: ”Tote Pixel Grad 3”. Rechts: ”Gleichmäßiges Rauschen Grad 3”. Notation wie zu Abbildung 2.7 ausführlich beschrieben.

In allen drei Fällen, in denen ”Gleichmäßiges Rauschen Grad 3” entweder auf den Monochrom oder den RGB Daten oder auf den Daten beider Sensoren liegt, steigt die Leistung nicht nur bei den Evaluationsdaten mit dem Schwierigkeitsgrad, der auf die Trainingsdaten angewendet wird, sondern auch bei den Evaluationsdaten mit einem höheren Schwierigkeitsgrad. So steigt beispielsweise das F1-Maß bei Störung der RGB Evaluationsaten mit ”Gleichmäßiges Rauschen Grad 3” mit $p = 30\%$ um 11% bei Evaluation auf die Störung ”Gleichmäßiges Rauschen Grad 3” und um 61% bei ”Gleichmäßiges Rauschen Grad 4” (siehe Abbildung 7.7).

Wie aus den Abbildungen in 7.6- 7.8 ersichtlich, ist die Performanzsteigerung, wenn mit der sehr starken Störung ”Tote Pixel Grad 5” trainiert wird, besonders deutlich. So steigert sich das F1-Maß, wenn ”Tote Pixel Grad 5” auf den RGB Evaluationsdaten vorliegt, um 239%, beim Training mit $p = 100\%$ von ”Tote Pixel Grad 5” auf den RGB Trainingsdaten, verglichen mit dem Fall, dass ohne Störung $p = 0\%$ trainiert wird. Hingegen fällt bei einem Anteil von $p = 100\%$ an Trainingsdaten mit ”Tote Pixel Grad 5” die Leistung, mit Ausnahme der Bewertung der spezifischen Störung ”Tote Pixel Grad 5”, auf der das Netz trainiert wird, auf den anderen Schwierigkeitsgraden und ungestörten Evaluationsdaten stark ab.

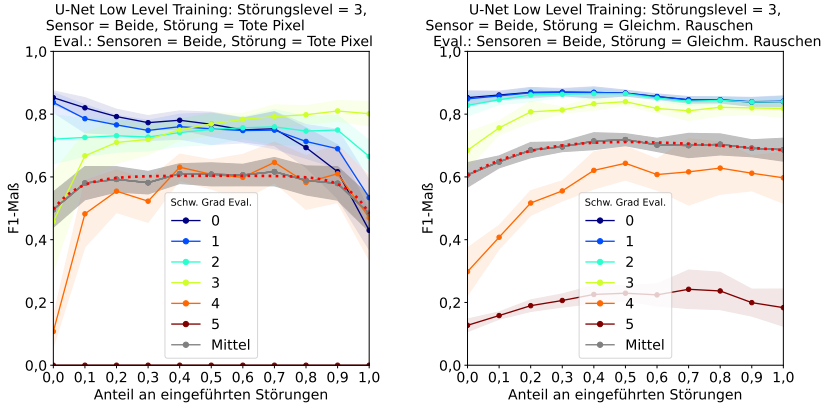
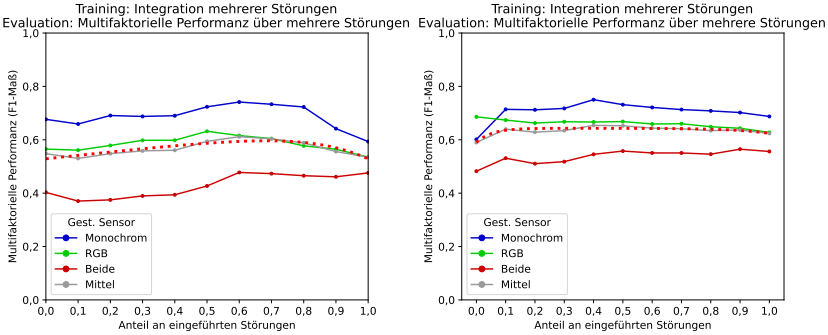


Abbildung 7.8: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung auf beiden Daten. Links: ”Tote Pixel Grad 3”. Rechts: ”Gleichmäßiges Rauschen Grad 3”. Notation wie zu Abbildung 2.7 ausführlich beschrieben.

Vergleicht man die U-Net Modelle, die mit fusionierten Daten arbeiten, mit denen, die ohne fusionierte Daten arbeiten, so zeigt sich, dass durch die Fusion die Störungen vollständig kompensiert werden können, wenn mit diesen trainiert wird. So hat U-Net Low Level, das mit $p = 90\%$ "Tote Pixel Grad 3" auf den RGB Daten trainiert wird, ein mittleres F1-Maß von 0,82 (siehe Abbildung 7.7), während ein Netz, das nur mit den ungestörten Monochrom Daten arbeitet, ein mittleres F1-Maß von nur 0,62 hat.

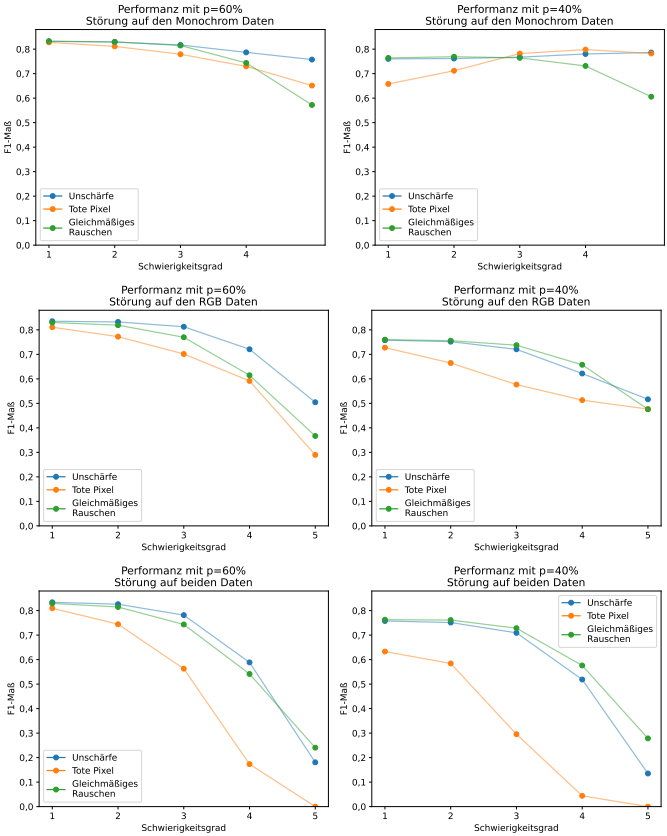
Inwiefern KISA mit verschiedenen Störungsarten die Robustheit der U-Net High und Low Level Modelle steigert, ist in der Multifaktoriellen Performanz-Evaluation in Abbildung 7.9 dargestellt. Für die einzelnen Störungen ist dies im Anhang F.5 dargestellt. Konkret werden folgende Störungen berücksichtigt $\mathbf{S}$ = {"Unschärfe Grad 1", ..., "Unschärfe Grad 5", "Gleichmäßiges Rauschen Grad 1",..., "Gleichmäßiges Rauschen Grad 5", "Tote Pixel Grad 1", ..., "Tote Pixel Grad 5"} und es werden die Anteile $p \in \{0\%, 10\%, 20\%, 30\%, 40\%, 50\%, 60\%, 70\%, 80\%, 90\%, 100\%\}$ betrachtet.



(a) U-Net Low Level **(b)** U-Net High Level

Abbildung 7.9: Multifaktorielle Performanz-Evaluation hinsichtlich des F1-Maßes von KISA mit U-Net Low Level (a) und mit U-Net High Level (b), und mit den Störungen $\mathbf{S}$ auf dem synthetischem Datensatz.

Für das U-Net Low Level erreicht die Robustheit des Modells bei einem Störungsanteil von $p_{\max} = 60\%$ mit einer multifaktoriellen Performanz von



(a) U-Net Low Level **(b)** U-Net High Level

Abbildung 7.10: Zweite Sensitivitätsanalyse von U-Net Low Level (a) und U-Net High Level (b), jeweils des Modells mit der höchsten multifaktoriellen Performanz (für U-Net Low Level $p = 60\%$ bzw. U-Net High Level $p = 40\%$).

$M_{p_{\max}}(\hat{S}) = 0{,}71$ ihren Höhepunkt, was einer Steigerung von 3,2% gegenüber dem Modell mit ungestörten Trainingsdaten ($p_0 = 0\%$, multifaktorielle Performanz: $M_{p_0}(\hat{S}) = 0{,}68$) entspricht. Die größten Leistungssteigerungen zeigen sich bei gleichzeitiger Störung beider Sensoren, mit einer Zunahme von 5,1%, während Störungen auf nur einem Sensor eine Leistungssteigerung von

2,7% bei den Monochrom Daten und 2,1% bei RGB Daten erreichen, jeweils bei $p_{\text{max}} = 60\%$.

Das U-Net High Level Modell erreicht eine maximale multifaktorielle Performanz von $M_{p_{\text{max}}}(\hat{S}) = 0{,}65$ bei einem Störungsanteil von $p_{\text{max}} = 40\%$. Diese Verbesserung entspricht einer erheblichen Leistungsverbesserung von 10,8% gegenüber dem Modell mit ungestörten Trainingsdaten ($p_0 = 0\%$, $M_{p_0}(\hat{S}) = 0{,}59$). Hier wird die größte Steigerung der Robustheit bei Störung der Monochrom Daten verzeichnet mit einem Anstieg um 25%, wohingegen bei Störungen der Daten beider Sensoren die Leistung um 13% gesteigert wird. Bei Störungen der RGB Daten wird die Leistung um 2,8% verringert, jeweils bei $p_{\text{max}} = 40\%$.

In Abbildung 7.10 ist die zweite Sensitivitätsanalyse von U-Net Low Level (Abbildung 7.10a) und U-Net High Level (Abbildung 7.10b), jeweils von dem Modell, welches die höchste multifaktorielle Performanz hat, somit $p = 60\%$ für U-Net Low Level und $p = 90\%$ für U-Net High Level, dargestellt. Daraus lässt sich erkennen, dass die Störung "Tote Pixel" weiterhin einen negativen Einfluss auf die Leistung der beiden U-Net Fusionsmodelle hat und auch durch Integration der Störung in das Training die U-Net Fusionsmodelle wenig robust gegen die hohen Schwierigkeitsgrade von "Tote Pixel" macht. Des Weiteren zeigt sich, dass die U-Net Fusionsmodelle robuster sind, wenn die Störung nur auf den Monochrom Daten vorliegt. Beispielsweise stellt "Gleichmäßiges Rauschen Grad 5" ein großes Problem für U-Net High Level dar, wenn es ohne Störung trainiert wird. Im Vergleich dazu ist U-Net High Level mit $p = 90\%$ robuster gegen diese Störung.

7.6 Anwendung von KISA bei MRT Daten

Um zu zeigen, dass KISA auch in Anwendungsbereichen außerhalb des autonomen Fahrens und mit weiteren Sensormodalitäten funktionieren kann, wird das Konzept auf MRT-Daten des BraTS-Datensatzes angewendet. Es wird das Modell U-Net zur semantischen Segmentierung verwendet, da U-Net bekannt

ist und damit die Erkenntnisse sich auf weitere Anwendungen übertragen werden können.

Die vier MRT-modalitäten werden zu einem vier Kanalbild kombiniert, was somit eine Low Level Fusion darstellt. Es werden dieselben Bildstörungen betrachtet, wie bei der Auswertung der simulierten Daten: ”Tote Pixel”, ”Unschärfe” und ”Gleichmäßiges Rauschen”. Die verwendeten Störungen werden in Anhang F.3 genauer beschrieben.

Die erste Sensitivitätsanalyse zeigt, dass alle drei Störungen die Leistung des Modells beeinflussen. Deshalb sind in KISA alle Störungsarten zu betrachten und es werden die Anteile p von 0% bis 100% in 10%-Schritten an integrierter Störung genommen.

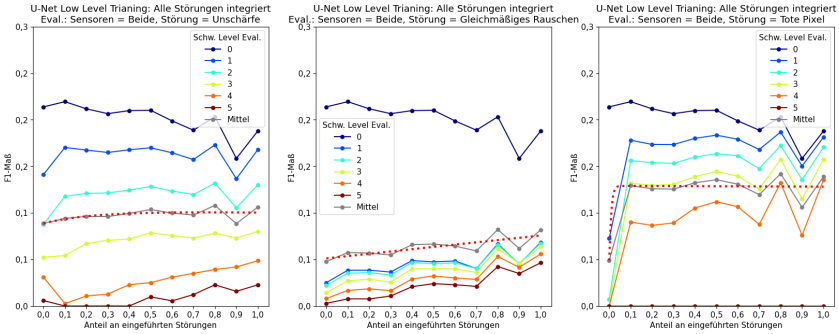


Abbildung 7.11: Multifaktoriellen Performanz-Evaluation in Bezug auf das F1-Maß von U-Net Low Level, das mit zufälligen Störungstypen und -stufen auf dem BraTS-Datensatz trainiert wurde. Das linke Diagramm zeigt die Ergebnisse für das U-Net-Modell, das auf der Störung ”Unschärfe” bewertet wurde. Das mittlere Diagramm zeigt die Ergebnisse für das U-Net-Modell, das für die Störung ”Gleichmäßiges Rauschen” evaluiert wurde. Das rechte Diagramm zeigt die Ergebnisse für das U-Netz-Modell, das für die Störung ”Tote Pixel” ausgewertet wurde.

Die Multifaktoriellen Performanz-Evaluation in Abbildung 7.11 liefert Daten zur Leistung von U-Net Low Level unter den Störungsarten. Für die Störung ”Tote Pixel” zeigt das Modell eine effektive Verarbeitung fehlender Informationen. Bei einer Störung von 10% der Trainingsdaten erhöht sich die Leistung um 203% in Szenarien mit Störung ”Tote Pixel Grad 3”, bei der 30% der Bildpunkte

auf Null gesetzt werden. Die Leistung bleibt unter Bedingungen ohne Störung auf den Bewertungsdaten weitgehend konstant. Bei den Störungen wie ”Unschärfe” und ”Gleichmäßiges Rauschen” wird die Robustheit gering verbessert. Zum Beispiel verbessert sich die Leistung bei ”Gleichmäßigem Rauschen Grad 2” um 60%, wenn 10% der Sensordaten gestört sind, bleibt jedoch mit einem F1-Maß von 0,04 relativ niedrig.

Die beste Leistung über alle Anteile hinweg wird mit einem Anteil von $p = 80\%$ erreicht, was zu einer signifikanten Verbesserung von 13,3% im gemittelten F1-Maß führt. Die Leistungssteigerung ist in erster Linie auf die Verbesserung im Umgang mit der Störung ”Tote Pixel” zurückzuführen.

7.7 Bewertung der Validierungsergebnisse

Die Ergebnisse aus diesem Validierungskapitel liefern wertvolle Einblicke in die Dynamik und die Effekte der Einbeziehung von Störungen in KISA von Objektdetektionsmodellen. Die systematische Analyse über verschiedene Datensätze hinweg zeigt, wie unterschiedliche Arten von Störungen die Leistungsfähigkeit der Modelle beeinflussen und in welchem Maße KISA, welches vorschlägt gestörte Daten während des Trainings zu integrieren, die Robustheit der Modelle erhöhen kann. Die durchgeführten Anwendungen validieren, dass die Integration von Störungen nicht nur die Robustheit auf den in Kapitel 6 untersuchtem Datensatz und Algorithmus verbessert, sondern auch erfolgreich auf verschiedene weitere Sensormodalitäten und mit U-Net und SFD auf weitere Algorithmen angewendet werden kann.

8 Zusammenfassung und Ausblick

In dieser Forschungsarbeit wird ein Konzept vorgestellt, wie in der Entwurfsphase eines Multisensorsystems vorgegangen werden muss. Der Schwerpunkt des Konzepts liegt dabei auf der Robustheit des Systems gegenüber Sensorstörungen. Das Konzept ist in mehrere Schritte unterteilt, wobei sich die Arbeit auf die Schritte Sensorplatzierung und Robustheit der Fusionsmethode gegenüber Störungen konzentriert.

Im Folgenden wird zusammengefasst, welche neuen Ansätze und Entwicklungen in dieser Arbeit erarbeitet wurden:

1 Ein umfassendes Konzept zur Entwicklung von Multisensorsystemen wurde erstellt, das gezielt die Auswirkungen von Sensorstörungen adressiert.

2 Einführung neuer Methoden GAOS und DLOS zur effizienten Optimierung der Sensorplatzierung.

3 Erarbeitung des Trainingskonzepts KISA zur systematischen Erhöhung der Robustheit von Fusionsmethoden.

4 Entwicklung eines neuen Evaluationskonzepts zur Robustheitsbewertung von Sensorfusionsmethoden, das die Sensitivitätsanalyse, die Multifaktorielle Performanz-Evaluierung und die Modellierung der Robustheitsdynamik umfasst, um die Widerstandsfähigkeit von Fusionsmethoden gegen Sensorstörungen zu beurteilen.

5 Erstmals wurde der Complex-YOLO-Ansatz für die Kombination von LiDAR- und RADAR-Daten erweitert und implementiert.

6 Vergleich von Low-Level- und High-Level-Fusionsstrategien für die Objektdetektion unter verschiedenen Sensorstörungen.

7 Anwendung der entwickelten Trainings- und Evaluationskonzepte auf unterschiedliche reale Daten verschiedene Fusionsmethoden. Dies hebt die modell- und modalitätsagnostische Einsetzbarkeit des Konzepts hervor.

8 Untersuchung des Einflusses der Netzwerkarchitekturtiefe auf die Robustheit gegenüber Sensorstörungen.

Mit den beiden Methoden GAOS und DLOS, welche in [Bere21b] und [Bere24a] veröffentlicht wurden, ist es möglich, die Sensorplatzierung zu optimieren. Insbesondere DLOS, die auf verschiedenen Fahrzeuggeometrien trainiert werden und somit effizient auf ein neues Fahrzeug angewendet werden kann, ermöglicht eine Anwendung bereits in der Systementwicklungsphase, wodurch wertvolle Zeit eingespart wird und der Entwicklungsprozess effizienter gestaltet werden kann.

Das neu entwickelte Konzept zur Steigerung der Robustheit, bestehend aus dem Trainingskonzept KISA und den Evaluationsmethodiken Multifaktorielle Performanz-Evaluierung und Sensitivitätsanalyse, wird in verschiedenen Anwendungen sowohl im autonomen Fahren als auch in der medizinischen Bildgebung getestet. Die Anwendung des Konzeptes zur Steigerung der Robustheit auf Complex-YOLO mit Low Level Fusionsstrategie wurde in [Bere24b] veröffentlicht. Die Arbeit hat gezeigt, dass durch die systematische Anwendung von KISA auf verschiedene Sensortypen und Datenmodalitäten die Robustheit von Fusionsnetzwerken verbessert werden kann. Die eingeführten neuen Evaluationsmethodiken der Multifaktoriellen Performanz-Evaluierung und Sensitivitätsanalyse ermöglichen es, die Dynamik der Robustheitsentwicklung quantitativ zu erfassen und zu modellieren.

Mit den Forschungsergebnissen dieser Dissertation können die in Kapitel 1.8 aufgestellten Forschungsfragen beantworten werden:

1 **Wie beeinflussen gestörte Trainingsdaten die Robustheit von Fusionsnetzwerken?**
Die Untersuchungen zeigen, dass durch KISA, bei der gestörte Trainingsdaten in die Trainingsroutine integriert werden, die Robustheit von Fusionsnetzwerken signifikant verbessert werden kann.

Modelle, die mit einem spezifischen Satz von Störungen trainiert wurden, zeigten eine verbesserte Fähigkeit, diese Störungen während des Einsatzes effektiv zu handhaben. Dies bestätigt die Hypothese, dass die Integration von Störungen während des Trainings die Robustheit des Netzwerks gegenüber ähnlichen Bedingungen im Einsatz erhöht. Weiterhin zeigt sich, dass durch die Anwendung von KISA auch die Leistung auf ungestörten Daten verbessert werden kann.

2 **Wie beeinflussen verschiedene Störungsarten das Trainingsergebnis?**
Sensitivitätsanalysen können zeigen, dass verschiedene Arten von Störungen unterschiedliche Auswirkungen auf die Modellleistung haben. Störungen, die Informationen aus den Daten entfernen, stellen eine besondere Herausforderung dar. Durch die Anwendung von KISA können diese informationslöschenden Störungen bis zu einem gewissen Grad, jedoch nicht vollständig, kompensiert werden, während Störungen, die die Information in den Daten verändern, aber nicht löschen, fast vollständig mit KISA kompensiert werden können.

3 **Welchen Einfluss haben Low Level und High Level Fusion auf das Trainingsergebnis?**
Die Analysen ergeben, dass High Level Fusion robuster gegenüber Störung der Eingangsdaten als Low Level Fusion ist, wenn die Fusionsmethoden nur mit ungestörten Daten trainiert sind. Wenn KISA angewendet wird und damit auch gestörte Trainingsdaten integriert werden, so zeigt sich, dass Low Level Fusion robuster gegenüber Störungen ist als High Level Fusion. Low Level Fusion fördert eine tiefergehende und frühzeitigere Fusion der Daten, was zu einer verbesserten Fehlerkorrektur und Informationsnutzung führt. High Level Fusion hingegen kann spezifische Störungen nicht immer effektiv kompensieren, da sie lediglich die Ergebnisse der einzelnen Sensorverarbeitungen zusammenführt, wodurch die Fehler, die durch Sensorstörung entstehen, in das Ergebnis übertragen werden.

4 **Wie muss ein Netzwerk zur Integration gestörter Daten gestaltet werden?**
Des Weiteren wird erstmals der Einfluss der Tiefe von Netzwerkarchitektur hinsichtlich der Robustheit gegenüber Sensorstörung untersucht. Die Anwendung des Trainings- und Evaluationskonzepts auf den verschiedenen Modellvarianten zeigt, dass größere Modelle eine höhere Robustheit auf gestörten Daten erreichen. Dies wird durch die multifaktorielle Performanz nachgewiesen, die bei den größeren Modellen auch bei Störungen näher an der Leistung der ungestörten Daten liegt. Größere und tiefere Netzwerke tendierten dazu, besser mit komplexen Störungen umzugehen und gleichzeitig hohe Leistungen auf ungestörten Daten zu erzielen. Dies deutet darauf hin, dass eine sorgfältige Überlegung der Netzwerktiefe und -komplexität wesentlich ist, um die Kapazität für die Verarbeitung und Korrektur von Störungen zu maximieren.

Diese Arbeit zeigt auf, dass KISA eine entscheidende Strategie darstellt, um die Robustheit gegen sensorische Störungen signifikant zu steigern. Die Ergebnisse der Analysen mit diesem Konzept verdeutlichen, dass unterschiedliche Störungsarten und Schwierigkeitsgrade die Leistungsfähigkeit von Modellen in variierendem Maße beeinflussen, was die Bedeutung einer sorgfältigen Abwägung des Störungsanteils im Trainingsdatensatz hervorhebt. Dadurch wird mit dem Trainings- und Evaluationskonzept die anfänglich aufgestellte Hypothese bestätigt: **Die Integration von Störungen während des Trainingsprozesses kann die Robustheit von Fusionsnetzwerken gegen Sensorstörungen verbessern.**

Es empfiehlt sich, in zukünftigen Entwicklungen von Multisensorsystemen diesem Konzept zu folgen und Störungen als integralen Bestandteil der Konzeption zu betrachten. Es sollte schon bei der Wahl der Sensoren über mögliche Störungen nachgedacht werden. Um die Integrität und Robustheit von Systemen gegenüber Sensorstörungen zu gewährleisten, sollte eine sorgfältige Auswahl und Platzierung der Sensoren in Verbindung mit einem strukturierten Trainingsansatz erfolgen. Die entwickelten Methoden der Sensorplatzierung können dies objektiv bearbeiten. Es wird empfohlen, den Trainingsdatensatz

mit 10% bis 20% gestörten Daten zu ergänzen, sofern keine detaillierte Analyse des optimalen Anteils möglich ist. Dieser Anteil sollte erhöht werden, wenn mehrere Störungsarten gleichzeitig berücksichtigt werden. Die Auswahl der Störungen hängt von den eingesetzten Sensoren und dem spezifischen Anwendungsfeld ab; eine initiale Sensitivitätsanalyse hilft dabei, kritische Störungen zu identifizieren. Sollten nach Anwendung von KISA weiterhin signifikante Leistungsverluste auftreten, ist die Wahl einer alternativen Fusionsstrategie oder eines Modells mit höherer Lernkapazität zu erwägen. Für die Fusionsstrategie wird die Low Level Fusion empfohlen, wenn Störungen vorhanden oder simulierbar sind, da sie eine höhere Anpassungsfähigkeit aufweist. Bei spezifischen, bekannten Störungen ist hingegen die High Level Fusion vorzuziehen, da sie durch die getrennte Auswertung der Sensordaten eine robuste Leistung sicherstellt.

Diese Dissertation hat wertvolle Einblicke in die Verbesserung der Robustheit von Fusionsnetzwerken durch die systematische Integration von Störungen im Trainingsprozess geliefert und die zwei Extremstellen, Low Level und High Level Fusion, bezüglich der Möglichkeiten der Robustheit untersucht. Ein spannendes Feld für zukünftige Forschungen bietet die Mid Level Fusion unter dem Aspekt der Robustheit. Insbesondere die Mid Level Fusionsstrategie mit mehreren Fusions-Schnittstellen kann tiefergehend evaluiert werden, um die optimale Stelle für die Datenfusion zu bestimmen und den Einfluss der Fusionsstelle auf die Robustheit gegenüber Sensorstörung zu messen. Dies wird nicht nur dazu beitragen, die Effektivität verschiedener Fusionsstrategien unter variierenden Störungsbedingungen zu verstehen, sondern auch dazu, die Informationsnutzung innerhalb des Netzwerks zu maximieren.

Darüber hinaus bietet KISA Potenzial für die Anwendung in anderen maschinellen Lernszenarien, die über sensorische Datenfusion hinausgehen. So kann dieses Konzept auf Probleme der Sprachverarbeitung, wo es beispielsweise um die Bewältigung von Hintergrundrauschen oder Dialektvariationen geht, oder auf die Vorhersage von Zeitreihen, etwa bei der Analyse von verrauschten Finanzmarktdaten oder unvollständigen Wetterdatensätzen, angewandt werden. In beiden Fällen ist die Robustheit gegenüber verrauschten Daten oder fehlerhaften Eingaben von entscheidender Bedeutung.

Daher eröffnet die erfolgreiche Anwendung dieses Konzepts auf neue, herausfordernde Bereiche nicht nur innovative Wege zur Verbesserung der Datenrobustheit, sondern markiert auch einen entscheidenden Schritt vorwärts in der Entwicklung intelligenter Systeme, die selbst in unvorhersehbaren und fehlerbehafteten Umgebungen zuverlässig funktionieren.

Literatur

[Asga21] Asgari Taghanaki, S.; Abhishek, K.; Cohen, J. P.; Cohen-Adad, J. und Hamarneh, G.: „Deep Semantic Segmentation of Natural and Medical Images: A Review“. In: *Artificial Intelligence Review* 54 (2021), S. 137–178 (siehe S. 170).

[Aydo20] Aydogdu, C.; Keskin, M. F.; Carvajal, G. K.; Eriksson, O.; Hellsten, H.; Herbertsson, H.; Nilsson, E.; Rydstrom, M.; Vanas, K. und Wymeersch, H.: „Radar Interference Mitigation for Automated Driving: Exploring Proactive Strategies“. In: *IEEE Signal Processing Magazine* 37.4 (2020), S. 72–84 (siehe S. 9, 10).

[Badr17] Badrinarayanan, V.; Kendall, A. und Cipolla, R.: „Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.12 (2017), S. 2481–2495 (siehe S. 165).

[Baid21] Baid, U.; Ghodasara, S.; Mohan, S.; Bilello, M.; Calabrese, E.; Colak, E.; Farahani, K.; Kalpathy-Cramer, J.; Kitamura, F. C. und Pati, S. et al.: „The rsna-asnr-miccai BraTS 2021 Benchmark on Brain Tumor Segmentation and Radiogenomic Classification“. In: *arXiv preprint arXiv:2107.02314* (2021) (siehe S. 52, 189).

[Baka17] Bakas, S.; Akbari, H.; Sotiras, A.; Bilello, M.; Rozycki, M.; Kirby, J. S.; Freymann, J. B.; Farahani, K. und Davatzikos, C.: „Advancing the Cancer Genome Atlas Glioma MRI Collections with Expert Segmentation Labels and Radiomic

Features“. In: *Scientific data* 4.1 (2017), S. 1–13 (siehe S. 52, 189).

[Bala16] Balal, N.; Pinhasi, G. A. und Pinhasi, Y.: „Atmospheric and Fog Effects on Ultra-Wide band Radar Operating at Extremely High Frequencies“. In: *Sensors* 16.5 (2016), S. 751 (siehe S. 9).

[Bast20] Bastan, M.; Yap, K.-H. und Chau, L.-P.: „Remote Detection of Idling Cars using Infrared Imaging and Deep Networks“. In: *Neural Computing and Applications* 32 (2020), S. 3047–3057 (siehe S. 3).

[Bere21a] Berens, F.; Elser, S. und Reischl, M.: „Evaluation of Four Point Cloud Similarity Measures for the use in Autonomous Driving“. In: *at-Automatisierungstechnik* 69.6 (2021), S. 503–514 (siehe S. 36, 37).

[Bere21b] Berens, F.; Elser, S. und Reischl, M.: „Genetic Algorithm for the Optimal LiDAR Sensor Configuration on a Vehicle“. In: *IEEE Sensors journal* 22.3 (2021), S. 2735–2743 (siehe S. 31, 64, 71, 77, 115).

[Bere24a] Berens, F.; Ambs, J.; Elser, S. und Reischl, M.: „A Novel Approach to Light Detection and Ranging Sensor Placement for Autonomous Driving Vehicles Using Deep Deterministic Policy Gradient Algorithm“. In: *SAE International Journal of Connected and Automated Vehicles* 7.12-07-03-0019 (2024) (siehe S. 31, 64, 71, 76, 77, 115).

[Bere24b] Berens, F.; Koschinski, Y.; Badami, M. K.; Geimer, M.; Elser, S. und Reischl, M.: „Adaptive Training for Robust Object Detection in Autonomous Driving Environments“. In: *IEEE Transactions on Intelligent Vehicles* (2024), S. 1–15 (siehe S. 79, 96, 98, 99, 115).

[Berg17] Bergelt, R.; Khan, O. und Hardt, W.: „Improving the Intrinsic Calibration of a Velodyne LiDAR Sensor“. In: *2017 IEEE SENSORS*. IEEE. 2017, S. 1–3 (siehe S. 159).

[Besl92] Besl, P. J. und McKay, N. D.: „Method for Registration of 3-D Shapes“. In: *Sensor Fusion IV: control paradigms and data structures*. Bd. 1611. Spie. 1992, S. 586–606 (siehe S. 161).

[Bije18] Bijelic, M.; Gruber, T. und Ritter, W.: „A Benchmark for LiDAR Sensors in Fog: Is Detection Breaking Down?“ In: *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2018, S. 760–767 (siehe S. 7, 8).

[Bije20] Bijelic, M.; Gruber, T.; Mannan, F.; Kraus, F.; Ritter, W.; Dietmayer, K. und Heide, F.: „Seeing through Fog without seeing Fog: Deep Multimodal Sensor Fusion in unseen Adverse Weather“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, S. 11682–11692 (siehe S. 19).

[Boch20] Bochkovskiy, A.; Wang, C.-Y. und Liao, H.-Y. M.: „Yolov4: Optimal Speed and Accuracy of Object Detection“. In: *arXiv preprint arXiv:2004.10934* (2020) (siehe S. 202).

[Brud03] Bruder, J.; Carlo, J.; Gurney, J. und Gorman, J.: „IEEE Standard for Letter Designations for Radar-Frequency Bands“. In: *IEEE Aerospace & Electronic Systems Society* (2003), S. 1–3 (siehe S. 4).

[Caic19] Caicedo, J. C. u. a.: „Nucleus Segmentation across imaging experiments: the 2018 Data Science Bowl“. In: *Nature Methods* 16.12 (2019), S. 1247–1253 (siehe S. 69).

[Chen17] Chen, L.-C.; Papandreou, G.; Kokkinos, I.; Murphy, K. und Yuille, A. L.: „Deeplab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFS“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4 (2017), S. 834–848 (siehe S. 165).

[Chen22] Chen, J.; Jia, K.; Chen, W.; Lv, Z. und Zhang, R.: „A Real-Time and High-Precision Method for Small Traffic-Signs Recognition“. In: *Neural Computing and Applications* (2022), S. 1–13 (siehe S. 3).

[Clif20] Clifton, J. und Laber, E.: „Q-Learning: Theory and Applications“. In: *Annual Review of Statistics and Its Application* 7 (2020), S. 279–301 (siehe S. 35).

[Comm17] Commission, Federal Communications: „Radar Services in the 76-81 GHz Band“. In: *Washington, DC, USA, Tech. Rep. ET Docket* 15-26 (2017) (siehe S. 4).

[Csur10] Csurgai-Horváth, L. und Bitó, J.: „Fog Attenuation on V Band Terrestrial Radio and a Low-Cost Measurement Setup“. In: *2010 Future Network & Mobile Summit*. IEEE. 2010, S. 1–9 (siehe S. 9).

[Dala05] Dalal, N. und Triggs, B.: „Histograms of Oriented Gradients for Human Detection“. In: *2005 IEEE computer society Conference on Computer Vision and Pattern Recognition (CVPR)*. Bd. 1. Ieee. 2005, S. 886–893 (siehe S. 163, 164).

[Devi17] Devices, Short Range: „Transport and Traffic Telematics (TTT); Short Range Radar Equipment Operating in the 77 GHz to 81 GHz Band; Harmonised Standard Covering the Essential Requirements of Article 3.2 of Directive 2014/53/EU“. In: *European Telecommunications Standards Institute EN* 302 (2017), S. 264 (siehe S. 4).

[Domh19] Domhof, J.; Kooij, J. F. und Gavrila, D. M.: „An Extrinsic Calibration Tool for Radar, Camera and LiDAR“. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, S. 8107–8113 (siehe S. 161).

[Doso17] Dosovitskiy, A.; Ros, G.; Codevilla, F.; Lopez, A. und Koltun, V.: „CARLA: An Open Urban Driving Simulator“. In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, S. 1–16 (siehe S. 63).

[Du13] Du, L.; Wu, C.; Liu, Q.; Huang, L. und Wang, P.: „Recent Advances in Olfactory Receptor-Based Biosensors“. In: *Biosensors and Bioelectronics* 42 (2013), S. 570–580 (siehe S. 3).

[Dybe20a] Dybedal, J. und Hovland, G.: „GPU-Based Occlusion Minimisation for Optimal Placement of Multiple 3D Cameras“. In: *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE. 2020, S. 967–972 (siehe S. 20).

[Dybe20b] Dybedal, J. und Hovland, G.: „GPU-Based Optimisation of 3D Sensor Placement Considering Redundancy, Range and Field of View“. In: *2020 15th IEEE Conference on Industrial Electronics and Applications (ICIEA)*. IEEE. 2020, S. 1484–1489 (siehe S. 20).

[Emad22] Emadi, M.: „Radar Technology“. In: *Advanced Driver Assistance Systems and Autonomous Vehicles: From Fundamentals to Applications* (2022), S. 265–304 (siehe S. 10).

[Even72] Evenson, K.; Wells, J.; Petersen, F.; Danielson, B.; Day, G. W.; Barger, R. und Hall, J.: „Speed of Light from Direct Frequency and Wavelength Measurements of the Methane-Stabilized Laser“. In: *Physical Review Letters* 29.19 (1972), S. 1346 (siehe S. 4).

[Ever15] Everingham, M.; Eslami, S. A.; Van Gool, L.; Williams, C. K.; Winn, J. und Zisserman, A.: „The Pascal Visual Object Classes Challenge: A Retrospective“. In: *International Journal of Computer Vision* 111 (2015), S. 98–136 (siehe S. 169).

[Feng20] Feng, D.; Haase-Schütz, C.; Rosenbaum, L.; Hertlein, H.; Glaeser, C.; Timm, F.; Wiesbeck, W. und Dietmayer, K.: „Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges“. In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2020), S. 1341–1360 (siehe S. 13, 15–17).

[Gao19] Gao, X.; Xing, G.; Roy, S. und Liu, H.: „Experiments with MMWAVE Automotive Radar Test-Bed“. In: *2019 53rd Asilomar Conference on Signals, Systems, and Computers*. IEEE. 2019, S. 1–6 (siehe S. 155).

[Geig12a] Geiger, A.; Lenz, P. und Urtasun, R.: „Are we Ready for Autonomous Driving? The KITTI Vision Benchmark Suite“. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2012, S. 3354–3361 (siehe S. 20, 159, 160).

[Geig12b] Geiger, A.; Moosmann, F.; Car, Ö. und Schuster, B.: „Automatic Camera and Range Sensor Calibration using a Single Shot“. In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, S. 3936–3943 (siehe S. 159, 160).

[Geig13] Geiger, A.; Lenz, P.; Stiller, C. und Urtasun, R.: „Vision Meets Robotics: The KITTI Dataset“. In: *The International Journal of Robotics Research* 32.11 (2013), S. 1231–1237 (siehe S. 11, 51).

[Geig23] Geiger, A.; Lenz, P.; Stiller, C. und Urtasun, R.: 3D Object Detection Evaluation 2017. 2023. url: https://www.cvlibs.net/datasets/kitti/eval_Object.php?obj_benchmark=3d (besucht am 25. 07. 2024) (siehe S. 169).

[Geye20] Geyer, J.; Kassahun, Y.; Mahmudi, M.; Ricou, X.; Durgesh, R.; Chung, A. S.; Hauswald, L.; Pham, V. H.; Mühlegg, M. und Dorn, S.: „A2d2: Audi Autonomous Driving Dataset“. In: *arXiv preprint arXiv:2004.06320* (2020) (siehe S. 51, 96, 223, 224).

[Girs14] Girshick, R.; Donahue, J.; Darrell, T. und Malik, J.: „Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, S. 580–587 (siehe S. 164).

[Girs15] Girshick, R.: „Fast R-CNN“. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, S. 1440–1448 (siehe S. 22, 164).

[Guan20] Guan, J.; Madani, S.; Jog, S.; Gupta, S. und Hassanieh, H.: „Through Fog High-Resolution Imaging using Millimeter Wave Radar“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, S. 11464–11473 (siehe S. 21).

[Guo17] Guo, X.; Liu, X.; Zhu, E. und Yin, J.: „Deep Clustering with Convolutional Autoencoders“. In: *Neural Information Processing: 24th International Conference, ICONIP 2017, Guangzhou, China, November 14-18, 2017, Proceedings, Part II 24*. Springer. 2017, S. 373–382 (siehe S. 175).

[Hahn21] Hahner, M.; Sakaridis, C.; Dai, D. und Van Gool, L.: „Fog Simulation on Real LiDAR Point Clouds for 3D Object Detection in Adverse Weather“. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, S. 15283–15292 (siehe S. 60, 211).

[Hahn22] Hahner, M.; Sakaridis, C.; Bijelic, M.; Heide, F.; Yu, F.; Dai, D. und Van Gool, L.: „Lidar Snowfall Simulation for Robust 3d Object Detection“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, S. 16364–16374 (siehe S. 60, 214).

[Hasi17] Hasirlioglu, S. und Riener, A.: „Introduction to Rain and Fog Attenuation on Automotive Surround Sensors“. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2017, S. 1–7 (siehe S. 6, 7, 9).

[Hata21] Hatamizadeh, A.; Nath, V.; Tang, Y.; Yang, D.; Roth, H. R. und Xu, D.: „Swin unetr: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images“. In: *International MICCAI Brainlesion Workshop*. Springer. 2021, S. 272–284 (siehe S. 52, 189).

[He15] He, K.; Zhang, X.; Ren, S. und Sun, J.: „Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.9 (2015), S. 1904–1916 (siehe S. 202).

[He16] He, K.; Zhang, X.; Ren, S. und Sun, J.: Deep Residual Learning for Image Recognition. 2016 (siehe S. 22).

[Heik97] Heikkila, J. und Silvén, O.: „A Four-Step Camera Calibration Procedure with Implicit Image Correction“. In: *Proceedings of IEEE computer society Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 1997, S. 1106–1112 (siehe S. 159).

[IEEE90] IEEE: „IEEE Standard Glossary of Software Engineering Terminology“. In: *IEEE Std 610.12-1990* (1990), S. 1–84 (siehe S. 18).

[Igna22] Ignatious, H. A. und Khan, M. et al.: „An Overview of Sensors in Autonomous Vehicles“. In: *Procedia Computer Science* 198 (2022), S. 736–741 (siehe S. 3, 5, 157).

[Ijaz12] Ijaz, M.; Ghassemlooy, Z.; Le Minh, H.; Rajbhandari, S. und Perez, J.: „Analysis of Fog and Smoke Attenuation in a Free Space Optical Communication Link under Controlled Laboratory Conditions“. In: *2012 International Workshop on Optical Wireless Communications (IWOW)*. IEEE. 2012, S. 1–3 (siehe S. 7, 8).

[Imra21] Imran, S.; Liu, X. und Morris, D.: „Depth completion with twin surface extrapolation at occlusion boundaries“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, S. 2583–2592 (siehe S. 101, 228).

[Jado20] Jadon, S.: „A Survey of Loss Functions for Semantic Segmentation“. In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. IEEE. 2020, S. 1–7 (siehe S. 176).

[Jana20] Janai, J.; Güney, F.; Behl, A. und Geiger, A. et al.: „Computer vision for Autonomous Vehicles: Problems, datasets and state of the art“. In: *Foundations and Trends® in Computer Graphics and Vision* 12.1–3 (2020), S. 1–308 (siehe S. 11, 168).

[Jezz99] Jezzard, P. und Clare, S.: „Sources of Distortion in Functional MRI Data“. In: *Human brain mapping* 8.2-3 (1999), S. 80–85 (siehe S. 7).

[Jung19] Jung, Y. H.; Park, B.; Kim, J. U. und Kim, T.: „Bioinspired Electronics for Artificial Sensory Systems“. In: *Advanced Materials* 31.34 (2019), S. 1803637 (siehe S. 3).

[Ki18] Ki, S.; Sim, H.; Choi, J.-S.; Kim, S. und Kim, M.: „Fully End-to-End Learning Based Conditional Boundary Equilibrium GAN with Receptive Field Sizes Enlarged for Single Ultra-High Resolution Image Dehazing“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2018, S. 817–824 (siehe S. 19).

[Kim18] Kim, K. W.: „The Comparison of Visibility Measurement between Image-Based Visual Range, Human Eye-Based Visual Range, and Meteorological Optical Range“. In: *Atmospheric Environment* 190 (2018), S. 74–86 (siehe S. 212).

[Kim19] Kim, T.-H. und Park, T.-H.: „Placement Optimization of Multiple LiDAR Sensors for Autonomous Vehicles“. In: *IEEE Transactions on Intelligent Transportation Systems* 21.5 (2019), S. 2139–2145 (siehe S. 21, 31).

[Koci18] Kocić, J.; Jovičić, N. und Drndarević, V.: „Sensors and Sensor Fusion in Autonomous Vehicles“. In: *2018 26th Telecommunications Forum (TELFOR)*. IEEE. 2018, S. 420–425 (siehe S. 13–17).

[Komi21] Komissarov, R. und Wool, A.: „Spoofing Attacks Against Vehicular FMCW Radar“. In: *Proceedings of the 5th Workshop on Attacks and Solutions in Hardware Security*. 2021, S. 91–97 (siehe S. 10).

[Kune12] Kunert, I.: Project Final Report, MOSARIM: More Safety for All by Radar Interference Mitigation. 2012 (siehe S. 9, 10).

[LeCl21] Le'Clerc Arrastia, J.; Heilenkötter, N.; Otero Baguer, D.; Hauberg-Lotte, L.; Boskamp, T.; Hetzer, S.; Duschner, N.; Schaller, J. und Maass, P.: „Deeply Supervised UNet for Semantic Segmentation to Assist Dermatopathological Assessment of Basal Cell Carcinoma“. In: *Journal of Imaging* 7.4 (2021) (siehe S. 69).

[Lian18] Liang, M.; Yang, B.; Wang, S. und Urtasun, R.: „Deep Continuous Fusion for Multi-Sensor 3d Object Detection“. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, S. 641–656 (siehe S. 16).

[Lian19] Liang, M.; Yang, B.; Chen, Y.; Hu, R. und Urtasun, R.: „Multi-Task Multi-Sensor Fusion for 3d Object Detection“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, S. 7345–7353 (siehe S. 16, 166, 170).

[Liu16] Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y. und Berg, A. C.: „SSD: Single Shot Multibox Detector“. In: *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer. 2016, S. 21–37 (siehe S. 165).

[Liu18] Liu, S.; Qi, L.; Qin, H.; Shi, J. und Jia, J.: „Path Aggregation Network for Instance Segmentation“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, S. 8759–8768 (siehe S. 202).

[Liu19] Liu, Z.; Arief, M. und Zhao, D.: „Where should we place LiDARs on the Autonomous Vehicle?-An Optimal Design Approach“. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, S. 2793–2799 (siehe S. 20, 21).

[Liu21] Liu, X.; Tian, W.; Yu, J. und Zhao, J.: „Lidar Points' Bumpy Distortion Model, Displacements and Experiments“. In: *2021 5th CAA International Conference on Vehicular Control and Intelligence (CVCI)*. IEEE. 2021, S. 1–6 (siehe S. 7, 156).

[Llor13] Llorca, D. F.; Arroyo, R. und Sotelo, M. A.: „Vehicle Logo Recognition in Traffic Images using HOG Features and SVM“. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. IEEE. 2013, S. 2229–2234 (siehe S. 164).

[Long15] Long, J.; Shelhamer, E. und Darrell, T.: „Fully Convolutional Networks for Semantic Segmentation“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, S. 3431–3440 (siehe S. 175).

[Lonn05] Lonnoy, J.; Le Guilloux, Y. und Moreira, R.: „Far-Infrared Cameras for Automotive safety“. In: *Photonics in the Automobile*. Bd. 5663. SPIE. 2005, S. 156–169 (siehe S. 3).

[Lowe99] Lowe, D. G.: „Object Recognition from Local Scale-Invariant Features“. In: *Proceedings of the seventh IEEE International Conference on Computer Vision*. Bd. 2. Ieee. 1999, S. 1150–1157 (siehe S. 163).

[Ma12] Ma, H. und Wu, J.: „Analysis of Positioning Errors Caused by Platform Vibration of Airborne LiDAR System“. In: *2012 8th IEEE International Symposium on Instrumentation and Control Technology (ISICT) Proceedings*. IEEE. 2012, S. 257–261 (siehe S. 8).

[Mait18] Maity, A. und Chatterjee, R.: „Impulsive Noise in Images: A Brief Review“. In: *Computer Vision Graphics and Image Processing* 4 (2018), S. 6–15 (siehe S. 5, 6, 18, 179).

[Malm03] Malm, H. und Heyden, A.: „Simplified Intrinsic Camera Calibration and Hand-Eye Calibration for Robot Vision“. In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*. Bd. 1. IEEE. 2003, S. 1037–1043 (siehe S. 159).

[Marc21] Marchegiani, L. und Fafoutis, X.: „How Well Can Driverless Vehicles Hear? An Introduction to Auditory Perception for Autonomous and Smart Vehicles“. In: *IEEE Intelligent Transportation Systems Magazine* 14.3 (2021), S. 92–105 (siehe S. 3).

[Mein98] Meinel, H. H.: „Automotive Millimeterwave Radar History and Present Status“. In: *1998 28th European Microwave Conference*. Bd. 1. IEEE. 1998, S. 619–629 (siehe S. 155).

[Menz14] Menze, B. H.; Jakab, A.; Bauer, S.; Kalpathy-Cramer, J.; Farahani, K.; Kirby, J.; Burren, Y.; Porz, N.; Slotboom, J. und Wiest, R. et al.: „The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)“. In: *IEEE transactions on medical imaging* 34.10 (2014), S. 1993–2024 (siehe S. 52, 189).

[Meye19] Meyer, M. und Kuschk, G.: „Automotive Radar Dataset for Deep Learning Based 3d Object Detection“. In: *2019 16th European Radar Conference (EuRAD)*. IEEE. 2019, S. 129–132 (siehe S. 51, 216).

[Mou18] Mou, S.; Chang, Y.; Wang, W. und Zhao, D.: „An Optimal LiDAR Configuration Approach for Self-Driving Cars“. In: *arXiv preprint arXiv:1805.07843* (2018) (siehe S. 20, 21).

[Münk24] Münke, F. R.; Schütze, J.; Berens, F. und Reischl, M.: „A Review of Adaptable Conventional Image Processing Pipelines and Deep Learning on limited Datasets“. In: *Machine Vision and Applications* 35.2 (2024), S. 25 (siehe S. 51, 55, 56, 164).

[Nand16] Nandwana, M. K. und Hasan, T.: „Towards Smart-Cars That Can Listen: Abnormal Acoustic Event Detection on the Road.“ In: *INTERSPEECH*. 2016, S. 2968–2971 (siehe S. 3).

[Nugr17] Nugraha, B. T. und Su, S.-F. et al.: „Towards Self-Driving Car using Convolutional Neural Network and Road Lane Detector“. In: *2017 2nd International Conference on Automation, Cognitive Science, Optics, Micro Electro-Mechanical System, and Information Technology (ICACOMIT)*. IEEE. 2017, S. 65–69 (siehe S. 5).

[Palf22] Palffy, A.; Pool, E.; Baratam, S.; Kooij, J. und Gavrila, D. M.: „Multi-Class Road user Detection with 3+ 1D Radar in the View-of-Delft Dataset“. In: *IEEE Robotics and Automation Letters* 7.2 (2022), S. 4961–4968 (siehe S. 51, 98, 225).

[Pang20] Pang, S.; Morris, D. und Radha, H.: „CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection“. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, S. 10386–10393 (siehe S. 17).

[Park23] Park, J.; Cho, J.; Lee, S.; Bak, S. und Kim, Y.: „An Automotive LiDAR Performance Test Method in Dynamic Driving Conditions“. In: *Sensors* 23.8 (2023), S. 3892 (siehe S. 7).

[Peršı17] Peršić, J.; Marković, I. und Petrović, I.: „Extrinsic 6DoF Calibration of 3D LiDAR and Radar“. In: *2017 European Conference on Mobile Robots (ECMR)*. IEEE. 2017, S. 1–6 (siehe S. 161).

[Peršı19] Peršić, J.; Marković, I. und Petrović, I.: „Extrinsic 6DoF Calibration of a Radar–LiDAR–Camera System Enhanced by Radar Cross Section Estimates Evaluation“. In: *Robotics and Autonomous Systems* 114 (2019), S. 217–230 (siehe S. 161).

[Peti15] Petit, J.; Stottelaar, B.; Feiri, M. und Kargl, F.: „Remote Attacks on Automated Vehicles Sensors: Experiments on Camera and LiDAR“. In: *Black Hat Europe* 11.2015 (2015), S. 995 (siehe S. 7).

[Peyn09] Peynot, T.; Underwood, J. und Scheding, S.: „Towards Reliable Perception for Unmanned Ground Vehicles in Challenging Conditions“. In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2009, S. 1170–1176 (siehe S. 7, 8).

[Pitr21] Pitropov, M.; Garcia, D. E.; Rebello, J.; Smart, M.; Wang, C.; Czarnecki, K. und Waslander, S.: „Canadian Adverse Driving Conditions Dataset“. In: *The International Journal of Robotics Research* 40.4-5 (2021), S. 681–690 (siehe S. 5, 7, 8).

[Qi17] Qi, C. R.; Su, H.; Mo, K. und Guibas, L. J.: „Pointnet: Deep Learning on Point Sets for 3d Classification and Segmentation“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, S. 652–660 (siehe S. 166).

[Qian21] Qian, K.; Zhu, S.; Zhang, X. und Li, L. E.: „Robust Multimodal Vehicle Detection in Foggy Weather using Complementary LiDAR and Radar Signals“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, S. 444–453 (siehe S. 23).

[Rabl19] Rablau, C.: „LIDAR–A new (Self-Driving) Vehicle for Introducing Optics to Broader Engineering and Non-Engineering Audiences“. In: *Education and Training in Optics and Photonics*. Optica Publishing Group. 2019, 11143_138 (siehe S. 4).

[Raj20] Raj, T.; Hashim, F. H.; Huddin, A. B.; Ibrahim, M. F. und Hussain, A.: „A Survey on LiDAR Scanning Mechanisms“. In: *Electronics* 9.5 (2020), S. 741 (siehe S. 156–158).

[Rass11] Rasshofer, R. H.; Spies, M. und Spies, H.: „Influences of Weather Phenomena on Automotive Laser Radar Systems“. In: *Advances in radio science* 9 (2011), S. 49–60 (siehe S. 7, 211).

[Redm16] Redmon, J.; Divvala, S.; Girshick, R. und Farhadi, A.: „You Only Look Once: Unified, Real-Time Object Detection“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, S. 779–788 (siehe S. 165, 208).

[Reic21] Reichert, H.; Lang, L.; Rösch, K.; Bogdoll, D.; Doll, K.; Sick, B.; Rellss, H.-C.; Stiller, C. und Zöllner, J. M.: „Towards Sensor Data Abstraction of Autonomous Vehicle Perception Systems“. In: *2021 IEEE International Smart Cities Conference (ISC2)*. IEEE. 2021, S. 1–4 (siehe S. 3, 157).

[Ren15] Ren, S.; He, K.; Girshick, R. und Sun, J.: „Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks“. In: *Advances in Neural Information Processing Systems* 28 (2015) (siehe S. 164).

[Rich94] Richter, P.: „Air Pollution Monitoring with LIDAR“. In: *TrAC Trends in Analytical Chemistry* 13.7 (1994), S. 263–266 (siehe S. 8).

[Ronn15] Ronneberger, O.; Fischer, P. und Brox, T.: „U-net: Convolutional Networks for Biomedical Image Segmentation“. In: *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer. 2015, S. 234–241 (siehe S. 19, 69, 165, 175, 176).

[Rori21] Roriz, R.; Cabral, J. und Gomes, T.: „Automotive LiDAR Technology: A Survey“. In: *IEEE Transactions on Intelligent Transportation Systems* 23.7 (2021), S. 6282–6297 (siehe S. 156).

[Royo19] Royo, S. und Ballesta-Garcia, M.: „An Overview of LiDAR Imaging Systems for Autonomous Vehicles“. In: *Applied sciences* 9.19 (2019), S. 4093 (siehe S. 4, 155–158).

[Saka18] Sakaridis, C.; Dai, D. und Van Gool, L.: „Semantic Foggy Scene Understanding with Synthetic Data“. In: *International Journal of Computer Vision* 126.9 (2018), S. 973–992 (siehe S. 3, 157).

[Saka21] Sakaridis, C.; Dai, D. und Van Gool, L.: „ACDC: The Adverse Conditions Dataset with Correspondences for Semantic Driving Scene Understanding“. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, S. 10765–10775 (siehe S. 5).

[Salt83] Salton, G.: „Introduction to Modern Information Retrieval“. In: *McGraw-Hill* (1983) (siehe S. 169).

[Sant18] Santurkar, S.; Tsipras, D.; Ilyas, A. und Madry, A.: „How does Batch Normalization Help Optimization?“ In: *Advances in Neural Information Processing Systems* 31 (2018) (siehe S. 175).

[Sche20] Scherr, T.; Löffler, K.; Böhland, M. und Mikut, R.: „Cell Segmentation and Tracking using CNN-Based Distance Predictions and a Graph-Based Matching Strategy“. In: *PLOS ONE* 15.12 (Dez. 2020), S. 1–22 (siehe S. 69).

[Schi22] Schilling, M. P.; Scherr, T.; Münke, F. R.; Neumann, O.; Schutera, M.; Mikut, R. und Reischl, M.: „Automated Annotator Variability Inspection for Biomedical Image Segmentation“. In: *IEEE Access* 10 (2022), S. 2753–2765 (siehe S. 69).

[Schl22] Schlager, B.; Goelles, T.; Behmer, M.; Muckenhuber, S.; Payer, J. und Watzenig, D.: „Automotive LiDAR and Vibration: Resonance, Inertial Measurement Unit, and effects on the

Point Cloud“. In: *IEEE Open Journal of Intelligent Transportation Systems* 3 (2022), S. 426–434 (siehe S. 7, 8, 156).

[Schu20] Schutera, M.; Hussein, M.; Abhau, J.; Mikut, R. und Reischl, M.: „Night-to-day: Online Image-to-Image Translation for Object Detection within Autonomous Driving by Night“. In: *IEEE Transactions on Intelligent Vehicles* 6.3 (2020), S. 480–489 (siehe S. 5, 19).

[Schu21] Schutera, Mark et al.: „Cuepervision: self-supervised learning for continuous domain adaptation without catastrophic forgetting“. In: *Image and Vision Computing* 106 (2021), S. 104079 (siehe S. 5).

[Shee21] Sheeny, M.; De Pellegrin, E.; Mukherjee, S.; Ahrabian, A.; Wang, S. und Wallace, A.: „RADIATE: A Radar Dataset for Automotive Perception in Bad Weather“. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, S. 1–7 (siehe S. 3, 5, 22, 157, 158).

[Shi19] Shi, S.; Wang, X. und Li, H.: „PointRCNN: 3d Object Proposal Generation and Detection from Point Cloud“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, S. 770–779 (siehe S. 166, 170).

[Shi20] Shi, S.; Guo, C.; Jiang, L.; Wang, Z.; Shi, J.; Wang, X. und Li, H.: „PV-RCNN: Point-Voxel Feature Set Abstraction for 3d Object Detection“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, S. 10529–10538 (siehe S. 170).

[Silv14] Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D. und Riedmiller, M.: „Deterministic Policy Gradient Algorithms“. In: *International Conference on Machine Learning*. Pmlr. 2014, S. 387–395 (siehe S. 35).

[Simo18] Simon, M.; Milzy, S.; Amendey, K. und Gross, H.-M.: „Complex-YOLO: An Euler-Eegion-Proposal for Real-Time 3d Object Detection on Point Clouds“. In: *Proceedings of the*

European Conference on Computer Vision (ECCV) Workshops. 2018 (siehe S. 166, 170, 195).

[Sing14] Singh, I. und Neeru, N.: „Performance Comparison of Various Image Denoising Filters under Spatial Domain“. In: *International Journal of Computer Applications* 96.19 (2014), S. 21–30 (siehe S. 5, 6, 18, 19, 179).

[Soma12] Somasundaram, K. und Kalavathi, P.: „Medical Image Denoising using Non-Linear Spatial Mean Filters for Edge Detection“. In: *Image* 7.4816 (2012), S. 2063 (siehe S. 18, 179).

[Song17] Song, H.; Liu, Z.; Du, H.; Sun, G.; Le Meur, O. und Ren, T.: „Depth-Aware Salient Object Detection and Segmentation via Multiscale Discriminative Saliency Fusion and Bootstrap Learning“. In: *IEEE Transactions on Image Processing* 26.9 (2017), S. 4204–4216 (siehe S. 14).

[Stei21] Steinhauser, D.; Held, P.; Thöresz, B. und Brandmeier, T.: „Towards Safe Autonomous Driving: Challenges of Pedestrian Detection in Rain with Automotive Radar“. In: *2020 17th European Radar Conference (EuRAD)*. IEEE. 2021, S. 409–412 (siehe S. 9).

[Suju17] Suju, D. A. und Jose, H.: „FLANN: Fast approximate Nearest Neighbour Search Algorithm for Elucidating Human-Wildlife Conflicts in Forest Areas“. In: *2017 Fourth International Conference on Signal Processing, Communication and Networking (ICSCN)*. IEEE. 2017, S. 1–6 (siehe S. 163).

[Sun20] Sun, Y.; Zuo, W.; Yun, P.; Wang, H. und Liu, M.: „FuseSeg: Semantic Segmentation of Urban Scenes Based on RGB and Thermal Data Fusion“. In: *IEEE Transactions on Automation Science and Engineering* 18.3 (2020), S. 1000–1011 (siehe S. 5, 19).

[Sun21] Sun, H.; Liu, X.; Xu, K.; Miao, J. und Luo, Q.: „Emergency Vehicles Audio Detection and Localization in Autonomous Driving“. In: *arXiv preprint arXiv:2109.14797* (2021) (siehe S. 3).

[Tsai87] Tsai, R.: „A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology using off-the-Shelf TV Cameras and Lenses“. In: *IEEE Journal on Robotics and Automation* 3.4 (1987), S. 323–344 (siehe S. 160).

[Varg21] Vargas, J.; Alsweiss, S.; Toker, O.; Razdan, R. und Santos, J.: „An Overview of Autonomous Vehicles Sensors and their Vulnerability to Weather Conditions“. In: *Sensors* 21.16 (2021), S. 5397 (siehe S. 3, 5, 157).

[Viik08] Viikari, V.; Varpula, T. und Kantanen, M.: „Automotive Radar Technology for Detecting Road Conditions. Backscattering Properties of Dry, Wet, and Icy Asphalt“. In: *2008 European Radar Conference*. IEEE. 2008, S. 276–279 (siehe S. 9).

[Vija19] Vijayan, V. und Kp, P.: „FLANN Based Matching with SIFT Descriptors for Drowsy Features Extraction“. In: *2019 Fifth International Conference on Image Information Processing (ICIIP)*. IEEE. 2019, S. 600–605 (siehe S. 163).

[Viol01] Viola, P. und Jones, M. et al.: „Robust Real-Time Object Detection“. In: *International Journal of Computer Vision* 4.34-47 (2001), S. 4 (siehe S. 163, 164).

[Wald21] Waldschmidt, C.; Hasch, J. und Menzel, W.: „Automotive Radar—From First efforts to Future Systems“. In: *IEEE Journal of Microwaves* 1.1 (2021), S. 135–148 (siehe S. 4, 155).

[Wang20a] Wang, C.-Y.; Liao, H.-Y. M.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W. und Yeh, I.-H.: „CSPNet: A new Backbone that can Enhance Learning Capability of CNN“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition workshops*. 2020, S. 390–391 (siehe S. 202).

[Wang20b] Wang, D.; Watkins, C. und Xie, H.: „MEMS Mirrors for LiDAR: A Review“. In: *Micromachines* 11.5 (2020), S. 456 (siehe S. 155, 156).

[Wang21] Wang, P.: „Research on Comparison of LiDAR and Camera in Autonomous Driving“. In: *Journal of Physics: Conference Series*. Bd. 2093. 1. IOP Publishing. 2021, S. 012032 (siehe S. 3, 5, 157).

[Watk92] Watkins, C. und Dayan, P.: „Q-Learning“. In: *Machine learning* 8 (1992), S. 279–292 (siehe S. 35).

[Winn11] Winner, H.; Hakuli, S. und Wolf, G.: Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort. Springer-Verlag, 2011 (siehe S. 9).

[Wrig19] Wright, R. S.: „Should Accounting Be the Language of Business?“ In: *Research-Technology Management* 62.4 (2019), S. 53–55 (siehe S. 155).

[Wu11] Wu, C.; Du, L.; Wang, D.; Wang, L.; Zhao, L. und Wang, P.: „A Novel Surface Acoustic Wave-Based Biosensor for Highly Sensitive Functional Assays of Olfactory Receptors“. In: *Biochemical and Biophysical Research Communications* 407.1 (2011), S. 18–22 (siehe S. 3).

[Wu22] Wu, X.; Peng, L.; Yang, H.; Xie, L.; Huang, C.; Deng, C.; Liu, H. und Cai, D.: „Sparse Fuse Dense: Towards High Quality 3d Detection with Depth Completion“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, S. 5418–5427 (siehe S. 101, 166, 170, 227).

[Wu23] Wu, H.; Wen, C.; Shi, S.; Li, X. und Wang, C.: „Virtual Sparse Convolution for Multimodal 3D Object Detection“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, S. 21653–21662 (siehe S. 166, 170).

[Yeon21] Yeong, D. J.; Velasco-Hernandez, G.; Barry, J. und Walsh, J.: „Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review“. In: *Sensors* 21.6 (2021), S. 2140 (siehe S. 15–17, 159).

[Yin21] Yin, T.; Zhou, X. und Krähenbühl, P.: „Multimodal Virtual Point 3d Detection“. In: *Advances in Neural Information Processing Systems* 34 (2021), S. 16494–16507 (siehe S. 155).

[Youn81] Young, A. T.: „Rayleigh Scattering“. In: *Applied optics* 20.4 (1981), S. 533–535 (siehe S. 9).

[Yu20] Yu, F.; Chen, H.; Wang, X.; Xian, W.; Chen, Y.; Liu, F.; Madhavan, V. und Darrell, T.: „Bdd100k: A Diverse Driving Dataset for Heterogeneous Multitask Learning“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, S. 2636–2645 (siehe S. 5, 19).

[Zait15] Zaitsev, M.; Maclaren, J. und Herbst, M.: „Motion Artifacts in MRI: A Complex Problem with many Partial Solutions“. In: *Journal of Magnetic Resonance Imaging* 42.4 (2015), S. 887–901 (siehe S. 7).

[Zang19] Zang, S.; Ding, M.; Smith, D.; Tyler, P.; Rakotoarivelo, T. und Kaafar, M. A.: „The Impact of Adverse Weather Conditions on Autonomous Vehicles: How Rain, Snow, Fog, and Hail Affect the Performance of a Self-Driving car“. In: *IEEE vehicular Technology magazine* 14.2 (2019), S. 103–111 (siehe S. 9).

[Zhan15] Zhang, R.; Candra, S. A.; Vetter, K. und Zakhor, A.: „Sensor Fusion for Semantic Segmentation of Urban Scenes“. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, S. 1850–1857 (siehe S. 5).

[Zhao17] Zhao, H.; Shi, J.; Qi, X.; Wang, X. und Jia, J.: „Pyramid Scene Parsing Network“. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, S. 2881–2890 (siehe S. 165).

[Zhao19] Zhao, Z.-Q.; Zheng, P.; Xu, S.-T. und Wu, X.: „Object Detection with Deep Learning: A Review“. In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), S. 3212–3232 (siehe S. 164).

[Zhen22] Zheng, T.; Huang, Y.; Liu, Y.; Tang, W.; Yang, Z.; Cai, D. und He, X.: „CLRNET: Cross Layer Refinement Network for Lane Detection“. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, S. 898–907 (siehe S. 3).

[Zhou19] Zhou, F.; Hu, Y. und Shen, X.: „MSANet: Multimodal Self-Augmentation and Adversarial Network for RGB-D Object Recognition“. In: *The Visual Computer* 35 (2019), S. 1583–1594 (siehe S. 166).

Eigene Publikationen

Dieser Abschnitt enthält eine vollständige Liste der eigenen Publikationen, die während der Promotionszeit entstanden sind. Die Publikationen [3], [5] adressieren das Thema Optimierung der Sensoranordnung und führen den GAOS- bzw. DLOS-Algorithmus ein. In den Publikationen [6], [7] werden die Ergebnisse der Anwendung von KISA auf Complex-YOLO mit einer Low-Level-Fusionsstrategie bei der Fusion von LiDAR- und RADAR-Daten aus dem Astyx-Datensatz sowie auf U-Net mit Low- und High-Level-Fusionsstrategien auf einem synthetischen Datensatz und MRT-Daten aus dem BraTS-Datensatz vorgestellt. Der synthetische Datensatz wird in der Publikation [8] beschrieben und vergleicht Deep Learning Methoden mit nicht Deep Learning Methoden für die semantische Segmentierung. Die Publikation [2] adressiert das Thema Metriken zum Vergleich von Punktwolken. In der Publikation [1] wird beschrieben, wie mithilfe eines GANs LiDAR-Sensordaten unterschiedlicher Modalität übertragen werden können. Die Publikation [4] beschreibt, wie MEMS-LiDAR-Sensoren in CARLA simuliert werden können. Die Publikation [9] betrachtet Evaluationsmetriken für die 6D-Posenschätzung.

[1] Berens, F.; Knapp, Y.; Reischl, M. und Elser, S.: „Transforming LiDAR Point Cloud Characteristics between different Datasets using Image-to-Image Translation“. In: *Proceedings-30. Workshop Computational Intelligence: Berlin, 26.-27. November 2020*. Bd. 26. KIT Scientific Publishing. 2020, S. 73.

[2] Berens, F.; Elser, S. und Reischl, M.: „Evaluation of Four Point Cloud Similarity Measures for the use in Autonomous Driving“. In: *at-Automatisierungstechnik* 69.6 (2021), S. 503–514.

[3] Berens, F.; Elser, S. und Reischl, M.: „Genetic Algorithm for the Optimal LiDAR Sensor Configuration on a Vehicle“. In: *IEEE Sensors journal* 22.3 (2021), S. 2735–2743.

[4] Berens, F.; Elser, S. und Reischl, M.: „MEMS LiDAR Sensor Simulation for Autonomous Driving: A Novel Framework Using Open-source Tools“. In: *ISR Europe 2023; 56th International Symposium on Robotics*. VDE. 2023, S. 298–303.

[5] Berens, F.; Ambs, J.; Elser, S. und Reischl, M.: „A Novel Approach to Light Detection and Ranging Sensor Placement for Autonomous Driving Vehicles Using Deep Deterministic Policy Gradient Algorithm“. In: *SAE International Journal of Connected and Automated Vehicles* 7.12-07-03-0019 (2024).

[6] Berens, F.; Koschinski, Y.; Badami, M. K.; Geimer, M.; Elser, S. und Reischl, M.: „Adaptive Training for Robust Object Detection in Autonomous Driving Environments“. In: *IEEE Transactions on Intelligent Vehicles* (2024), S. 1–15.

[7] Berens, F.; Münke, F. R.; Elser, S. und Reischl, M.: „Improving the Robustness of Low-level Fusion for Semantic Segmentation“. In: *eingereicht bei Künstliche Intelligenz* (2024).

[8] Münke, F. R.; Schütze, J.; Berens, F. und Reischl, M.: „A Review of Adaptable Conventional Image Processing Pipelines and Deep Learning on limited Datasets“. In: *Machine Vision and Applications* 35.2 (2024), S. 25.

[9] Niedermaier, Tobias; Berens, Felix; Reischl, Markus und Elser, Stefan: „A novel metric for 6D pose estimation: Addressing errors and false detections for more reliable evaluation“. In: *at-Automatisierungstechnik* 73.2 (2025), S. 125–135.

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AP	Average Precision
AR	Average Ratio
BEGAN	Boundary Equilibrium Generative Adversial Network
BEV	Bird's Eye View
BraTS	Brain Tumor Segmentation
CD	Chamfer Distanz
cGAN	Conditional Generative Adversial Network
CLOCs	Camera-LiDAR Object Candidates
CNN	Convolutional Neural Network
DLOS	Deep Learning-Optimierung für Sensorplatzierung
FN	False Negative
FP	False Positive
GAOS	Genetische Algorithmus-Optimierung für Sensorplatzierung
HOG	Histogram of Oriented Gradients
ICP	Iterative Closest Point Algorithm
IMU	Interne Trägheitsmessungseinheit
IoU	Intersection-over-Union
KISA	Konzept zur Integration von Störungs-Adaptivität
LiDAR	Ligth Detection and Ranging
mAP	mean Average Precision

MEMS	Micro-Electro-Mechanical Systems
MOR	Meteorological Optical Range
MRT	Magnetresonanztomographie
NIR	Near-Infrared
NMS	Non-Maximum Suppression
OPA	Optical Phased Array
RADAR	Radio Detection and Ranging
R-CNN	Recurrent Convolutional Neural Network
ROI	Region of Interest
SFD	Sparse Fused Dense
SIFT	Scale-Invariant Feature Transform
SSD	Single Shot Multibox Detector
TOF	Time-Of-Flight
TP	True Positive
TWISe	Twin-Surface Extrapolation
UNIT	Unsupervised Image-to-Image Translation
YOLO	You Only Look Once

A Vergleich von LiDAR und RADAR-Sensor

Ein Vorteil von LiDAR- und RADAR- Sensoren gegenüber Kameras ist die Möglichkeit der genauen Entfernungsmessung. Mithilfe einer Vielzahl solcher Messungen ermöglichen LiDAR- und RADAR-Sensoren eine räumliche Rekonstruktion der Umgebung [Gao19, Wang20b, Royo19], die zur Objektdetektion verwendet werden können [Yin21]. Aufgrund der kürzeren Wellenlänge sind LiDAR-Messungen besonders genau, aber auch stärker von Umwelteinflüssen wie Nebel, Regen oder Schnee beeinflusst als RADAR-Messungen, die besonders wetterunabhängig sind (siehe Kapitel 1.3).

Ein weiterer wichtiger Punkt bei der Auswahl der Sensoren für ein autonomes Fahrzeug sind die Kosten, die Größe und die Beständigkeit der Sensoren. Da Kamera (seit den 1950er im Konzeptauto Buick Centurion [Wrig19]) und RADAR-Sensoren (seit den 1970er [Mein98, Wald21]) schon seit langem in Fahrzeugen eingesetzt werden, gibt bereits Produkte, deren Abmessungen und Beständigkeit an die Bedingungen in einem Fahrzeug angepasst sind. Des weiteren ist der Stückkosten für die Sensoren mit der Zeit gesunken. Neu entwickelte Imaging RADAR Sensoren, die die Umgebung auch in vertikaler Richtung abtasten und eine höhere Auflösung haben, können jedoch noch einen hohen Preis haben [Wald21]. Die Entwicklung von in Serie produzierte LiDAR-Sensoren ist relativ neu, weshalb LiDAR-Sensoren (Stückpreis für einen Ouster OS0 12.050 €[1]) noch verglichen mit RADAR-Sensoren (Stückpreis für einen MMWCAS-RF-EVM 1.099 $[2]) teuer sind. Die am weitesten verbreiteten

[1] https://www.reichelt.de/ouster-multi-layer-3d-lidar-os0-rev7-64-layer-90-0-100-m-os0-64-rev7-p363166.html, aufgerufen am 25. Juli 2024

[2] https://www.ti.com/tool/MMWCAS-RF-EVM, aufgerufen am 25. Juli 2024

LiDAR-Sensoren gehören zu den sogenannten optomechanischen Scannern. Bei diesem LiDAR-Typ wird der Laserstrahl durch optische Komponenten über rotierende Spiegel so gelenkt, dass der Laserstrahl bis zu 360° des horizontalen Sichtfeldes Punkt für Punkt abtastet [Raj20]. Diese mechanischen Komponenten sind anfällig für Verschleiß und Vibrationen, was die Qualität der Daten beeinflusst und die Lebensdauer des Sensors begrenzt [Schl22, Liu21, Royo19].

In den letzten Jahren wurden verschiedene innovative LiDAR-Technologien entwickelt, die möglicherweise die Zukunft das autonomen Fahren prägen könnten:

Einer dieser neuen Sensortypen ist der Micro-Electro-Mechanical Systems (MEMS)-LiDAR. Diese Sensoren implementieren mikroskopisch kleine, bewegliche Spiegel, die dazu dienen, Laserstrahlen präzise und schnell zu lenken. Der Hauptvorteil von MEMS-LiDAR liegt in seiner kompakten Bauweise, die eine einfache Integration in verschiedene Geräte ermöglicht, sowie in der hohen räumlichen Auflösung, die für detaillierte Umgebungserfassungen essentiell ist. Weiterhin sind MEMS-LiDAR-Sensoren aufgrund ihrer Robustheit und Langlebigkeit für den Einsatz unter rauen Umweltbedingungen prädestiniert. Ihre geringe Anfälligkeit für mechanische Abnutzung, resultierend aus dem minimalen Bewegungsumfang der MEMS-Komponenten, führt zu einer erhöhten Zuverlässigkeit im Feld [Wang20b, Royo19, Raj20, Rori21].

Ein weiterer zukunftsweisender LiDAR Sensortyp ist der Optical Phased Array (OPA)-LiDAR. Im Gegensatz zu herkömmlichen LiDAR-Systemen, die auf mechanisch bewegte Teile angewiesen sind, nutzt der OPA-LiDAR ein Array von Phasenschiebern, um die Wellenfront des Lasers elektrooptisch zu manipulieren. Diese Technik ermöglicht eine äußerst schnelle und präzise Steuerung der Richtung des Laserstrahls. Der größte Vorteil dieser Technologie liegt in ihrer Fähigkeit, ohne bewegliche Teile zu funktionieren, was die mechanische Komplexität reduziert und die Haltbarkeit und Zuverlässigkeit des Systems verbessert. Diese Eigenschaften sind besonders vorteilhaft für Anwendungen, die eine robuste und langlebige Sensorik erfordern, wie beispielsweise im Bereich des autonomen Fahrens. Durch die schnelle Strahlsteuerung des OPA-LiDAR können hochdynamische Umgebungen effektiv und in Echtzeit

erfasst werden, was essentiell für die Sicherheit und Effizienz autonomer Fahrzeuge ist. Die Technologie bietet zudem eine hohe Auflösung und kann auch auf große Entfernungen präzise Messdaten liefern, was sie für eine Vielzahl von Anwendungen attraktiv macht, darunter auch die Verkehrsüberwachung und städtische Infrastrukturprojekte.

MEMS-, oder OPA-LiDAR-Sensoren werden voraussichtlich eine längere Lebensdauer haben, sind aber zum Zeitpunkt dieser Arbeit noch nicht weit genug entwickelt, um im autonomen Fahren eingesetzt zu werden [Royo19, Raj20]. Auch sind keine öffentliche Datensätze für diese Typen von LiDAR-Sensoren bekannt.

Tabelle A.1: Vor- und Nachteile möglicher Sensoren für die Umfelderkennung beim autonomen Fahren[Shee21, Saka18, Reic21, Wang21, Igna22, Varg21].
+: Sensor erfüllt die Bedingung vollständig
o: Sensor erfüllt die Bedingung teilweise.
-: Sensor erfüllt die Bedingung nicht .

	Kamera	LiDAR	RADAR
Reichweite	o	o	+
Datendichte	+	o	-
Distanzmessung	o	+	+
Geschwindigkeitsmessung	-	o	+
Farbkontrast	+	-	-
Dunkelheit	-	+	+
Gegenlicht	-	o	+
Nebel	-	-	+
Schnee	-	o	+
Regen	-	o	+
Größe	+	-	+
Kosten	+	-	o

In Tabelle A.1 sind die Vor- und Nachteile von Kamera, LiDAR- und RADAR-Sensor dargestellt. Zusammenfassend lässt sich sagen, dass nur die Fusion aller dieser Sensoren es einem autonomen Fahrzeug ermöglicht, sicher zu operieren. Die Kamera ist unerlässlich, um Verkehrszeichen zu erkennen, andere

Fahrzeuge, Fußgänger und Hindernisse zu identifizieren sowie Straßenmarkierungen zu verfolgen. Auch werden derzeit sowohl LiDAR-Sensoren werden benötigt, als auch RADAR Sensoren benötigt, da nur LiDAR-Sensoren eine Datendichte bei der Distanzmessung liefern, die es ermöglicht, auch kleinere Objekte wie Fußgänger zu erkennen [Royo19, Raj20]. Wohingegen RADAR Sensoren besonders wegen ihrer Robustheit gegenüber Wettereinflüsse benötigt werden [Shee21].

B Sensor-Kalibrierung

Für die Kalibrierung werden Kalibrierungsobjekte benötigt und müssen damit schon während der Aufnahme vorhanden sein. Die Kalibrierung wird deshalb bereits bei der Aufnahme der Datensätzen durchgeführt und die entsprechenden Kalibrierungsparameter werden in Datensätzen angegeben [Geig12a].

Bei der Sensor-Kalibrierung wird zwischen intrinsischer Kalibrierung und extrinsischer Kalibrierung unterschieden [Yeon21].

B.1 Intrinsische Sensor-Kalibrierung

Die intrinsische Sensor-Kalibrierung ist ein Verfahren zur Bestimmung der inneren Parameter eines Sensors. Es geht darum, die Fehler, die bei der Messung von Bildern oder anderen Daten durch den Sensor entstehen, zu minimieren, um genaue Informationen über die reale Welt zu erhalten. Dies wird durch die Messung der charakteristischen Merkmale von Objekten in der realen Welt erreicht, die vom Sensor erfasst werden, und durch den Vergleich dieser Messungen mit den erwarteten Werten aus einer mathematischen Modellierung [Heik97, Malm03, Geig12b, Berg17, Yeon21]. Die intrinsische Kalibrierung ist eine wichtige Voraussetzung für die extrinsische Kalibrierung.

Die intrinsische Parameter für die Kamera lassen sich in der folgenden Matrix zusammenfassen:

$$\mathbf{K} := \begin{pmatrix} f \cdot \kappa_x & -\kappa_x / \tan(\theta) & s_u \\ 0 & f \cdot \kappa_y / \sin(\theta) & s_v \\ 0 & 0 & 1 \end{pmatrix},$$

wobei f die physikalische Brennweite der Kamera ist, κ_x, κ_y sind Skalierungsfaktoren für die Höhe beziehungsweise Breite eines Pixels, θ ist der Scherungswinkel der Verzerrungen aufgrund der Nicht-Orthogonalität der Achsen des Bildsensors berücksichtigt und s_u, s_v ist der optischer Mittelpunkt der den Schnittpunkt der optischen Achse mit dem Bildsensor darstellt. Häufig wird das vereinfachte Lochkameramodell angenommen, bei dem $\theta = 90°$ und $\kappa_x = \kappa_y = 1$ ist [Tsai87, Geig12b]. Für den KITTI Datensatz wurde als Kalibrierungsobjekt zur Bestimmung dieser Werte ein Schachbrettmuster verwendet [Geig12b].

Für LiDAR oder RADAR Sensoren werden bei den Datensätzen keine vergleichbaren intrinsischen Parameter angegeben.

B.2 Extrinsische Sensor-Kalibrierung

Bei der extrinsische Sensor-Kalibrierung wird die Position und Ausrichtung der Sensoren in realen Koordinaten bestimmt, indem sie die relativen Positionen bekannter Merkmale vergleicht, wie sie von den Sensoren erfasst werden. Diese Parameter können in einer Orientierungsmatrix $\mathbf{M} := (\mathbf{R},\mathbf{t}) \in \mathbb{R}^{3\times4}$ zusammengefasst werden. Dabei ist $\mathbf{R} \in \mathbb{R}^{3\times3}$ die Rotationsmatrix und $\mathbf{t} \in \mathbb{R}^3$ der Translationsvektor:

$$\mathbf{R} := \begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{pmatrix} \qquad \mathbf{t} := \begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix}$$

Mit der Orientierungsmatrix lassen sich alle Sensordaten in ein gemeinsames Koordinatensystem überführen [Geig12b, Geig12a].

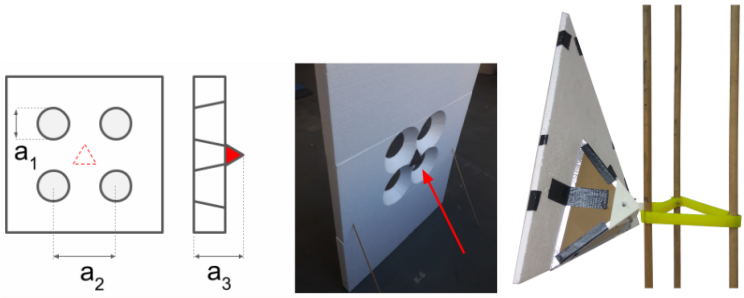


Abbildung B.1: Darstellung von RADAR/LiDAR/Kamera Kalibrierungsobjekte. Von links nach rechts: Zeichnung der Vorderansicht, Seitenansicht und zwei Bilder von Kalibriungsobjekts welche mit Eckreflektoren und Styroporplatten arbeiten [Domh19, Peršl7, Peršl9].

Um diese Werte zu bestimmen, werden Kalibrierungsobjekte benötigt, welche von allen Sensoren erkannt werden können. Für die Kalibrierung von Kamera, LiDAR und RADAR Sensoren wird als Kalibrierungsobjekt eine Konstruktion bestehend aus einem Styropordreieck mit Schachbrettmuster und einem Dreiflächen-Winkelreflektor vorgeschlagen [Peršl7, Peršl9] (siehe Abb. B.1). Das Styropordreieck ist aufgrund seines hohen Luftanteils nahezu unsichtbar für den RADAR Sensor, ermöglicht jedoch eine präzise Lokalisierung innerhalb der Punktwolke. Der Dreiflächen-Winkelreflektor reflektiert alle einfallenden RADAR-Strahlen in die gleiche Richtung, was zu einem hohen und orientierungsunabhängigen Radarquerschnitt führt und damit präzise in der RADAR-Punktwolke detektiert werden kann. Alternativ wird statt einem Styropordreieck auch eine Styroporplatte mit vier Löchern und einem Dreiflächen-Winkelreflektor vorgeschlagen [Domh19] (siehe Abb. B.1). Die Orientierungsmatrix $\mathbf{M}$ wird mittels geeigneter Optimierungsverfahren wie dem Iterative Closest Point Algorithm (ICP) bestimmt [Besl92].

Liegen Punktwolken zweier Sensoren vor (Φ^1 und Φ^2) und die Orientierungsmatrix $\mathbf{M}$ ist bekannt, so kann man diese Punktwolken zu einer gemeinsamen Punktwolke kombinieren. Das Koordinatensystem eines Sensors wird als Zielkoordinatensystem betrachtet, in welches die Punktwolke des anderen Sensors transformiert wird. O.B.d.A. wird hier das Koordinatensystem, welches zu Punktwolke Φ^2 gehört, als Zielkoordinatensystem betrachtet und $\boldsymbol{M}$ ist die

entsprechende Transformationsmatrix zwischen den Koordinatensystemen. Um Punkte $\Phi^1[i]$ in das Koordinatensystem Φ^2 zu transformieren, werden die Koordinaten $\Phi^1[i]$ zu homogenen Koordinaten $\Phi^1_{hom}[i]$ erweitert, indem eine vierte Koordinate, die auf 1 gesetzt wird, hinzugefügt wird:

$$\Phi^1_{hom}[i] = (\Phi^1[i,0],\Phi^1[i,1],\Phi^1[i,2],1).$$

Um die homogenen Koordinaten $\Phi^1_{hom}[i]$ in das Koordinatensystem Φ^2 zu transformieren, wird die Orientierungsmatrix $\mathbf{M}$ mit dem Punkt multiplizieren:

$$\Phi^1_{trans}[i] = \mathbf{M} \cdot \Phi^1_{hom}[i].$$

Anschließend wird die hinzugefügte vierte Koordinate wieder aus den transformierten Punkten gestrichen und die zusätzliche Information (Intensität bei LiDAR Punkten, Magnitude oder relative radiale Geschwindigkeit bei RADAR Punkten) hinzugefügt:

$$\Phi^1_{trans} = (\Phi^1_{trans}[i,0],\Phi^1_{trans}[i,1],\Phi^1_{trans}[i,2],\Phi^1[i,3]).$$

Somit sind die Punktwolken Φ^1_{trans} und Φ^2 in einem gemeinsamen Koordinatensystem. Danach werden die zusätzlichen Informationen beider Punktwolken $\Phi^1_{trans}[i,3]$ und $\Phi^2[i,3]$ normiert:

$$\Phi^1_{trans}[i,3] = \frac{\Phi^1_{trans}[i,3] - a_{\min,.}}{a_{\max,.} - a_{\min,.}}$$

$$\Phi^2[i,3] = \frac{\Phi^2[i,3] - a_{\min,.}}{a_{\max,.} - a_{\min,.}}.$$

Im letzten Schritt werden die Punktwolken Φ^1_{trans} und Φ^2 vereinigt:

$$\Phi = \Phi^1_{trans} \cup \Phi^2.$$

C Beschreibung vom Stand der Objektdetektion

Traditionelle Methoden der Objekterkennung und Merkmalsextraktion sind beispielsweise Scale-Invariant Feature Transform (SIFT) [Lowe99], Viola-Jones-Algorithmus [Viol01] oder , Histogram of Oriented Gradients (HOG) [Dala05]. SIFT identifiziert und beschreibt Merkmale in einem Bild auf eine Weise, die gegenüber Skalierung und Rotation invariant sind. Für jedes Merkmale wird ein Deskriptor aus den lokalen Gradienten berechnet. Nachdem die Merkmale eines Bildes mit SIFT extrahiert wurden, müssen diese mit einem Satz von bekannten Merkmalen eines Objekts, das erkannt werden soll, abgeglichen werden. Dies erfordert eine Art von Matching-Algorithmus, dafür werden beispielsweise der Brute-Force-Matcher oder FLANN-Based Matcher verwendet, um ähnliche Merkmale zwischen den Bildern zu finden. Der Brute-Force-Matcher vergleicht jedes Merkmal des einen Bildes mit jedem Merkmal des anderen Bildes, um das beste Match zu finden. Anstatt jedes Merkmal mit jedem anderen zu vergleichen, verwendet FLANN heuristische Algorithmen um Näherungslösungen zu finden [Suju17, Vija19].

Während SIFT sich auf die Beschreibung lokaler Bildmerkmale konzentrieren, die für eine Vielzahl von Objekterkennungsaufgaben genutzt werden können, ist der Viola-Jones-Algorithmus speziell für die Gesichtserkennung entwickelt worden. Der Viola-Jones-Algorithmus nutzt sogenannte Haar-ähnliche Merkmale, eine kaskadierende Klassifikation und eine Bildpyramide, um schnell Gesichter in Bildern zu identifizieren. Der Algorithmus besteht aus drei Hauptkomponenten: Der Integralbildmethode zur schnellen Merkmalsberechnung, einem Adaboost-basierten Verfahren zur Auswahl weniger, aber effektiver

Merkmale, und einer Kaskade von schwachen Klassifikatoren, die schrittweise komplexer werden, um schnell Nicht-Gesichtsbereiche zu verwerfen und Fehlalarme zu minimieren [Viol01].

Damit sind SIFT zusammen mit einem Matcher und der Viola-Jones-Algorithmus Beispiele für Objekdetektoren die ihre Merkmalsextraktions- und Erkennungsmethoden auf sorgfältig ausgearbeiteten mathematischen Modellen und Heuristiken basieren, nicht auf einem Trainingsprozess mit maschinellem Lernen auf Basis von Datensätzen. Der HOG-Deskriptor funktioniert, indem er das Bild in kleine, verbundene Zellen teilt, für jede Zelle ein Histogramm von Gradientenrichtungen oder Kantenorientierungen erstellt und diese Histogramme dann über die Zellen aggregiert, um einen Merkmals-Deskriptor zu bilden. Damit nutzt auch der HOG-Deskriptor auf vordefinierte Prozesse, um relevante Informationen aus dem Bild zu extrahieren, ohne dass ein explizites Lernverfahren durchgeführt wird. Der HOG-Deskriptor wird oft mit Support Vector Machines (SVM) kombiniert. Dieser Klassifikator wird auf einem Satz von Trainingsdaten trainiert, um zu lernen, wie die HOG-Merkmale zur Unterscheidung zwischen verschiedenen Klassen von Objekten verwendet werden können [Dala05, Llor13].

Die Leistung traditioneller Methoden der Objekterkennung stagniert oder fällt mit wachsender Aufgabenkomplexität [Zhao19]. In [Münk24] wurde gezeigt, dass Deep-Learning-Modelle traditionelle Bildverarbeitungstechniken mit wachsender Trainigsdatenbasis übertreffen. Insbesondere die Entwicklung von Convolutional Neural Networks (CNNs) ermöglicht es, komplexe Merkmale zu erlernen, ohne sie manuell entwerfen zu müssen, was zu einer deutlichen Leistungssteigerung führt. Die Entwicklung von Recurrent Convolutional Neural Network (R-CNN) [Girs14] markiert einen Durchbruch in der Objekterkennung [Zhao19]. R-CNN generiert zunächst etwa 2000 Regionenvorschläge pro Bild und nutzt dann CNNs, um eine hochdimensionale Merkmalsdarstellung für jede Region zu extrahieren. Diese Merkmale werden dann zur Klassifizierung und Lokalisierung von Objekten verwendet. Nachfolgemodelle wie Fast R-CNN [Girs15] und Faster R-CNN [Ren15] verbessern die Effizienz und Genauigkeit, wobei Faster R-CNN eine Region Proposal Network verwendet, um Vorschläge effizienter zu generieren. R-CNN und seine Varianten sind Beispiele

für Zwei-Schritt-Frameworks in der Objekterkennung. You Only Look Once (YOLO) [Redm16] und Single Shot Multibox Detector (SSD) [Liu16] repräsentieren einen Paradigmenwechsel zu Ein-Schritt-Frameworks, die direkt von Bildpixeln zu Bounding-Box-Koordinaten und Klassenwahrscheinlichkeiten übergehen, was die Zeitkosten reduzierte und sich für Echtzeitanwendungen eignen. Der YOLO-Algorithmus teilt das Eingabebild in eine Rasterstruktur auf. Jede Zelle des Rasters ist verantwortlich für die Vorhersage von Objekten, deren Mittelpunkt in dieser Zelle liegt. Jede Zelle prognostiziert mehrere Bounding Boxes und Vertrauenswerte, die angeben, wie sicher das System ist, dass ein Rahmen ein Objekt enthält. Zusätzlich werden Klassenwahrscheinlichkeiten für jede Box berechnet, die angeben, zu welcher Klasse das erkannte Objekt wahrscheinlich gehört. Für eine detailliertere Beschreibung des YOLO Algorithmus sei auf den Appendix H verwiesen.

Auch basieren viele Modelle für die semantische Segmentierung auf CNNs, wie SegNet [Badr17] und U-Net [Ronn15]. SegNet verwendet CNNs, um durch eine Encoder-Decoder-Architektur Bilder zu segmentieren, wobei der Encoder Merkmale extrahiert und der Decoder diese Merkmale nutzt, um eine pixelweise Klassifikation des Bildes zu erzeugen. U-Net nutzt eine symmetrische Encoder-Decoder-Architektur, die mit Skip-Verbindungen angereichert ist. Für eine detailliert Beschreibung von U-Net sei auf das Kapitel F.2 verwiesen. DeepLab [Chen17] integrierte Dilated Convolution und Conditional Random Fields, um sowohl die Fähigkeit zur Erfassung von Kontextinformationen auf verschiedenen Skalen als auch die Feinabstimmung von Segmentierungsgrenzen zu verbessern. Eine weitere Methode ist das Pyramid Scene Parsing Network [Zhao17], das eine pyramidale Pooling-Architektur verwendet, um globale Kontextinformationen effektiver zu aggregieren und so eine detaillierte und präzise Segmentierung zu ermöglichen.

Auch die Objekterkennung in 3D Sensordaten hat durch die Einführung von CNNs an Leistung gewonnen. Nach der Art wie die 3D Daten verarbeitet werden, lassen sich die Detektoren in drei Gruppen aufteilen:

- Projektionsbasierte Ansätze: Diese Methode wandelt 3D-Daten in eine 2D-Darstellung um, die dann mit herkömmlichen 2D-Bildverarbeitungstechniken analysiert wird. So kann entweder ein

Tiefenbild erzeugt werden, welches eine ähnlichen Sichtpunkt wie eine Kamera hat. Dieser Ansatz wird bei RGB-D-Daten verwendet, bei denen die Tiefeninformation als zusätzlicher Kanal neben den herkömmlichen RGB-Kanälen betrachtet wird. Alternativ können die Punktwolken in ein BEV Bild projiziert werden. Dieser Ansatz profitiert von der Entwicklung im Bereich der 2D-Bildverarbeitungstechnologie. Beispiele für diesen Ansatz sind Complex-YOLO [Simo18] und MSANet [Zhou19].

- Direkte Punktwolkenverarbeitung: Dieser Ansatz verarbeitet 3D-Daten direkt in ihrer Form als Punktwolken, ohne sie in eine andere Darstellungsform zu überführen. Modelle wie PointNet [Qi17] und PointNet++ [Qi17] sind speziell für die Verarbeitung von Punktwolken konzipiert. Sie können Merkmale aus der ungeordneten und unstrukturierten Natur der Punktwolken extrahieren. So verwendet beispielsweise PointRCNN [Shi19] PointNet++ als Backbone um punktweise Merkmale aus der Punktwolke zu erzeugen.
- Voxelbasierte Ansätze: Bei dieser Methode werden 3D-Daten in ein 3D-Voxelgitter umgewandelt. Auf diese Voxelgitter können dann 3D-CNNs angewendet werden, die in der Lage sind, räumliche Strukturen und Beziehungen direkt aus den 3D-Daten zu erfassen. Voxel-basierte Ansätze können eine hohe Genauigkeit in der Objekterkennung erreichen, sind aber oft rechenintensiv, insbesondere bei hohen Auflösungen. Beispiele für Objekdetektoren, welche die Punktwolke in ein 3D-Voxelgitter transformieren sind: VirConv-T [Wu23], SFD [Wu22] oder UberATG-MMF [Lian19].

D Evaluationsmetriken

Evaluationsmetriken für Objektdetektoren sind quantitative Maße, die dazu verwendet werden, die Leistungsfähigkeit und Genauigkeit von Algorithmen zur Objekterkennung zu bewerten. Diese Metriken dienen dazu, zu beurteilen, wie gut ein Objektdetektionsmodell in der Lage ist, Objekte innerhalb einer Dateninstanz korrekt zu identifizieren und zu lokalisieren.

Definition D.1. *Eine Evaluationsmetrik f ist eine Abbildung, welche einen Objektdetektor O und einen Datenmenge D erhält und dies auf einen metrischen Wert in $\mathbb{R}$ abbildet.*

Bei Objektdetektoren für 2D oder 3D Bounding Boxen wird für die Bestimmung, ob ein Objekt richtig ist, die sogenannte IoU berechnet, auch als Jaccard Index bekannt.

Definition D.2. *Die IoU misst das Verhältnis der Überlappung zwischen der Vorhersage (A) und dem tatsächlichen Objekt (B):*

$$IoU = \frac{|A \cap B|}{|A \cup B|}.$$

Die IoU kann sowohl für 2D Bounding Boxen verwendet werden, als auch für 3D Bounding Boxen. Wenn der IoU Wert größer als ein zuvor festgelegter Schwellenwert ist und die Klassen übereinstimmen, wird die Bounding Box als richtig positiv (True Positive (TP)) angesehen. Vorhergesagte Bounding Boxen, welche keiner wahren Bounding Box entsprechen, werden als falsch positiv (False Positive (FP)) bezeichnet und solche wahren Bounding Boxen, die nicht detektiert werden, als falsch negativ (False Negative (FN)).

Definition D.3. *Mit diesen Werten werden diese Evaluierungsmaße definieren:*

- *Genauigkeit (Precision): misst das Verhältnis zwischen den richtig Positiven (TP) Objekten und der Gesamtanzahl an vorhergesagten Objekten ($TP + FP$):*

$$\frac{TP}{TP + FP}.$$

  Ein Nachteil dieses Maß ist, dass die Genauigkeit nicht die falsch Negativen Ergebnisse in die Berechnung mit einbezieht.

- *Sensitivität (Recall): misst das Verhältnis zwischen den richtig Positiven (TP) und allen wahren Objekten ($TP + FN$):*

$$\frac{TP}{TP + FN}.$$

  Ein Nachteil dieses Maß ist, dass die Sensitivität nicht die falsch positiven einbezieht.

Die beiden Maße haben eine inverse Beziehung, dies bedeutet, dass die eine erhöht werden kann, wodurch die andere verkleinert wird. Eine Möglichkeit beide Maße zu kombinieren ist das F_1-Maß.

Definition D.4. *Das F_1-Maß ist der harmonische Mittelwert zwischen Genauigkeit und Sensitivität:*

$$F_1 = \frac{2TP}{2TP + FP + FN}.$$

Jedoch ist das gängigste Evaluierungsmaß für den Vergleich von Objektdetektoren von Bounding Boxen die sogenannte AP [Jana20], welche beide Maße kombiniert. Die AP ist auch das Evaluierungsmaß, welches in dieser Arbeit Verwendung findet, um Bounding Box Detektoren zu evaluieren. Die AP gibt die Fläche unter der Precison-Recall Kurve an und ist abhängig von dem gewählten IoU Schwellwert. Dieser Schwellwert bestimmt, ab welchem Grad der Überlappung zwischen vorhergesagten und tatsächlichen Bounding Boxen

eine Vorhersage als korrekt betrachtet wird. Ein häufig verwendetes Maß ist die AP bei einem IoU Schwellwert von 50%, üblicherweise als AP_{50} notiert. Hierfür werden alle vorhergesagten Bounding Boxen einer Klasse nach dem jeweiligen Konfidenzwert in absteigender Reihenfolge sortiert. Nach dieser Reihenfolge wird die Genauigkeit $p_1, \dots, p_N$ und Sensitivität $r_1, \dots, r_N$ berechnet. Während für die Werte der Sensitivität gilt: $r_1 \leq r_2 \leq \cdots \leq r_N$, gilt dies nicht für die Werte der Genauigkeit. Dies macht die Berechnung der Fläche unter der Precisoin-Recall Kurve schwerer, weshalb Approximationen verwendet werden, wobei die Genauigkeit interpoliert wird:

$$p_{\text{interp}}(r) = \max_{\tilde{r}:\tilde{r}\geq r}(\tilde{r}).$$

Für die Pascal Visual Object Classes Challenge und auch für die Evaluierung auf dem KITTI Datensatz wurden zur Approximation der AP die Elf-Punkte Approximation [Salt83] verwendet.

Definition D.5. *Die Elf-Punkte Average Precision AP^{11} wird über die mittlere Genauigkeit von elf gleichmäßig verteilte Sensitivitäten $\{0, 0.1, \dots, 1\}$ berechnet:*

$$AP^{11} := \sum_{r\in\{0,0.1,\dots,1\}} p_{interp}(r).$$

Für die offizielle Evaluierung für das Leaderboard auf dem KITTI Datensatz wird ab dem 08.10.2019 mit 40 statt 11 Punkten approximiert [Geig23] und beim Pascal Visual Object Classes Challenge werden 2009 alle Sensitivitäten verwendet [Ever15].

Definition D.6. *Die Average Precision AP wird über die mittlere Genauigkeit aller N Sensitivitäten berechnet:*

$$AP = \sum_{n=1}^{N} (r_{n+1} - r_n) p_{interp}(r_{n+1}).$$

Für mehrklassen Objektdetektoren wird die sogenannte mean Average Precision (mAP) berechnet, welche die mittlere Average Precision der Klassen angibt. Auch hier wird ein IoU Schwellwert von 50% durch mAP_{50} notiert.

Tabelle D.1: Vergleich von Genauigkeit (AP_{70} auf der Klasse 'Car') und Geschwindigkeit (fps) verschiedener 3D Objektdetektoren auf dem KITTI Datensatz. Complex-YOLO unterscheidet sich in seiner Evaluierung von anderen Methoden, da es auf einem Validierungssplit des KITTI Trainingset getestet wurde, während die anderen Methoden auf dem KITTI Testset evaluiert wurden. Ein weiterer wichtiger Unterschied liegt in der Art der Bounding-Box-Erkennung: Complex-YOLO berechnet 2D-Bounding-Boxen in der Birds-Eye-View Perspektive. Für die Bestimmung der 3D-Bounding-Boxen werden daher feste Werte für die Höhe der Objekte angenommen [Simo18].
Zu beachten ist, dass die Anzahl der Daten, die pro Sekunde ausgewertet werden können, von der verwendeten Hardware abhängt.

Name	Modalitäten	Car 3D AP			fps
		Easy	Moderate	Hard	
VirConv-T [Wu23]	LiDAR	92.54	86.25	81.24	10.9
SFD [Wu22]	LiDAR+Kamera	91.73	84.76	77.92	10.2
PV-RCNN [Shi20]	LiDAR	90.25	81.43	76.82	12.5
UberATG-MMF [Lian19]	LiDAR+Kamera	88.40	77.43	70.22	12.5
PointRCNN [Shi19]	LiDAR	85.94	75.76	68.32	10
Complex-YOLO [Simo18]	LiDAR	67.72	64.00	63.01	50.4

Die Bewertung der Leistung von Modellen für semantische Segmentierung basiert auf dem Vergleich der modellgenerierten Segmentierungen mit den tatsächlichen, genauen Segmentierungen. Ein einfaches Maß dafür ist die Pixelgenauigkeit (Pixel Accuracy), die das Verhältnis von richtig klassifizierten zu falsch klassifizierten Pixeln angibt. Ein Problem für die Pixelgenauigkeit sind jedoch ungleich große Klassen. Wenn eine Klasse extrem unausgewogen ist, bedeutet dies, dass eine oder mehrere Klassen das Bild dominieren, während einige andere Klassen nur einen kleinen Teil des Bildes ausmachen. Um dies zu berücksichtigen, wird häufig das F_1-Maß, auch (Sørensen-)Dice-Score [Asga21] genannt, verwendet.

Wenn mehrere Klassen vorliegen, wird in dieser Arbeit das Makro F_1-Maß berechnetet, indem das F_1-Maß für jede Klasse separat berechnet und dann

der Durchschnitt über alle Klassen gebildet wird. Dieser Ansatz behandelt alle Klassen gleich, unabhängig von ihrer Häufigkeit im Datensatz.

Um einen Wert über alle Evaluationsdaten zu erhalten, wird das Mittel über alle F_1 Werte gebildet. Dieser Mittelwert wird in dieser Arbeit auch als F_1-Maß bezeichnet.

E Parameter der Algorithmen zur Sensorplatzierung

Dieses Kapitel beschreibt die Parameter der Implementierungen für GAOS und DLOS. Die in Tabelle E.1 aufgeführten Parameter beziehen sich auf den verwendeten genetischen Algorithmus in GAOS. Die Parameter in Tabelle E.2 wurden für den DDPG-Algorithmus in DLOS verwendet.

Tabelle E.1: Parameter der GAOS Implementation.

Iterationen	2000
Populationsgröße P	10
Reproduktion Nachkommen	6
α	20
β	$\max(\{0{,}5, 10/(1 + e^{itertionIdx/250})\})$

Tabelle E.2: Standardparameter der DLOS Implementation für den DDPG Algorithmus.

Anzahl an LiDAR Sensoren	3
Bufferkapazität	100.000
Batchgröße	32
ρ	0,99
θ	0,15
dt	0,01
Optimierer	Adam
Actor Lernrate	0,001
Critic Lernrate	0,002
Discount Faktor	0,99
τ	0,005
Training Iterationen	200

F Anwendung des Robustheitsoptimierungskonzepts bei der Semantische Segmentierung

F.1 Übersicht

In diesem Anhang wird eine Vorstudie zur Steigerung der Robustheit bei der Fusion von Bilddaten für die semantische Segmentierung. Dafür wird das erstellte Robustheitsoptimierungskonzept auf U-Net angewendet und ein synthetisch erzeugter Datensatz verwendet. U-Net wird dabei speziell erweitert, um Daten aus verschiedenen Sensorquellen effektiver zu integrieren. Zum ersten Mal werden Low Level und High Level Fusionsstrategien für U-Net systematisch daraufhin untersucht, wie effektiv sie Störungen bewältigen können und inwieweit sie die Robustheit der semantischen Segmentierung unter verschiedenen Bedingungen verbessern. Basierend auf dem im Kapitel 2 vorgestellten Trainings- und Evaluationskonzept, wird dessen spezifische Anwendung erstmals auf U-Net-Modelle evaluiert, um zu analysieren, wie die gezielte Integration von Störungen in den Trainingsprozess die Robustheit steigern kann.

F.2 Übersicht über die Architektur von U-Net

In diesem Anhang wird das U-Net, eine Architektur für die Bildsegmentierung, eingehend beschrieben. Diese Informationen bieten einen tieferen Einblick in die technischen Details der U-Net-Architektur, ihre Verlustfunktion und Evaluationsmaße, die im Rahmen der Studie verwendet wurden.

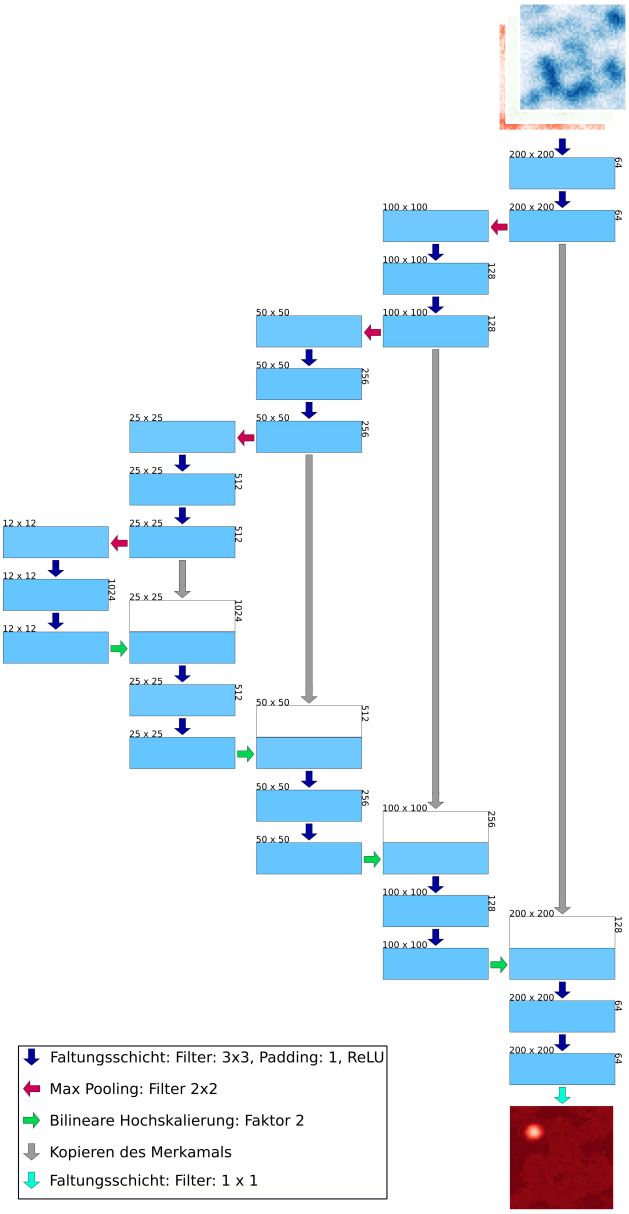


Abbildung F.1: Schematische Darstellung der verwendeten U-Net Architektur für ein RGB-Bild.

Die Architektur von U-Net ist in F.1 dargestellt. Das Netzwerk ist in zwei Teile unterteilt. Der erste Teil wird als Encoder (Kontraktionspfad) und der zweite Teil als Decoder (Expansionspfad) bezeichnet. Encoder und Decoder bestehen jeweils aus vier Blöcken. Ein Encoderblock besteht aus zwei 3×3 Faltungsschichten mit Batch-Normalisierung und ReLU (Rectifed Linear Unit), die als Aktivierungsfunktion nach jeder Schicht verwendet wird. Batch Normalisierung wurde in der ursprünglichen Arbeit nicht verwendet [Ronn15], hat aber den Vorteil, dass die Optimierungsumgebung geglättet wird, was zu einem stabileren Verhalten der Gradienten und einem schnelleren Training führt [Sant18]. Für das Downsampling wird am Ende jedes Encoderblocks ein 2×2 Maxpooling-Layer mit Stride 2 hinzugefügt, wodurch sich die räumliche Dimension der Merkmalskarte nach jedem Encoderblock halbiert, während sich die Anzahl der Feature-Kanäle verdoppelt. Im Gegensatz zu einfachen Autoencodern [Guo17, Long15], bei denen nur die Merkmalskarte des letzten Encoderblocks an den ersten Decoderblock weitergegeben wird, werden bei U-Net die Merkmalskartes von jedem Encoderblock an den entsprechenden Decoderblock zusätzlich zur Merkmalskarte des vorhergehenden Decoderblocks weitergegeben. Ein Decoderblock besteht somit aus Upsampling durch ein 2×2 Faltungsschicht, welche die Anzahl der Merkmalskanäle halbiert und die räumliche Dimension verdoppelt, einer Verknüpfung dieser Merkmalskarte mit der Merkmalskarte des entsprechenden Encoderblocks und zwei 3×3 Faltungsschichten mit Batch Normalization und ReLU als Aktivierungsfunktion. In der hier verwendeten Implementierung von U-Net wird Zero-Padding auf die Merkmalskarte angewandt, die vom Decoderblock stammt, um die räumliche Dimension der beiden Merkmalskartes anzupassen. Es ist zu beachten, dass in der Originalarbeit [Ronn15] die Merkmalskarte vom Encoder-Block ausgeschnitten wird. Im letzten Schritt wird ein 1×1 Faltungsschicht verwendet, um die Merkmalskarte auf die gewünschte Anzahl von Klassen abzubilden. Das Endprodukt dieser Architektur ist eine Segmentierungskarte, die dieselbe Breite und Höhe wie das Eingangsbild aufweist. Die Tiefe dieser Karte entspricht der Anzahl der zu identifizierenden Klassen, sodass U-Net für jeden Bildpunkt eine Wahrscheinlichkeitsverteilung über alle möglichen Klassen bereitstellt. Zur eindeutigen Klassenzuordnung jedes Bildpunktes wird der Klassenindex mit dem höchsten Wahrscheinlichkeitswert ausgewählt.

Als Verlustfunktion wird für U-Net in [Ronn15] der Cross-Entropy-Loss vorgeschlagen. In der hier verwendeten Implementierung wird jedoch eine Kombination aus Cross-Entropy-Loss und Dice-Loss verwendet. Laut [Jado20] wird durch die Kombination der Verlustfunktion die Flexibilität des Dice-Loss bei Klassenungleichgewichten und die Kurvenglättung des Cross-Entropy-Loss genutzt.

Im Folgenden wird ein U-Net, welches auf den Daten eines einzigen Sensors trainiert wurde, als U-Net *Sensor* bezeichnet, wobei *Sensor* den Namen des jeweiligen Sensors hat.

Definition F.1. *Es werden Bilddaten als ein mehrdimensionales Array* $\mathbf{B} \in [0{,}255]^{n_c \times n_h \times n_b}$ *definiert, wobei der Wertebereich [0,255] die standardmäßige 8-Bit-Farbkodierung pro Kanal widerspiegelt. Die Dimensionen des Arrays repräsentieren:*

- n_c *- Die Anzahl der Kanäle, abhängig von der Art des Bildes. Bei einem RGB-Bild einer visuellen Kamera gibt es drei Kanäle (Rot, Grün, Blau), während ein Graustufenbild, wie von einer Infrarotkamera, nur einen Kanal hat. Erweiterte Formate können zusätzliche Kanäle, wie den Alpha-Kanal für Transparenz, enthalten.*
- n_h *- Die Höhe des Bildes in Bildpunkten, repräsentiert die Anzahl der Bildpunkte in vertikaler Richtung.*
- n_b *- Die Breite des Bildes in Bildpunkten, repräsentiert die Anzahl der Bildpunkte in horizontaler Richtung.*

Für ein RGB-Bild bezeichnen die Kanäle $\mathbf{B}_R$, $\mathbf{B}_G$ *und* $\mathbf{B}_B \in [0{,}255]^{n_h \times n_b}$ *die Farbkomponenten Rot, Grün und Blau.*

Jeder Bildpunkt in diesem $n_c \times n_h \times n_b$-Array repräsentiert Informationen über Farbe und Intensität an einem bestimmten Punkt im Bild.

U-Net erhält Bilddaten $\mathbf{B} \in [0{,}255]^{n_b \times n_h \times n_c}$, wobei n_b die Breite, n_h die Höhe und n_c die Anzahl an Kanälen ist, entgegen. Als Ausgabe generiert U-Net eine Segmentierungskarte $\mathbf{K} \in [0{,}1]^{n_b \times n_h \times n_k}$, die dieselbe Breite n_b und Höhe n_h wie

das Eingangsbild aufweist. Die Anzahl der Kanäle n_k dieser Karte entspricht der Anzahl der zu identifizierenden Klassen, sodass U-Net für jeden Bildpunkt eine Wahrscheinlichkeitsverteilung über alle möglichen Klassen bereitstellt. Zur eindeutigen Klassenzuordnung jedes Bildpunktes wird der Klassenindex mit dem höchsten Wahrscheinlichkeitswert ausgewählt.

F.2.1 Erweiterung des U-Net Ansatzes durch Low Level Fusion

U-Net wird durch eine Low Level Fusionsstrategie erweitert.

Ein wesentlicher Vorteil dieser Methode ist die Erhaltung der ursprünglichen Datenstruktur und -qualität jedes einzelnen Sensors, während gleichzeitig eine umfassendere Datenbasis für nachfolgende Analyseprozesse geschaffen wird. Diese Art der Fusion ermöglicht eine tiefere und genauere Interpretation der kombinierten Daten, da sie die unterschiedlichen Perspektiven und Eigenschaften jedes Sensors nutzt.

Ein U-Net, welches die Low Level Fusionsstrategie verwendet, wird im Folgenden als U-Net Low Level bezeichnet.

Für die Low Level Fusion werden die Bilddaten $\mathbf{B}_1 \in [0{,}255]^{n_b \times n_h \times n_{c1}}$, $\mathbf{B}_2 \in [0{,}255]^{n_b \times n_h \times n_{c2}}$ zweier verschiedener Sensoren systematisch zusammengeführt, indem die Kanäle verbunden werden:

$$\mathbf{B} = \mathbf{B}_1 \circ \mathbf{B}_2 \in [0{,}255]^{n_b \times n_h \times (n_{c1} + n_{c2})}.$$

Die Operation $\circ$ symbolisiert das kanalweise Verbinden der Bilddaten $\mathbf{B}_1$ und $\mathbf{B}_2$, wodurch die Kanalanzahl der resultierenden Bilddaten $\mathbf{B}$ die Summe der Kanalanzahlen von $\mathbf{B}_1$ und $\mathbf{B}_2$ ist.

Die Daten $\mathbf{B}$ werden dann in das U-Net geladen. Die U-Net Architektur muss für diese Fusionsstrategie, bis auf die Anzahl der Eingangskanäle, nicht geändert werden.

Diese kanalweise Methode für die Low Level Fusion ist nur möglich, wenn beide Sensoren die gleiche Dimension haben.

F.2.2 Erweiterung des U-Net Ansatzes durch High Level Fusion

Die zweite Methode ist eine High Level Fusionsstrategie.

Der Vorteil dieser High Level Fusionsstrategie liegt in ihrer Fähigkeit, die Vielfalt und Komplementarität der Informationen aus verschiedenen Bildquellen zu nutzen, ohne dass eine direkte Interaktion oder Modifikation der ursprünglichen Bilddaten erforderlich ist.

Ein U-Net, welches die High Level Fusionsstrategie verwendet, wird im Folgenden als U-Net High Level bezeichnet.

Die High Level Fusionsstrategie bei semantischer Segmentierung beinhaltet einen zweistufigen Prozess. In der ersten Phase dieser Strategie werden alle verfügbaren Bilddaten $\mathbf{B}_1 \in [0,255]^{n_b \times n_h \times n_{c1}}$ und $\mathbf{B}_2 \in [0,255]^{n_b \times n_h \times n_{c2}}$ unabhängig voneinander analysiert und verarbeitet. Dabei wird für jedes Bild eine individuelle Segmentierungskarte $\mathbf{K}_1 \in [0,1]^{n_b \times n_h \times n_k}$ und $\mathbf{K}_2 \in [0,1]^{n_b \times n_h \times n_k}$ erstellt. In der zweiten Phase werden die erzeugten Segmentierungskarten zusammengeführt, um eine einzige Segmentierungskarte $\mathbf{K} \in [0,1]^{n_b \times n_h \times n_k}$ zu erstellen. Diese Fusion wird durch die Berechnung des Mittelwerts der Wahrscheinlichkeitsverteilungen über die korrespondierenden Bildpunkte der verschiedenen Segmentierungskarten hinweg realisiert. Wie auch beim ursprünglichen U-Net wird zur eindeutigen Klassenzuordnung jedes Bildpunktes der Klassenindex mit dem höchsten Wahrscheinlichkeitswert in $\mathbf{K}$ ausgewählt.

F.3 Definition und Parametrisierung der Bildstörungen

1 "**Unschärfe**": Es wird ein normalisierter Boxfilter mit einer Kernelgröße von $10, 20, 40, 80$ oder 160 angewendet. In realen Szenarien kann "Unschärfe" durch verschiedene Faktoren entstehen, z. B. durch Bewegungsunschärfe, die durch Kamera- oder

Objektbewegungen während der Bildaufnahme verursacht wird. Sie kann auch durch defokussierte Optiken entstehen.

2 "**Gleichmäßiges Rauschen**": Für jeden Superpixel der Größe 10×10 des Bildes wird eine ganze Zahl in $[0, \delta]$ zufällig gewählt und zum Superpixel addiert oder subtrahiert, wobei $\delta \in [16, 32, 64, 128, 255]$. Anschließend werden die Intensitätswerte der Pixel auf den Wertebereich zwischen $[0, 255]$ begrenzt. Bildrauschen ist eine zufällige Variation von Helligkeits- oder Farbinformationen in Bildern, die im abgebildeten Objekt so nicht vorhanden sind, und ist in der Regel ein Aspekt des elektronischen Rauschens. "Gleichmäßiges Rauschen" ist eine häufige Art von Bildrauschen. Diese Art von Rauschen kann durch fehlerhafte Sensorpixel, Bildkomprimierung, Übertragungsfehler oder Umgebungsfaktoren verursacht werden [Sing14, Mait18, Soma12].

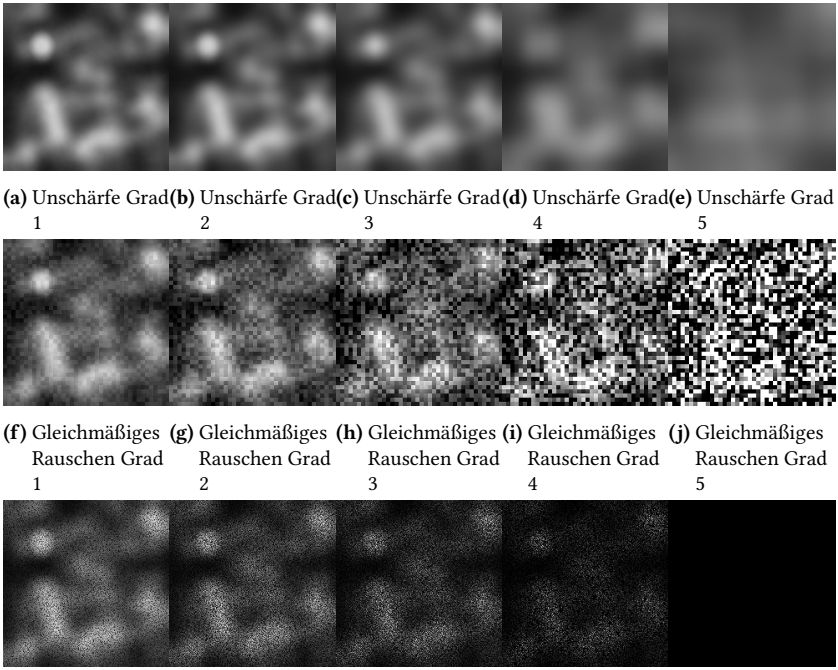
3 "**Tote Pixel**": Ein Anteil von 20%, 40%, 60%, 80% oder 100% der Pixel ist auf 0 gesetzt. Dies kann aufgrund von teilweisen oder, im Falle von 100%, vollständigen Sensorausfällen auftreten.

Die Menge aller betrachteten Störungen ist $\mathbf{S}$ = {"Unschärfe Grad 1", …, "Unschärfe Grad 5", "Gleichmäßiges Rauschen Grad 1", …, "Gleichmäßiges Rauschen Grad 5", "Tote Pixel Grad 1", …, "Tote Pixel Grad 5"}. Die zugehörigen Parameter bestimmen die Stärke beziehungsweise den Schwierigkeitsgrad der Störung. So entspricht beispielsweise eine "Unschärfe Grad 3" der Anwendung eines Boxfilters mit einer Kernelgröße von 40 auf das Bild. Für visuelle Beispiele aller Störungen sei auf den Appendix F.4 verwiesen.

F.4 Beispiele für die betrachteten Bildstörungen

Im folgenden Abschnitt werden Beispiele für die Bildstörungen "Unschärfe", "Gleichmäßiges Rauschen" und "Tote Pixel" präsentiert. Dadurch wird dargestellt, welche Art von Störungen auftreten können und wie sie sich auf die Ellipse auswirken. Die Abbildungen F.2a-F.2o zeigen die Störungen auf

den monochrom Sensordaten und die Abbildungen F.3a-F.3o auf den RGB Sensordaten.



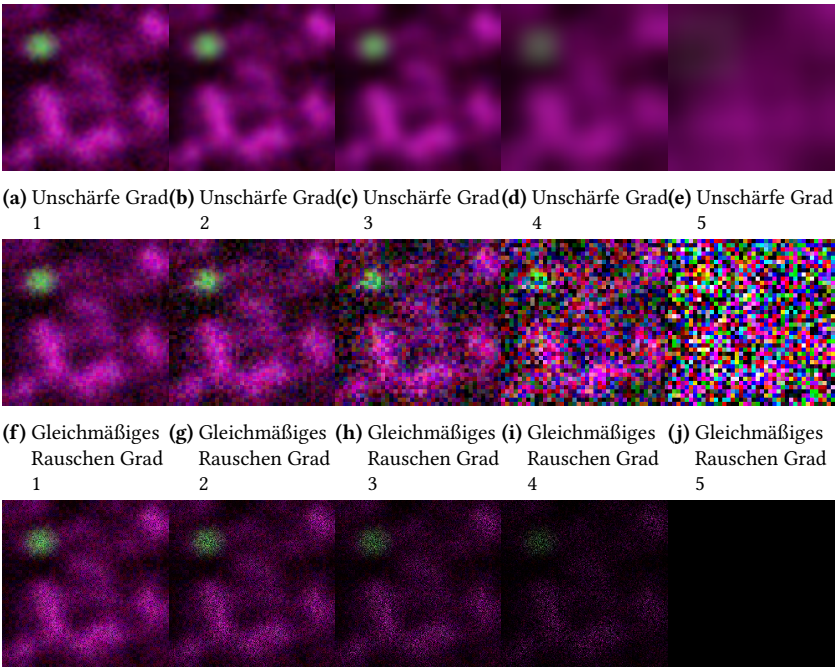
(a) Unschärfe Grad 1 **(b)** Unschärfe Grad 2 **(c)** Unschärfe Grad 3 **(d)** Unschärfe Grad 4 **(e)** Unschärfe Grad 5

(f) Gleichmäßiges Rauschen Grad 1 **(g)** Gleichmäßiges Rauschen Grad 2 **(h)** Gleichmäßiges Rauschen Grad 3 **(i)** Gleichmäßiges Rauschen Grad 4 **(j)** Gleichmäßiges Rauschen Grad 5

(k) Tote Pixel Grad 1 **(l)** Tote Pixel Grad 2 **(m)** Tote Pixel Grad 3 **(n)** Tote Pixel Grad 4 **(o)** Tote Pixel Grad 5

Abbildung F.2: Beispiel für die Störungen: ”Unschärfe”, ”Gleichmäßiges Rauschen” und ”Tote Pixel” mit den Schwierigkeitsgraden: 1, 2, 3, 4 und 5 auf den monochrom Sensordaten auf dem synthetisch erzeugten Datensatz.

Aus den Abbildungen F.2a-F.3o wird deutlich, dass bei allen drei Störungen die Ellipse bei den Schwierigkeitsgraden 1 und 2 noch gut erkennbar ist und eine klare Abgrenzung zum Hintergrund besteht. Bei Grad 3 sind die Lage und die Dimension der Ellipse immer noch gut erkennbar, jedoch sind die Ränder zum Hintergrund unscharf oder ausgefranst. Bei Grad 4 und 5 hingegen ist es äußerst schwierig, die Ellipse zu identifizieren. Optisch sind die Störungen auf den monochrom Sensordaten stärker, als auf den RGB Daten, da durch



(a) Unschärfe Grad 1 **(b)** Unschärfe Grad 2 **(c)** Unschärfe Grad 3 **(d)** Unschärfe Grad 4 **(e)** Unschärfe Grad 5

(f) Gleichmäßiges Rauschen Grad 1 **(g)** Gleichmäßiges Rauschen Grad 2 **(h)** Gleichmäßiges Rauschen Grad 3 **(i)** Gleichmäßiges Rauschen Grad 4 **(j)** Gleichmäßiges Rauschen Grad 5

(k) Tote Pixel Grad 1 **(l)** Tote Pixel Grad 2 **(m)** Tote Pixel Grad 3 **(n)** Tote Pixel Grad 4 **(o)** Tote Pixel Grad 5

Abbildung F.3: Beispiel für die Störungen: "Unschärfe", "Gleichmäßiges Rauschen" und "Tote Pixel" mit den Schwierigkeitsgraden: 1, 2, 3, 4 und 5 auf den RGB Sensordaten auf dem synthetisch erzeugten Datensatz.

die zusätzliche Farbinformation die Lage der Ellipse länger erkennbar ist. So kann man bei allen drei Störungen bis in das Grad 4 bei den RGB Sensordaten die Ellipse erkennen.

F.5 Ergebnis der Robustheits-optimierungskonzept bei U-Net

Dieser Abschnitt zeigt die weiteren Ergebnisse, welche im Rahmen dieser Arbeit mit dem U-Net Modellen und KISA erzeugt wurden. Die verwendeten Störungen sind ”Unschärfe”, ”Gleichmäßiges Rauschen” und ”Tote Pixel” präsentiert. In den Abbildungen F.4 - F.12 wird die Multifaktorielle Performanz-Evaluation der U-Net Low Level Fusion dargestellt, wenn KISA für eine speziellen Störung angewendet wird.

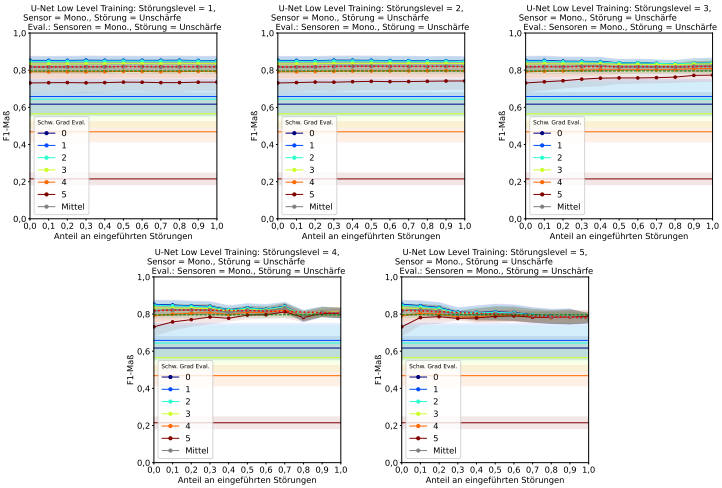


Abbildung F.4: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung ”Unschärfe” mit verschiedenen Graden auf den Monochrom Daten.

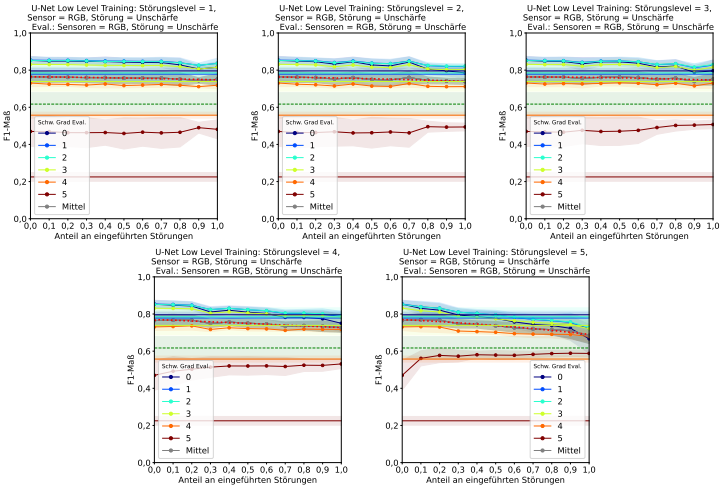


Abbildung F.5: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung "Unschärfe" mit verschiedenen Graden auf den RGB Daten.

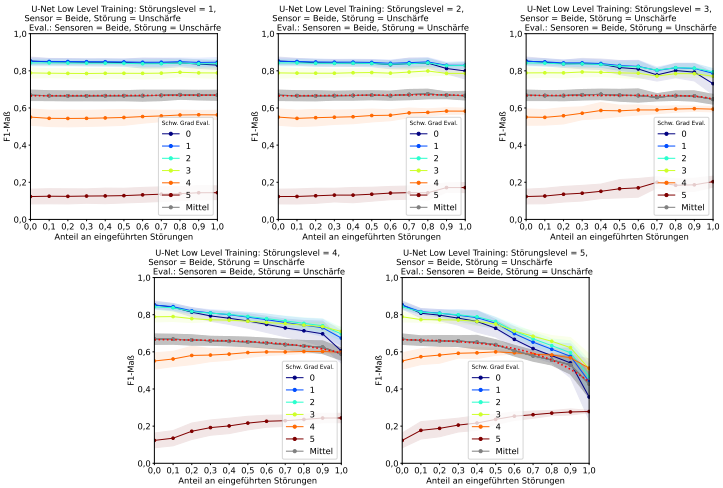


Abbildung F.6: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung "Unschärfe" mit verschiedenen Graden auf den Daten beider Sensoren.

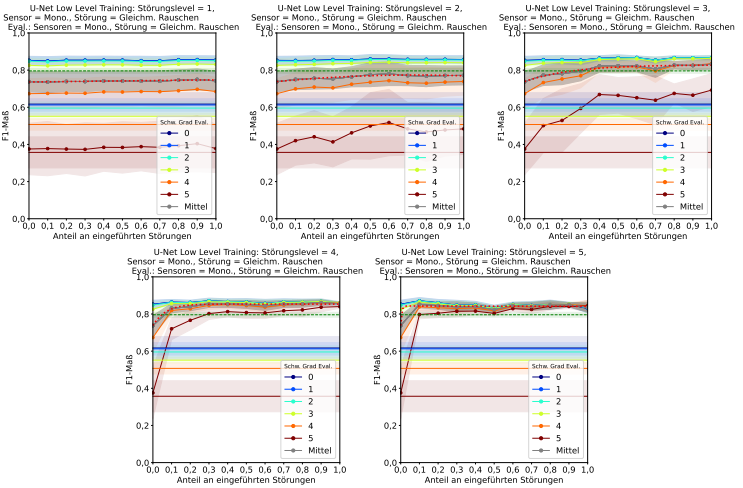


Abbildung F.7: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung "Gleichmäßiges Rauschen" mit verschiedenen Graden auf den Monochrom Daten.

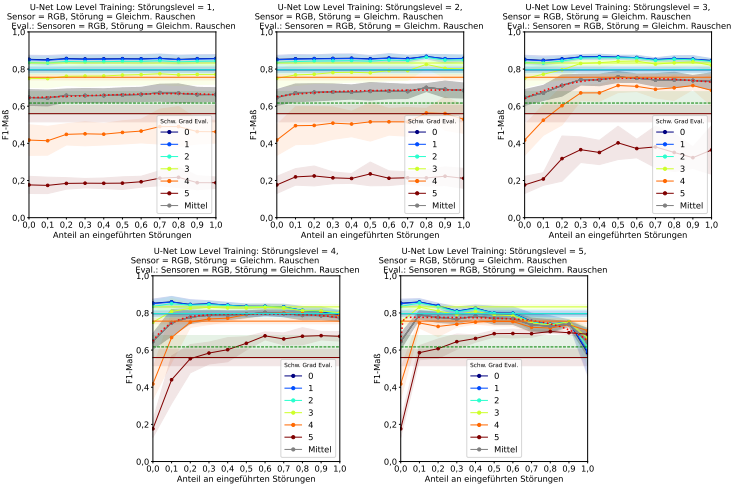


Abbildung F.8: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung "Gleichmäßiges Rauschen" mit verschiedenen Graden auf den RGB Daten.

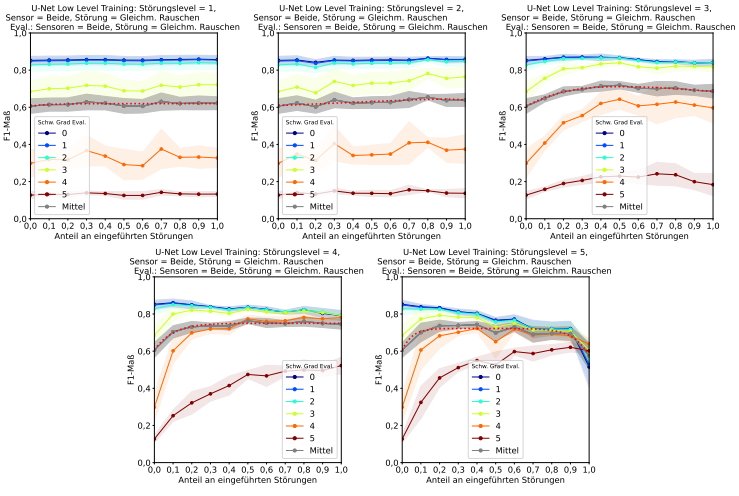


Abbildung F.9: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung "Gleichmäßiges Rauschen" mit verschiedenen Graden auf den Daten beider Sensoren.

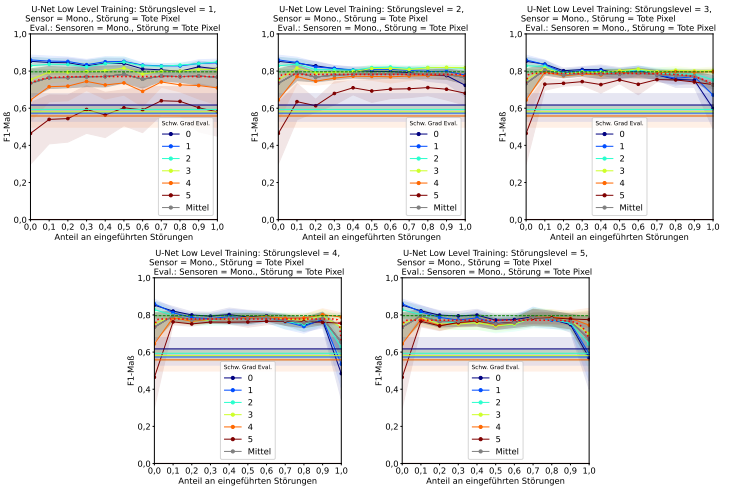


Abbildung F.10: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung "Tote Pixel" mit verschiedenen Graden auf den Monochrom Daten.

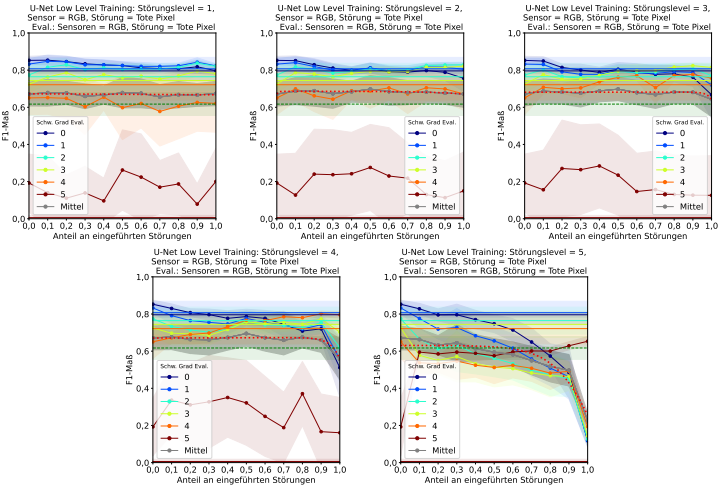


Abbildung F.11: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung ”Tote Pixel” mit verschiedenen Graden auf den Daten RGB Daten.

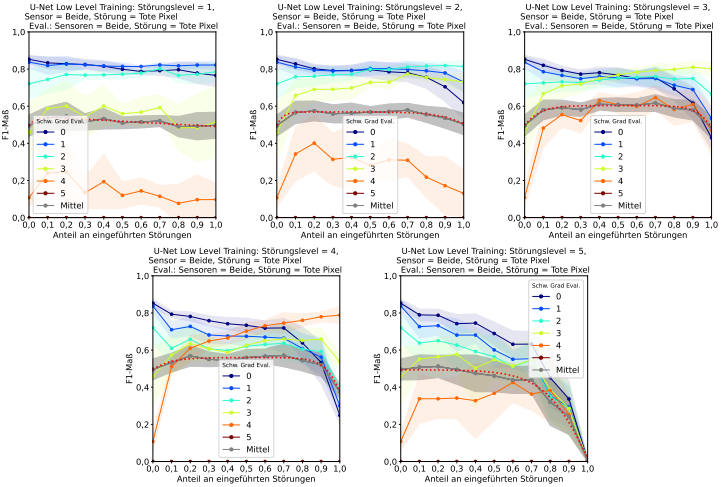


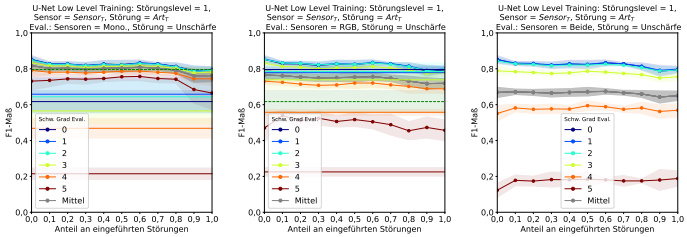
Abbildung F.12: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störung ”Tote Pixel” mit verschiedenen Graden auf den Daten beider Sensoren.

In Tabelle F.1 sind die Mittelwerte über die Schwierigkeitsgrad der multifaktoriellen Performanz für p_0, die Sättigungswerte und maximale multifaktoriellen Performanz für die einzelnen Störungsarten dargestellt. Daran lässt sich ablesen, dass bei der Störung "Unschärfe" die Robustheit von U-Net schnell gesättigt ist, da die Störung "Unschärfe" nur einen geringen negativen Einfluss auf die Leistung von U-Net Low Level hat, wie Abbildung 7.4 zeigt. Über alle betrachteten Störungsarten und Schwierigkeitsgrad beträgt der Mittelwert p_{satt} des Anteils, bei dem die Sättigung erreicht wird, 5,6% und die mittlere Multifaktorielle Performanz dieser Sättigungswerte ist 0,71.

Tabelle F.1: Mittelwerte über alle Schwierigkeitsgrad hinweg für die Multifaktorielle Performanz $M_p(\hat{S})$ von U-Net Low Level, beim Training ohne Störung p_0, bei dem Anteil, an dem die Sättigung p_{satt} erreicht wird und bei dem Anteil, bei dem die maximale Multifaktorielle Performanz p_{max} erreicht wird, aufgeteilt nach den einzelnen Störungsarten.

Störungsart	$(p_0, M_{p_0}(\hat{S}))$	$(p_{\text{satt}}, M_{p_{\text{satt}}}(\hat{S}))$	$(p_{\text{max}}, M_{p_{\text{max}}}(\hat{S}))$
"Tote Pixel"	(0%, 0,61)	(2,0%, 0,62)	(30%, 65)
"Gleichmäßiges Rauschen"	(0%, 0,65)	(14%, 0,77)	(34%, 0,79)
"Unschärfe"	(0%, 0,74)	(0%, 0,74)	(0,4%, 0,74)

Für die Abbildungen F.13-F.15 wurde KISA mit alle drei Bildstärungen "Unschärfe", "Gleichmäßiges Rauschen" und "Tote Pixel" mit 5 Störungsgraden auf U-Net Low Level angewendet.

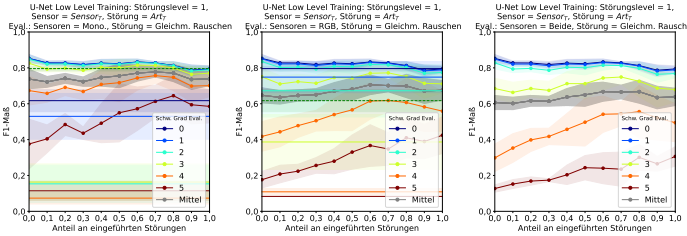


(a) "Unschärfe" auf den Monochrom Daten

(b) "Unschärfe" auf den RGB Daten

(c) "Unschärfe" auf beiden Sensordaten

Abbildung F.13: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störungen auf den Daten des Synthetischen Datensatzes und evaluiert bei "Unschärfe".

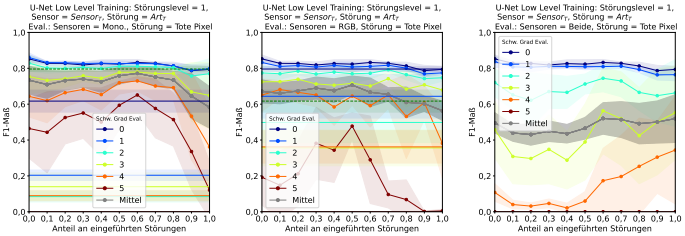


(a) "Gleichmäßiges Rauschen" auf den Monochrom Daten

(b) "Gleichmäßiges Rauschen" auf den RGB Daten

(c) "Gleichmäßiges Rauschen" auf beiden Sensordaten

Abbildung F.14: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störungen auf den Daten des Synthetischen Datensatzes und evaluiert bei "Gleichmäßiges Rauschen".



(a) "Tote Pixel" auf den LiDAR Daten

(b) "Tote Pixel" auf den RADAR Daten

(c) "Tote Pixel" auf beiden Sensordaten

Abbildung F.15: Performanz von U-Net Low Level trainiert mit unterschiedlichen Anteilen an eingeführter Störungen auf den Daten des Synthetischen Datensatzes und evaluiert bei "Tote Pixel".

G BraTS - Datensatz

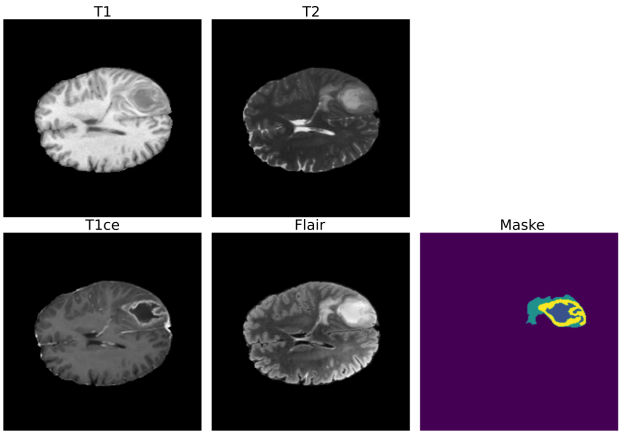


Abbildung G.1: Beispiel für MRT-Scans (T1, T2, T1ce, Flair) ohne Störung, zusammen mit der entsprechenden Segmentierungskarte (grün - peritumorales Ödem, gelb - verstärkter Kern, blau - nekrotischer und nicht verstärkter Tumorkern) aus dem BraTS Datensatz.

Der BraTS-Datensatz [Baid21, Menz14, Baka17, Hata21] ist ein bekannter Benchmark-Datensatz zur Bewertung von Hirntumorsegmentierungsalgorithmen. Der Datensatz umfasst multimodalen MRT-Daten und manuellen Tumorsegmentierungen. Die vier MRT-Modalitäten sind: T1-gewichtet (T1), kontrastverstärktes T1-gewichtet (T1ce), T2-gewichtet (T2) und Flair (Fluid Attenuated Inversion Recovery). Die Daten stammen aus multi-institutionellen Quellen und bieten so eine breite Vielfalt hinsichtlich der Bildgebungsbedingungen. Die Expertenannotationen kennzeichnen die drei Tumorunterregionen: das peritumorale Ödem, den verstärkten Tumorkern sowie den nekrotischen und nicht verstärkten Tumorkern. Der Trainingssatz des BraTS 2021 beinhaltete

1.251 Scans, wobei jeder Scan starr ausgerichtet, auf eine isotrope Auflösung von $1 \times 1 \times 1$ mm skaliert und von Schädelstrukturen befreit wurde. Jeder Scan hat die Dimension $240 \times 240 \times 155$ Pixeln. Dies bedeutet, dass jeder Scan aus 155 Schichten besteht, wobei eine Schicht die Dimension 240×240 Pixel hat. Im Folgenden wird jede Schicht als ein Grauwertbild betrachtet. Der BraTS-Datensatz wird regelmäßig überarbeitet, um den sich ändernden Anforderungen der Forschung gerecht zu werden. Für diese Arbeit wird die Version BraTS 2021 verwendet. Abbildung G.1 zeigt ein Beispiel der MRT-Modalitäten und der entsprechenden Tumorkarte für eine Schicht.

H Complex-YOLO

Dieser Anhang erläutert die Netzwerkarchitektur und die Funktion der einzelnen Schichten der verwendeten auf YOLO basierenden Netzwerke: Mini Complex-YOLO, Kleines Complex-YOLO und Complex-YOLO. Das Complex-YOLO-Modell besteht aus drei Schritten, einer Datenvorbereitung, in der werden Punktwolkedaten in BEV-Projektionen umgewandelt. Aus diesen BEV-Projektionen werden im nächsten Schritt Merkmale extrahiert, aus denen wiederum im letzten Schritt eine Liste von Objektvorschlägen generiert wird, welche durch Kofidenzschwellwert und Non-Maximum-Suppression verfeinert wird.

H.1 Datenvorbereitung

Der YOLO-Algorithmus benötigt zweidimensionale Bilddaten als Eingabe. Deshalb werden die dreidimensionalen Punktwolken Φ von LiDAR- oder RADAR-Sensoren zunächst in eine zweidimensionale Bird's-Eye-View-Projektion (BEV-Projektion) $\mathbf{B} \in [0{,}255]^{3 \times n_h \times n_b}$ umgewandelt. Abbildung H.1 zeigt beispielhaft die Erzeugung einer BEV-Projektion aus einer LiDAR-Punktwolke.

Eine beliebige Punktwolke ist folgendermaßen definiert:

Definition H.1. *Eine Punktwolke Φ ist definiert als eine Menge von n_P Punkten im Raum, wobei jeder Punkt repräsentiert wird von einem Vektor $\Phi[i] \in \mathbb{R}^{3+a}$. Dabei gilt:*

- *Die ersten drei Komponenten jedes Vektors $\Phi[i]$ repräsentieren die räumlichen Koordinaten des Punktes (x, y, z) in Bezug auf einen bestimmten Ursprung.*

- *$a \in \mathbb{N}$ repräsentiert die Anzahl zusätzlicher Attribute, die jedem Punkt zugeordnet sind. Diese können verschiedene Daten umfassen, wie z.B. die Intensität bei LiDAR-Sensoren oder Signalmerkmale bei RADAR-Sensoren, die weitere Informationen über die Eigenschaften und Zustände der erfassten Objekte oder Oberflächen bieten.*
- *$n_P \in \mathbb{N}$, die Gesamtzahl der Punkte in der Punktwolke. Die Größe von n_P variiert je nach den Fähigkeiten des eingesetzten Sensors und der Umgebung.*

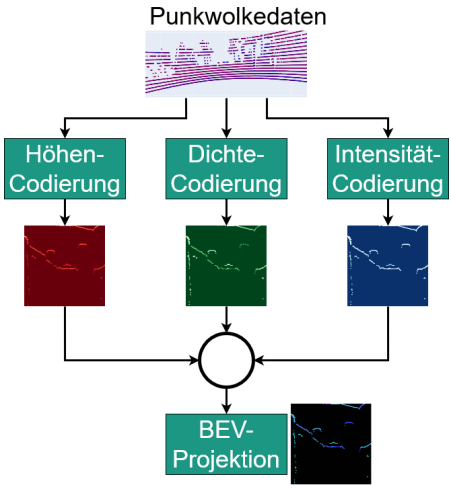


Abbildung H.1: Visualisierung der Datenvorbereitung für Complex-YOLO. Die Darstellung zeigt die Zusammenstellung einer BEV-Projektion aus den Kanälen: In Rot wird die Höhe des Punktes, in Grün die Dichte Punktwolke, und in Blau die Intensität des LiDAR-Punktes codiert. Für die Darstellung wurde ein Ausschnitt der BEV-Projektion dargestellt.

Zuerst werden irrelevante Punkte entfernt, sodass nur Punkte innerhalb einer Region of Interest (ROI) $[0,\mathrm{ROI}_X] \times [-\frac{\mathrm{ROI}_Y}{2}, \frac{\mathrm{ROI}_Y}{2}] \times [\mathrm{ROI}_Z^0, \mathrm{ROI}_Z^1]$ beibehalten

werden. Dieser Filter führt zu einer bereinigten Punktwolke Φ':

$$\begin{aligned}\Phi' = \{\Phi[i] \in \Phi; \Phi[i,0] \in \left[0, \mathrm{ROI}_X\right] \ \wedge \\ \Phi[i,1] \in \left[-\frac{\mathrm{ROI}_Y}{2}, \frac{\mathrm{ROI}_Y}{2}\right] \ \wedge \\ \Phi[i,2] \in \left[\mathrm{ROI}_Z^0, \mathrm{ROI}_Z^1\right]\}.\end{aligned}$$

In Abbildung K.1 ist die Region von Interesse für den Astyxdatensatz beispielhaft dargestellt.

Im nächsten Schritt wird jedem Punkt $\Phi'[i]$, basierend auf seiner Position, ein Indexpaar (u,v) in der BEV-Projektion zugeordnet:

$$\begin{aligned}u &= n_h \cdot \frac{\Phi'[i,0]}{\mathrm{ROI}_X}, \\ v &= n_b \cdot \frac{\Phi'[i,1]}{\mathrm{ROI}_Y} + \frac{n_b}{2}.\end{aligned}$$

Damit der Ursprung des Sensors in der Mitte der BEV-Projektion liegt, wird bei v die Hälfte der Breite der BEV-Projektion addiert. Da hier nur der Bereich vor dem Sensor in Betracht gezogen wird, muss nicht die Hälfte der Höhe n_h der BEV-Projektion zu u aufaddiert werden.

In den Farbkanälen $(\mathbf{B}_R, \mathbf{B}_G, \mathbf{B}_B)$ der BEV-Projektion werden verschiedene Informationen über die Punktwolke codiert:

- Im ersten Farbkanal $\mathbf{B}_R$ wird die Höhe des Punktes entlang der z-Achse $\Phi'[i,2]$ codiert:

  $$\mathbf{B}_R[u,v] = \frac{\Phi'[i,2] - \mathrm{ROI}_Z^0}{\mathrm{ROI}_Z^1 - \mathrm{ROI}_Z^0} \cdot 255 \in [0,255]$$

  Durch die Subtraktion von ROI_Z^0 im Zähler und Division durch $\mathrm{ROI}_Z^1 - \mathrm{ROI}_Z^0$ im Nenner wird die Höhe standardisiert.

- Im zweiten Farbkanal $\mathbf{B}_G$ wird ein Wert, der die Dichte der Punktwolke an der entsprechenden Stelle beschreibt, codiert:

$$\mathbf{N}[u,v] = \left| \left\{ \Phi'[i] \in \Phi'; u = n_h \cdot \frac{\Phi'[i,0]}{\mathrm{ROI}_X} \wedge v = n_b \cdot \frac{\Phi'[i,1]}{\mathrm{ROI}_Y} + \frac{n_b}{2} \right\} \right|,$$
$$\mathbf{B}_G[u,v] = \min \left\{ 1, \frac{\log(\mathbf{N}[u,v]+1)}{\log(64)} \right\} \cdot 255 \in [0{,}255],$$

  wobei $|.|$ die Kardinalität der gegebenen Menge bezeichnet. $\mathbf{N}[u,v]$ beschreibt also die Anzahl an Punkten in Φ', die auf (u,v) projiziert werden. Durch den Bruch $\frac{\log(\mathbf{N}[u,v]+1)}{\log(64)}$ wird die normalisierte Dichte angegeben. 64 wurde in der ursprünglichen Implementierung gewählt, da diese für Daten des Veldoyne-64e ausgelegt war, welcher 64 Laserstrahlen hat. Um die Konsistenz mit der ursprünglichen Implementierung zu wahren, wird hier weiterhin 64 verwendet.

- Im dritten Farbkanal $\mathbf{B}_B$ wird weitere Information, die der Sensor für jeden Punkt misst, codiert:

$$\mathbf{B}_B[u,v] = \frac{\Phi'[i,3] - a_{\min,\mathrm{Inf}}}{a_{\max,\mathrm{Inf}} - a_{\min,\mathrm{Inf}}} \cdot 255 \in [0{,}255].$$

  Diese Information kann die Intensität bei LiDAR Punkten und bei RADAR Punkten die relative radiale Geschwindigkeit v_r oder Magnitude sein. Dabei stehen $a_{\min,.}$, $a_{\max,\mathrm{Inf}}$ für den minimalen beziehungsweise maximalen Wert der Information, der für Intensität ($a_{\max,\mathrm{Int}} = 255$, $a_{\min,\mathrm{Int}} = 0$), Magnitude ($a_{\max,\mathrm{Mag}} = 102$, $a_{\min,\mathrm{Mag}} = 0$) oder v_r ($a_{\max,v_r} = 5{,}20$, $a_{\min,v_r} = -5{,}11$) gemessen werden kann.

H.2 Erkennungsköpfe

Der Erkennungskopf berechnet aus den extrahierten Merkmalen Bounding Boxen. Complex-YOLO teilt die BEV-Projektion in $G \times G$ Gitterzellen auf

und berechnet für jede dieser G^2 Zellen Bounding Boxen $\boldsymbol{B}$, Konfidenzwerte $K \in [0,1]$ sowie einen Klassifikationsvektor $V \in [0,1]^{n_k}$ unter Verwendung von drei spezifischen Ankerpunkten. n_k steht dabei für die Anzahl an Klassen. Der Konfidenzwert gibt an, wie sicher der Detektor mit der vorgeschlagenen Detektion ist. Der Klassifikationsvektor ist ein sogenannter One-Hot-Vektor. Der One-Hot-Vektor hat für jede der n_k Klassen einen Wert zwischen 0 und 1 und gibt die Sicherheit an, mit der diese Vorhersage zu der jeweiligen Klasse gehört.

Eine bekannte Limitierung des ursprünglichen YOLO-Algorithmus ist, dass jede Bounding Box zwangsläufig parallel zu den Bildrändern ausgerichtet ist, was zu einer fixen Orientierung führt. Im Kontext des Straßenverkehrs ist jedoch die Ausrichtung der Verkehrsteilnehmer von entscheidender Bedeutung, da sie wesentliche Informationen über deren aktuelle und zukünftige Fahrtrichtungen liefert. Complex-YOLO adressiert diese Einschränkung durch die Integration der Orientierungsdetektion, speziell des Gier-oder Yaw-Winkels, im Erkennungskopf, basierend auf den Erkenntnissen von [Simo18].

Um neben Position und Dimension der Bounding Box auch die Orientierung effektiv zu bestimmen, nutzt Complex-YOLO das Konzept des Euler-Region-Proposals. Hierbei wird die Orientierung mittels einer komplexen Zahl repräsentiert. Diese Darstellungsweise hat den mathematischen Vorteil, dass sie einen geschlossenen Raum bildet und Singularitäten vermeidet, was die Berechnung und Interpretation der Orientierung vereinfacht.

Für jede erkannte Bounding Box generiert das Netzwerk sechs Prior-Werte $(t_x, t_y, t_w, t_h, t_{im}, t_{re})$, die in Verbindung mit den Ankerpunkten die Lage (t_x, t_y), Größe (t_w, t_h) und Orientierung (t_{im}, t_{re}) der Bounding Box präzise definieren. Die Berechnung dieser Attribute erfolgt durch die folgenden Formeln, wobei σ die Sigmoid-Funktion darstellt, die eine Normalisierung der Zentrumskoordinaten bewirkt, und e^{t_w} beziehungsweise e^{t_h} eine exponentielle Skalierung für Breite und Höhe der Bounding Box ermöglichen. Die Orientierung b_α wird schließlich aus den komplexen Zahlen t_{im} und t_{re} mittels des

Arkustangens berechnet:

$$
\begin{aligned}
b_x &= \sigma(t_x) + c_x, \\
b_y &= \sigma(t_y) + c_y, \\
b_w &= p_w \cdot e^{t_w}, \\
b_h &= p_h \cdot e^{t_h}, \\
b_\alpha &= \arctan_2(t_{im}, t_{re}).
\end{aligned}
$$

Die Schätzung des Winkels mittels komplexer Zahl hat den Vorteil, dass Singularitäten, wie der Übergang von 359° zu 0°, vermieden werden. Somit sorgt die komplexe Darstellung dafür, dass alle Winkelberechnungen innerhalb des geschlossenen Raums der komplexen Zahlen stattfinden.

Somit wird jedes Objekt $\mathcal{O}$ durch die Werte der Bounding Box, den Konfidenzwert und den Index, an dessen Stelle der Klassifikationsvektors V maximal ist, beschrieben:

$$
\mathcal{O} = (b_x, b_y, b_w, b_h, b_\alpha, K, \arg\max(V)).
$$

Im Vergleich zum YOLO Algorithmus werden bei Complex-YOLO zusätzlich t_{re} und t_{im} geschätzt. Damit werden bei Complex-YOLO $G \times G \times (C + B \cdot 7)$ Werte je Erkennungskopf geschätzt. In der Implementation dieser Arbeit ist $G = 26$, $C = 3$, $B = 1$ und es werden für jede Bounding Box 3 verschiedene Ankerpunkte verwendet. Die genauen Ankerpunkte sind in Tabelle H.1 angegeben.

Tabelle H.1: Verwendete Ankerpunkte (Weite, Breite) für die unterschiedlichen Erkennungsköpfe.

Header	Anker 1	Anker 2	Anker 3
1	11, 15	11, 25	23, 49
2	23, 55	24, 53	25, 61
3	11, 15	11, 25	23, 49
4	11, 15	10, 24	11, 25
5	23, 49	23, 55	24, 53
6	24, 60	27, 63	29, 74

H.3 Objektliste

Complex-YOLO berechnet $G^2 \cdot B \cdot 3$ Objekte pro BEV-Projektion.

Definition H.2. *Objekte werden in einer Objektliste $\mathcal{O}$ zusammengefasst. Dabei hat jedes Objekt einen eindeutig zu identifizierenden Index. Die Menge aller Indizes wird in der Menge $\mathcal{I}$ zusammengefasst.*

Nach der Berechnung der Bounding Boxen für die BEV-Projektion durch Complex-YOLO, wird die initiale Objektliste durch zwei Methoden verfeinert, um die Genauigkeit zu erhöhen und redundante Einträge zu minimieren: den Konfidenzschwellenwert und eine gewichtete NMS.

Definition H.3. *Konfidenzschwellenwert: Jedes der $G^2 \cdot B \cdot 3$ Objekte einer Objektliste ist mit einem Konfidenzwert K versehen, der angibt, wie sicher das Modell ist, dass die Box ein tatsächliches Objekt enthält. Objekte, deren Konfidenzniveau unter einem vordefinierten Schwellenwert K_s liegt, werden aus der Objektliste entfernt:*

$$\mathcal{O}_{Konf} = \left\{(b_x, b_y, b_w, b_h, b_\alpha, K, \arg\max(V)) \in \mathcal{O}; K \geq K_s\right\}$$

Gewichtete NMS adressiert das Problem, dass mehrere berechnete Objekte dasselbe wahre Objekt umschließen, was zu Überlappungen führt. Dafür werden für jede Gruppe überlappender Objekte die Werte $(b_x, b_y, b_w, b_h, b_\alpha)$ der Objekte durch ein gewichtetes Mittel berechnet. Die Gewichte für dieses Mittel sind die Konfidenzwerte K der einzelnen Objekte. Dieser Prozess gewährleistet, dass jedes erkannte Objekt in der Szene durch genau eine Bounding Box repräsentiert wird.

Definition H.4. *Die Operation der Non-Maximum Suppression (NMS) wird notiert durch:*

$$\mathcal{O}_{NMS} = NMS(\mathcal{O}).$$

Hierbei ist NMS(.) eine Funktion, die folgende Schritte umfasst:

1 *Für jedes Objekt* $o \in \mathcal{O}$ *wird diese Gütezahl berechnet:*

$$s = K \cdot max(V),$$

dabei sind K *und* V *der Konfidenzwert und Klassenvektor des Objekts* o.

2 *Die Objektindizes* $\mathcal{I}$ *werden anhand der Gütezahl* s *sortiert:* $s_0 \geq s_1 \geq \cdots \geq s_{|\mathcal{I}|-1}$.

3 *Für das Objekt* $o_i \in \mathcal{O}$ *ist* s_i *die größte Gütezahl und es wird die Intersection over Union (IoU) (siehe Anhang D für Details zur IoU) mit allen anderen Objekten* o_j, $i < j$, *der selben Klasse* $\arg\max(V_i) = \arg\max(V_j)$ *als:*

$$\mathcal{J} := \{j \in \mathcal{I}; i < j \wedge \arg\max(V_i) = \arg\max(V_j)\},$$

berechnet.

4 *Wenn die IoU von* o_i *und* o_j, $j \in \mathcal{J}$ *einen Schwellenwert* S_{IoU} *überschreitet, werden diese Objekte zusammengefasst:*

$$\mathcal{J}_{neu} = \{i\} \cup \{j \in \mathcal{J}; IoU(o_i, o_j) \geq S_{IoU}\}$$

5 *Mit den Objekten, welche die Indizes in* $\mathcal{J}_{neu}$ *haben, wird ein neues Objekt* o_{NMS} *durch Bildung des gewichteten Mittels der Boundingboxwerte berechnet:*

$$b^{NMS}_{x,y,w,h,\alpha} = \frac{\sum_{j \in \mathcal{J}_{NMS}} K_j \cdot b^j_{x,y,w,h,\alpha}}{\sum_{j \in \mathcal{J}_{NMS}} K_j}$$

$$K_{NMS} = K_i$$

$$V_{NMS} = V_i$$

6 *Das neu berechnetet Objekt* o_{NMS} *wird in eine neue Objektliste* $\mathcal{O}_{NMS}$ *hinzugefügt. Die Indizes aus* $\mathcal{J}_{neu}$ *werden aus* $\mathcal{I}$ *entfernt.*

7 *Die Schritte 3-6 werden wiederholt, bis keine überlappenden Objekte mehr vorhanden sind, also* $\mathcal{I}$ *leer ist.*

H.4 Netzwerkarchitektur

Die Architekturen der verwendeten Netzwerke setzen sich aus einer sequenziellen Abfolge folgender Schichttypen zusammen:

- Faltungsschichten (Conv.) führen Faltungsoperationen durch, wobei die Gewichte jedes Filters während des Trainings erlernt werden. Die ”Filter” Spalte gibt die Anzahl der Filter pro Schicht an. ”Größe” bezieht sich auf die Dimension des Filters in den Faltungsschichten (Breite x Höhe), und der ”Schritt” zeigt an, wie der Filter über die Eingabedaten gleitet (z.B., bedeutet 3 x 3 / 2 einen 3x3 Filter mit einem Schritt von 2). Zudem werden die Dimensionen der Ein- und Ausgangsmerkmalskarten angegeben (Breite x Höhe x Kanäle).
- Max Pooling Schichten (Max Po.) verringern die räumliche Dimension der Merkmalskarten, indem sie den maximalen Wert innerhalb eines definierten Bereichs (Poolgröße) auswählen. ”Größe” definiert hier die Dimension des Pooling-Fensters (Breite x Höhe), und der SSchritt"gibt die Verschiebung des Pooling-Fensters über die Eingabedaten an (z.B., bedeutet 2 x 2 / 2 ein 2x2 Fenster mit einem Schritt von 2). Weiterhin wird die Dimension der Ein- und Ausgangsmerkmalskarten dargestellt (Breite x Höhe x Kanäle).
- Route Schichten (Route) erlauben die flexible Kombination von Merkmalskarten aus unterschiedlichen Bereichen des Netzwerks. Eine Zeile, die eine Route Schicht beschreibt, gibt an, welche Merkmalskarten früherer Schichten einbezogen werden. Wird nur eine Schicht genannt, so wird ausschließlich diese Merkmalskarte weitergeleitet. Bei Nennung von zwei oder mehr Schichten werden die entsprechenden Merkmalskarten entlang der Kanaldimension verbunden.
- Skip Connection (Skip Con.) addiert auf die Merkmalskarte der vorherigen Schicht eine Merkmalskarte einer Schicht, die in der ersten Spalte angegeben wird.

- Upsampling (Ups.) werden die Pixelwerte direkt wiederholt. Dadurch wird die räumliche Auflösung der Merkmalskarte um den Faktor der in der Spalte Filter steht erhöht.
- Erkennungskopf (Header) bezieht sich auf den YOLO-Erkennungskopf. Die verwendeten Ankerpunkte werden in Tabelle H.1 aufgeführt.

Das Mini und Kleines Complex-YOLO-Netzwerk stellen eine vereinfachte Version des vollständigen Complex-YOLO-Netzwerks dar.

Tabelle H.2: Detaillierte Architektur des Mini YOLO-Netzwerks. Diese Tabelle präsentiert die sequenzielle Konfiguration der Netzwerkschichten, einschließlich Faltungsschichten (Conv.), Max Pooling Schichten (Max Po.), Route Schichten (Route) und Detektierungs Header (Header). Die Tabelle umfasst Informationen zu den Schichttypen, Filtergrößen und den Dimensionen der Ein- und Ausgaben.

Nr	Typ	Filt.	Größe	Eingang	Ausgang
0	Conv.	32	3 x 3 / 2	416 x 416 x 3	208 x 208 x 32
1	Conv.	64	3 x 3 / 1	208 x 208 x 32	208 x 208 x 64
2	Route	1			
2	Conv.	64	1 x 1 / 1	208 x 208 x 64	208 x 208 x 64
3	Route	1	2		
4	Max Po.		2 x 2 / 2	208 x 208 x 128	104 x 104 x 64
5	Conv.	128	3 x 3 / 1	104 x 104 x 64	104 x 104 x 128
7	Route	6			
6	Conv.	128	1 x 1 / 1	104 x 104 x 128	104 x 104 x 128
7	Route	5	6		
8	Max Po.		2 x 2 / 2	104 x 104 x 256	52 x 52 x 128
9	Conv.	256	3 x 3 / 1	52 x 52 x 128	52 x 52 x 256
12	Route	11			
10	Conv.	128	3 x 3 / 1	52 x 52 x 256	52 x 52 x 128
14	Route	13			
11	Conv.	256	1 x 1 / 1	52 x 52 x 128	52 x 52 x 256
12	Route	8	11		
13	Max Po.		2 x 2 / 2	52 x 52 x 384	26 x 26 x 256
14	Conv.	512	3 x 3 / 1	26 x 26 x 256	26 x 26 x 512
15	Conv.	30	1 x 1 / 1	26 x 26 x 512	26 x 26 x 30
16	Header 1				

Tabelle H.3: Detaillierte Architektur des Kleines YOLO-Netzwerks. Diese Tabelle präsentiert die sequenzielle Konfiguration der Netzwerkschichten, einschließlich Faltungsschichten (Conv.), Max Pooling Schichten (Max Po.), Route Schichten (Route), Upsampling Schichten (Ups.) und Detektierungs Header (Header). Die Tabelle umfasst Informationen zu den Schichttypen, Filtergrößen und den Dimensionen der Ein- und Ausgaben.

Nr.	Typ	Filt.	Größe	Eingang	Ausgang
0	Conv.	32	3 x 3 / 2	416 x 416 x 3	208 x 208 x 32
1	Conv.	64	3 x 3 / 2	208 x 208 x 32	104 x 104 x 64
2	Conv.	64	3 x 3 / 1	104 x 104 x 64	104 x 104 x 64
4	Conv.	32	3 x 3 / 1	104 x 104 x 64	104 x 104 x 32
5	Conv.	32	3 x 3 / 1	104 x 104 x 32	104 x 104 x 32
6	Route	5	4		
7	Conv.	64	1 x 1 / 1	104 x 104 x 64	104 x 104 x 64
8	Route	2	7		
9	Max Po.		2 x 2 / 2	104 x 104 x 128	52 x 52 x 64
10	Conv.	128	3 x 3 / 1	52 x 52 x 64	52 x 52 x 128
11	Route	10			
12	Conv.	64	3 x 3 / 1	52 x 52 x 128	52 x 52 x 64
13	Conv.	64	3 x 3 / 1	52 x 52 x 64	52 x 52 x 64
14	Route	13	12		
15	Conv.	128	1 x 1 / 1	52 x 52 x 128	52 x 52 x 128
16	Route	10	15		
17	Max Po.		2 x 2 / 2	52 x 52 x 256	26 x 26 x 128
18	Conv.	256	3 x 3 / 1	26 x 26 x 128	26 x 26 x 256
19	Route	18			
20	Conv.	128	3 x 3 / 1	26 x 26 x 256	26 x 26 x 128
21	Conv.	128	3 x 3 / 1	26 x 26 x 128	26 x 26 x 128
22	Route	21	20		
23	Conv.	256	1 x 1 / 1	26 x 26 x 256	26 x 26 x 256
24	Route	18	23		
25	Max Po.		2 x 2 / 2	26 x 26 x 512	13 x 13 x 256
26	Conv.	512	3 x 3 / 1	13 x 13 x 256	13 x 13 x 512
27	Conv.	256	1 x 1 / 1	13 x 13 x 512	13 x 13 x 256
28	Conv.	512	3 x 3 / 1	13 x 13 x 256	13 x 13 x 512
29	Conv.	30	1 x 1 / 1	13 x 13 x 512	13 x 13 x 30
30	Header 2				
31	Route	27			

Nr	Typ	Filt.	Größe	Eingang	Ausgang
32	Conv.	128	1 x 1 / 1	13 x 13 x 256	13 x 13 x 128
33	Ups.	* 2		13 x 13 x 128	26 x 26 x 128
34	Route	33	23		
35	Conv.	256	3 x 3 / 1	26 x 26 x 384	26 x 26 x 256
36	Conv.	30	1 x 1 / 1	26 x 26 x 256	26 x 26 x 30
37	Header 3				

Das für Complex-YOLO verwendete Netzwerk basiert auf YOLO Version 4 [Boch20]. YOLO Version 4 verwendet CSPDarknet53 als Backbone. CSPDarknet53 werden Cross-Stage-Partial-Blocks (CSP-Blocks) verwendet [Wang20a]. Im CSP-Block wird die Feature-Map in zwei Teile geteilt. Auf einen Teil der Merkmalskarten werden weitere Faltungsschichten angewandt, der andere Teil bleibt unverändert. Die beiden Teile werden wieder zusammengeführt, indem eine Route-Schicht verwendet wird, die die Merkmalskarten konkateniert. Zwischen bestimmten Schichten werden Skip Connections eingefügt, um das Verschwinden des Gradientenproblems zu verhindern und die Merkmals-Propagation im Netzwerk zu verbessern. Die Schichten 2 bis 9 sind ein Beispiel für ein CSP-Block. Das CSPDarknet53 erstreckt sich insgesammt von der Schicht 2 bis 107. Darauf folgt ein Spatial Pyramid Pooling (SPP) Block. SPP wird verwendet, um das rezeptive Feld zu vergrößern und die wichtigsten Merkmale vom Backbone zu trennen [He15]. Das SPP erstreckt sich von Schicht 108 bis 113. Path Aggregation Network (PANet) wird als Neck für die Feature-Aggregation aus verschiedenen Ebenen des Backbones für drei verschiedene Detektorebenen verwendet [Liu18]. Auf diese Weise trägt PANet dazu bei, den Kontext der Merkmale auf verschiedenen Skalen und Ebenen zu integrieren und damit die Leistung zu verbessern.

Tabelle H.4: Detaillierte Architektur des YOLO-Netzwerks. Diese Tabelle präsentiert die sequenzielle Konfiguration der Netzwerkschichten, einschließlich Faltungsschichten (Conv.), Max Pooling Schichten (Max Po.), Route Schichten (Route), Skip Connections (Skip Con.), Upsampling Schichten (Ups.) und Detektierungs Header (Header). Die Tabelle umfasst Informationen zu den Schichttypen, Filtergrößen und den Dimensionen der Ein- und Ausgaben.

Nr	Typ	Filt.	Größe	Eingang	Ausgang
0	Conv.	32	3 x 3 / 1	608 x 608 x 3	608 x 608 x 32
1	Conv.	64	3 x 3 / 2	608 x 608 x 32	304 x 304 x 64
2	Conv.	64	1 x 1 / 1	304 x 304 x 64	304 x 304 x 64
3	Route	1			
4	Conv.	64	1 x 1 / 1	304 x 304 x 64	304 x 304 x 64
5	Conv.	32	1 x 1 / 1	304 x 304 x 64	304 x 304 x 32
6	Conv.	64	3 x 3 / 1	304 x 304 x 32	304 x 304 x 64
7	Skip Con.	4			
8	Conv.	64	1 x 1 / 1	304 x 304 x 64	304 x 304 x 64
9	Route	8	2		
10	Conv.	64	1 x 1 / 1	304 x 304 x 128	304 x 304 x 64
11	Conv.	128	3 x 3 / 2	304 x 304 x 64	152 x 152 x 128
12	Conv.	64	1 x 1 / 1	152 x 152 x 128	152 x 152 x 64
13	Route	11			
14	Conv.	64	1 x 1 / 1	152 x 152 x 128	152 x 152 x 64
15	Conv.	64	1 x 1 / 1	152 x 152 x 64	152 x 152 x 64
16	Conv.	64	3 x 3 / 1	152 x 152 x 64	152 x 152 x 64
17	Skip Con.	14			
18	Conv.	64	1 x 1 / 1	152 x 152 x 64	152 x 152 x 64
19	Conv.	64	3 x 3 / 1	152 x 152 x 64	152 x 152 x 64
20	Skip Con.	17			
21	Conv.	64	1 x 1 / 1	152 x 152 x 64	152 x 152 x 64
22	Route	21	12		
23	Conv.	128	1 x 1 / 1	152 x 152 x 128	152 x 152 x 128
24	Conv.	256	3 x 3 / 2	152 x 152 x 128	76 x 76 x 256
25	Conv.	128	1 x 1 / 1	76 x 76 x 256	76 x 76 x 128
26	Route	24			
27	Conv.	128	1 x 1 / 1	76 x 76 x 256	76 x 76 x 128
28	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
29	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
30	Skip Con.	27			

Nr	Typ	Filt.	Größe	Eingang	Ausgang
31	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
32	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
33	Skip Con.	30			
34	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
35	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
36	Skip Con.	33			
37	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
38	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
39	Skip Con.	36			
40	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
41	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
42	Skip Con.	39			
43	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
44	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
45	Skip Con.	42			
46	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
47	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
48	Skip Con.	45			
49	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
50	Conv.	128	3 x 3 / 1	76 x 76 x 128	76 x 76 x 128
51	Skip Con.	48			
52	Conv.	128	1 x 1 / 1	76 x 76 x 128	76 x 76 x 128
53	Route	52	25		
54	Conv.	256	1 x 1 / 1	76 x 76 x 256	76 x 76 x 256
55	Conv.	512	3 x 3 / 2	76 x 76 x 256	38 x 38 x 512
56	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
57	Route	55			
58	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
59	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
60	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
61	Skip Con.	58			
62	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
63	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
64	Skip Con.	61			
65	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
66	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
67	Skip Con.	64			

Nr	Typ	Filt.	Größe	Eingang	Ausgang
68	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
69	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
70	Skip Con.	67			
71	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
72	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
73	Skip Con.	70			
74	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
75	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
76	Skip Con.	73			
77	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
78	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
79	Skip Con.	76			
80	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
81	Conv.	256	3 x 3 / 1	38 x 38 x 256	38 x 38 x 256
82	Skip Con.	79			
83	Conv.	256	1 x 1 / 1	38 x 38 x 256	38 x 38 x 256
84	Route	83	56		
85	Conv.	512	1 x 1 / 1	38 x 38 x 512	38 x 38 x 512
86	Conv.	1024	3 x 3 / 2	38 x 38 x 512	19 x 19 x 1024
87	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
88	Route	86			
89	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
90	Conv.	512	1 x 1 / 1	19 x 19 x 512	19 x 19 x 512
91	Conv.	512	3 x 3 / 1	19 x 19 x 512	19 x 19 x 512
92	Skip Con.	89			
93	Conv.	512	1 x 1 / 1	19 x 19 x 512	19 x 19 x 512
94	Conv.	512	3 x 3 / 1	19 x 19 x 512	19 x 19 x 512
95	Skip Con.	92			
96	Conv.	512	1 x 1 / 1	19 x 19 x 512	19 x 19 x 512
97	Conv.	512	3 x 3 / 1	19 x 19 x 512	19 x 19 x 512
98	Skip Con.	95			
99	Conv.	512	1 x 1 / 1	19 x 19 x 512	19 x 19 x 512
100	Conv.	512	3 x 3 / 1	19 x 19 x 512	19 x 19 x 512
101	Skip Con.	98			
102	Conv.	512	1 x 1 / 1	19 x 19 x 512	19 x 19 x 512
103	Route	102	87		
104	Conv.	1024	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 1024

Nr	Typ	Filt.	Größe	Eingang	Ausgang
105	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
106	Conv.	1024	3 x 3 / 1	19 x 19 x 512	19 x 19 x 1024
107	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
108	Max Po.		5 x 5 / 1	19 x 19 x 512	19 x 19 x 512
109	Route	107			
110	Max Po.		9 x 9 / 1	19 x 19 x 512	19 x 19 x 512
111	Route	107			
112	Max Po.		13 x 13 / 1	19 x 19 x 512	19 x 19 x 512
113	Route	112	110	108	107
114	Conv.	512	1 x 1 / 1	19 x 19 x 2048	19 x 19 x 512
115	Conv.	1024	3 x 3 / 1	19 x 19 x 512	19 x 19 x 1024
116	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
117	Conv.	256	1 x 1 / 1	19 x 19 x 512	19 x 19 x 256
118	Ups.	* 2		19 x 19 x 256	38 x 38 x 256
119	Route	85			
120	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
121	Route	120	118		
122	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
123	Conv.	512	3 x 3 / 1	38 x 38 x 256	38 x 38 x 512
124	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
125	Conv.	512	3 x 3 / 1	38 x 38 x 256	38 x 38 x 512
126	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
127	Conv.	128	1 x 1 / 1	38 x 38 x 256	38 x 38 x 128
128	Ups.	* 2		38 x 38 x 128	76 x 76 x 128
129	Route	54			
130	Conv.	128	1 x 1 / 1	76 x 76 x 256	76 x 76 x 128
131	Route	130	128		
132	Conv.	128	1 x 1 / 1	76 x 76 x 256	76 x 76 x 128
133	Conv.	256	3 x 3 / 1	76 x 76 x 128	76 x 76 x 256
134	Conv.	128	1 x 1 / 1	76 x 76 x 256	76 x 76 x 128
135	Conv.	256	3 x 3 / 1	76 x 76 x 128	76 x 76 x 256
136	Conv.	128	1 x 1 / 1	76 x 76 x 256	76 x 76 x 128
137	Conv.	256	3 x 3 / 1	76 x 76 x 128	76 x 76 x 256
138	Conv.	30	1 x 1 / 1	76 x 76 x 256	76 x 76 x 30
139	Header 4				
140	Route	136			
141	Conv.	256	3 x 3 / 2	76 x 76 x 128	38 x 38 x 256

Nr	Typ	Filt.	Größe	Eingang	Ausgang
142	Route	141	126		
143	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
144	Conv.	512	3 x 3 / 1	38 x 38 x 256	38 x 38 x 512
145	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
146	Conv.	512	3 x 3 / 1	38 x 38 x 256	38 x 38 x 512
147	Conv.	256	1 x 1 / 1	38 x 38 x 512	38 x 38 x 256
148	Conv.	512	3 x 3 / 1	38 x 38 x 256	38 x 38 x 512
149	Conv.	30	1 x 1 / 1	38 x 38 x 512	38 x 38 x 30
150	Header 5				
151	Route	147			
152	Conv.	512	3 x 3 / 2	38 x 38 x 256	19 x 19 x 512
153	Route	152	116		
154	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
155	Conv.	1024	3 x 3 / 1	19 x 19 x 512	19 x 19 x 1024
156	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
157	Conv.	1024	3 x 3 / 1	19 x 19 x 512	19 x 19 x 1024
158	Conv.	512	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 512
159	Conv.	1024	3 x 3 / 1	19 x 19 x 512	19 x 19 x 1024
160	Conv.	30	1 x 1 / 1	19 x 19 x 1024	19 x 19 x 30
161	Header 6				

H.5 Verlustfunktion

In YOLO wird eine Multi-Task-Verlustfunktion verwendet, die sich aus drei Verlustkomponenten zusammensetzt:

- Bounding Box Verlust: Diese Verlustfunktion misst die Abweichung zwischen der vorhergesagten Bounding Box $(\hat{x}_{(i,j)},\hat{y}_{(i,j)},\hat{w}_{(i,j)},\hat{h}_{(i,j)})$ des Objektbereichs und der tatsächlichen Bounding Box (x_i, y_i, w_i, h_i).
- Objekt-/Kein-Objekt-Konfidenz-Verlust: Dieser Verlust misst die Abweichung zwischen der vorhergesagten Konfidenz, dass ein bestimmter Bereich ein Objekt $\hat{C}_{(i,j)}$ enthält, und dem tatsächlichen Vorhandensein eines Objekts C_i bzw. keinem Objekt.

- Klassenverlust: Dieser Verlust misst die Abweichung zwischen der vorhergesagten Konfidenz $\hat{p}_i(c)$ für jede Klasse und der tatsächlichen Klasse $p_i(c)$ des Objekts.

Mathematisch lässt sich die Verlustfunktion wie folgt formulieren:

$$\begin{aligned}\mathcal{L}_{YOLO} &= \lambda_{coord} \sum_{i=0}^{G^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(x_i - \hat{x}_{(i,j)})^2 + (y_i - \hat{y}_{(i,j)})^2 \right] \\ &+ \lambda_{coord} \sum_{i=0}^{G^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(\sqrt{w_i} - \sqrt{\hat{w}_{(i,j)}})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_{(i,j)}})^2 \right] \\ &+ \sum_{i=0}^{G^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_{(i,j)})^2 + \lambda_{noobj} \sum_{i=0}^{G^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_{(i,j)})^2 \\ &+ \sum_{i=0}^{G^2} \mathbb{1}_{i}^{obj} \sum_{c \in Class} (p_i(c) - \hat{p}_i(c))^2,\end{aligned}$$

wobei:

- $\mathbb{1}_{ij}^{obj} = \begin{cases} 1, \text{falls sich in der Gitterzelle } i \text{ ein Objekt befindet und die Bounding Box } j \text{ für die Vorhersage "verantwortlich" ist (d.h. die Bounding Box } j \text{ in Gitterzellen } i \text{ die größte IoU mit dem entsprechendem Ground Truth Objekt hat)} \\ 0, \text{sonst} \end{cases}$
- $\mathbb{1}_{i}^{obj}$: 1, wenn ein Objekt in Gitterzellen i ist, sonst 0.
- $\mathbb{1}_{i}^{noobj}$: 1, wenn sich kein Objekt in Gitterzellen i befindet, sonst 0.
- $\lambda_{coord}, \lambda_{noobj}$ sind Gewichte für den Bounding-Box-Verlust bzw. den No-Object-Confidence-Loss.
- $Class$: sind alle existierenden Klassen.

Nach der Veröffentlichung von YOLO Version 1 [Redm16] werden die Werte $\lambda_{coord} = 5$ und $\lambda_{noobj} = 0.5$ vorgeschlagen.

Im Rahmen des Complex-YOLO Algorithmus setzt sich die Verlustfunktion $\mathcal{L}_{total}$ aus mehreren Komponenten zusammen, die unterschiedliche Aspekte der Vorhersage bewerten:

- Position und Größe der Bounding Boxen
- Objektvertrauen
- Klassifikationsverlust
- Orientierungsverlust

Die ersten drei Komponenten sind Teil der im ursprünglichen YOLO-Algorithmus definierten Verlustfunktion $\mathcal{L}_{YOLO}$.

Der Orientierungsverlust $\mathcal{L}_{Euler}$ misst die Abweichung zwischen der vorhergesagten ($\hat{t}_{im}$,$\hat{t}_{re}$) und der tatsächlichen (t_{im},t_{re}) Orientierung der Objekte. Dieser Verlust wird wie folgt berechnet:

$$\mathcal{L}_{Euler} = \sum_{i=0}^{G^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} \left[(t_{im,i} - \hat{t}_{im,i,j})^2 + (t_{re,i} - \hat{t}_{re,i,j})^2 \right]$$

Hierbei ist $\mathbb{1}_{ij}^{obj}$ ein Indikator, der den Wert 1 annimmt, wenn im i-ten Zellbereich des Eingangsbildes, aufgeteilt in ein $G \times G$ Raster, ein Objekt vorhergesagt wird. B steht für die Anzahl der Bounding Boxen pro Rasterzelle.

Somit ergibt sich die gesamte Verlustfunktion bei Complex-YOLO $\mathcal{L}_{total}$ als Summe aus dem YOLO-spezifischen Verlust $\mathcal{L}_{YOLO}$ und dem Orientierungsverlust $\mathcal{L}_{Euler}$:

$$\mathcal{L}_{total} = \mathcal{L}_{YOLO} + \mathcal{L}_{Euler}.$$

H.6 Parameter

Dieses Abschnitt beschreibt in Tabelle H.5 weitere zuvor nicht aufgeführten Parameter der Implementierungen von Complex-YOLO.

Tabelle H.5: Weitere Parameter der Complex-YOLO Implementation, welche zuvor noch nicht genannt wurden.

Batch-Größe	8
IoU-Schwellwert	0,5
NMS-Schwellwert	0,5
Konfidenz-Schwell.	0,5
Optimierer	Adam
Momentum	0,949
Lernrate	0,001
Minimale Lernrate	0,0000001
Lernraten-Scheduler	Kosinus-Annealing
Gewichtsregularisierung	0,0005
Burn In	50
Burn In Grenzen	1500, 4000

I Modell für die Simulation von Nebel bei LiDAR Sensoren

In diesem Kapitel wird ein Modell zur Simulation der Auswirkungen von Nebel auf die Leistung von LiDAR-Sensoren vorgestellt. Die Beschreibung basiert auf den Arbeiten [Hahn21, Rass11].

Die empfangene Leistung $P(R)$ eines Sensors von einem Laserstrahl, der in einer Entfernung R reflektiert wird, kann mit der Sendeenergie E_P des Lasers durch folgende vereinfachte Formel ausgedrückt werden:

$$P(R) = C_A \cdot E_P \cdot H(R).$$

Dabei ist C_A eine systemspezifische Konstante, welche von den Eigenschaften des verwendeten Sensors abhängt, und $H(R)$ repräsentiert die Modellierung der Impulsantwort, die in zwei Komponenten unterteilt ist:

$$H(R) = H_C(R) \cdot H_T(R).$$

Komponente der Übertragung: $H_C(R)$

Die Komponente $H_C(R)$, definiert durch:

$$H_C(R) = \frac{T^2(R)}{R^2}\zeta(R),$$

berücksichtigt die Beeinflussung des Laserimpulses beim Durchgang durch ein Medium. Die Übertragungsfunktion $T(R)$ gibt den Verlust durch das Medium

an und wird als:

$$T(R) = \exp\left(-\int_0^R \alpha(r)dr\right),$$

definiert, wobei $\alpha(R)$ den räumlich variierenden Dämpfungskoeffizienten des Mediums darstellt. Der Dämpfungskoeffizient $\alpha(r)$, auch als Extinktionskoeffizient bekannt, kann sich unter ungünstigen Wetterbedingungen wie Nebel durch Absorption und Streuung signifikant erhöhen.

Es wird eine homogene optische Beschaffenheit des Mediums angenommen, was zu einem konstanten Wert von $\alpha(R) = \alpha$ führt. Daraus folgt die vereinfachte Form:

$$T(R) = \exp(-\alpha R).$$

Zielimpulsantwort: $H_T(R)$

Die Zielimpulsantwort $H_T(R)$, die beschreibt, wie der Laserimpuls vom Ziel reflektiert wird, ist in der Nebelsimulation definiert als:

$$H_T(R) = \beta U(R_0 - R) + \beta_0 \delta(R - R_0) = H_T^{\text{soft}} + H_T^{\text{hard}}.$$

Hierbei stellt H_T^{soft} den Einfluss des Nebels auf die Rückstreuung dar, während H_T^{hard} die direkte Reflexion vom Ziel mit der Reflexionsfähigkeit β_0 in der Entfernung R_0 beschreibt. Der Rückstreukoeffizient β steigt mit zunehmender Nebeldichte, was zu einem erhöhten Anteil der Rückstreuung an der gesamten Impulsantwort führt.

Die meteorologische Sichtweite, auch bekannt als Meteorological Optical Range (MOR), bezieht sich auf die maximale Distanz, bei der ein Beobachter ein schwarzes Objekt vor einem hellen Hintergrund gerade noch erkennen kann. Sie ist ein Maß für die Transparenz der Atmosphäre und wird häufig in der Meteorologie und Luftfahrt verwendet, um die Qualität der Sichtverhältnisse zu beschreiben [Kim18]. Anhand des Dämpfungskoeffizient α kann die MOR

bestimmt werden:

$$\mathrm{MOR} = \frac{\ln(20)}{\alpha}.$$

In der Simulation wird der Rückstreukoeffizient β durch den MOR bestimmt:

$$\beta = \frac{0.046}{\mathrm{MOR}}.$$

J Modell für die Simulation von Schnee bei LiDAR Sensoren

In diesem Kapitel wird ein Modell zur Simulation der Auswirkungen von Schnee auf die Leistung von LiDAR-Sensoren vorgestellt. Die Beschreibung basiert auf den Arbeiten [Hahn22].

Die empfangene Leistung $P(R)$ eines Sensors von einem Laserstrahl, der in einer Entfernung R reflektiert wird, kann durch folgende vereinfachte Formel ausgedrückt werden:

$$P(R) = C_A \cdot P_0 \rho_0 \cdot \frac{\cos(\alpha_{in})}{R^2},$$

wobei der ausgesendete Laserimpuls P_0 von einem festen Objekt mit einem Reflexionsvermögen ρ_0 reflektiert und vom Empfänger erfasst wird.

$H(R)$ repräsentiert die Modellierung der Impulsantwort, die in zwei Komponenten unterteilt ist:

$$H(R) = H_C(R) \cdot H_T(R).$$

Die **Komponente der Übertragung** $H_C(R)$, definiert durch:

$$H_C(R) = \frac{T^2(R)}{R^2} \zeta(R),$$

berücksichtigt die Beeinflussung des Laserimpulses beim Durchgang durch ein Medium.

Die Übertragungsfunktion $T(R)$ ist in dem Teil des Mediums, der nicht von Schneepartikeln besetzt ist, gleich 1, wobei davon ausgegangen wird, dass es keine anderen Streuer gibt. Die Überlappung $\zeta(R)$ wird definiert als:

$$\zeta(R) := \begin{cases} 0, & \text{wenn } R \leq R_1 \\ \frac{R-R_1}{R_2-R_1}, & \text{wenn } R_1 < R < R_2 \\ 1, & \text{wenn } R \geq R_2 \end{cases}.$$

Die **Zielimpulsantwort** $H_T(R)$, die beschreibt, wie der Laserimpuls vom Ziel reflektiert wird, ist in der Schneesimulation definiert als:

$$H_T(R) = \rho_0 \delta(R - R_0),$$

wobei ρ_0 das Reflexionsvermögen des Objekts und δ die Dirac-Delta-Funktion ist. Bei Schneefall wird der Laserstrahl jedoch nicht nur von festen Zielobjekten reflektiert, sondern auch teilweise von Schneepartikeln.

K Astyx Dataset

In diesem Appendix werden ausgewählte Beispiele aus dem Astyx-Datensatz präsentiert, um einen Eindruck von der Art der Daten und deren Eigenschaften zu vermitteln.

Der Astyx-Datensatz ist ein bedeutender Datensatz im Bereich der autonomen Fahrzeuge und Sensorfusion. Er enthält umfangreiche Daten von verschiedenen Sensoren wie hochauflösendem Radar, LiDAR und RGB-Kameras. Dieser Datensatz wird verwendet, um Algorithmen für die Sensorfusion und Objekterkennung zu entwickeln und zu testen [Meye19].

Der Astyx HiRes Datensatz wurde 2019 von der Astyx GmbH veröffentlicht [Meye19]. Der Datensatz enthält Messungen der folgenden Sensoren:

- Kamera: Point Grey Blackfly (Auflösung: 2048×618 Bildpunkten),
- LiDAR: Velodyne VLP-16 (Kanäle: 16 , vertikales Sichtfeld: 30°, horizontales Sichtfeld: 360°),
- RADAR: Astyx 6455 HiRes (vertikales Sichtfeld: 10°, horizontales Sichtfeld: 110°).

Es sei darauf hingewiesen, dass der RADAR-Sensor von der Firma Astyx selbst entwickelt wurde und in diesem Datensatz demonstriert wird. Der Datensatz war zu Beginn dieser Arbeit öffentlich über Astyx verfügbar, wurde aber in der Zwischenzeit wieder entfernt und ist über offizielle Quellen nicht mehr verfügbar.

Die Daten wurden in Ottobrunn (südlich von München, Deutschland) aufgezeichnet. Es wurden sowohl Industrie- und Wohngebiete als auch Landstraßen

befahren. Das Wetter war während der Aufnahmen sonnig oder bewölkt, so dass die Sensoren nicht negativ beeinflusst wurden.

Insgesamt umfasst der Datensatz 546 Datenelemente und die folgenden Klassen: 'Car', 'Truck', 'Bus', 'Person', 'Trailer', 'Cyclist', 'Motorcyclist', 'Towed Object' und 'Other Vehicle'. Mit 2941 Objekten ist jedoch nur die Klasse 'Car' in ausreichender Anzahl vorhanden, um einen Objektdetektor zu trainieren. Im Durchschnitt befinden sich 5,39 Objekte der Klasse 'Car' in einer Aufnahme und maximal 24. Solche hohen Werte ergeben sich bei Aufnahmen auf Parkplätzen, da dort viele Objekte eng geparkt sind.

Für jeden synchronen Zeitpunkt gibt es eine eindeutige ID und für jeden Sensor sowie für die Objektlabel und Kalibrierdaten einzelne Unterordner existieren. Die LiDAR-Daten werden jeweils als eine sechsdimensionale Punktwolke (*x, y, z, Reflectivity, LaserID, Timestamp*) beschrieben:

- (*x, y, z*) - gibt die Lage des Messpunktes im Raum an, mit dem Sensor als Ursprung.
- *Reflectivity* - gibt den Intensitätswert an, wie stark der Laserstrahl reflektiert wurde.
- *LaserID* - ordnet jedem Punkt eindeutig den zugehörigen Channel zu.
- *Timestamp* - gibt den Zeitpunkt an, zu dem der Laserstrahl vom Sensor empfangen wurde.

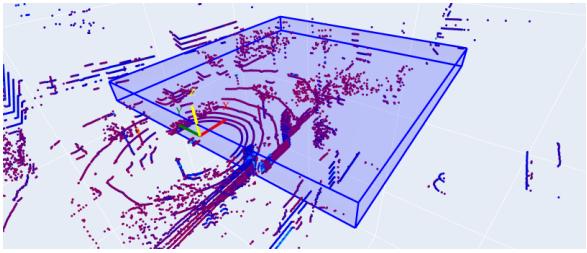


Abbildung K.1: Beispiel für die LiDAR Daten aus dem Astyx Datensatz. In Blau ist die Region von Interesse dargestellt mit $[0, 50] \times [-25, 25] \times [-2,2]$.

Die RADAR-Daten werden jeweils als eine fünfdimensionale Punktwolke (x, y, z, V_r, *magnitude*) beschrieben:

- (x, y, z) - gibt die Lage des Messpunktes im Raum an, mit dem Sensor als Ursprung.
- v_r - ist die relative Radialgeschwindigkeit in $\frac{km}{h}$ des Objekts, von dem die Radarwelle reflektiert wird.
- *magnitude* - gibt an, wie stark die Radarwelle reflektiert wird.

Ein Beispiel für die LiDAR Daten des Datensatzes ist in Abbildung K.1 dargestellt. Weiterhin ist in der Abbildung die Region von Interesse dargestellt, welche $[0, 50] \times [-25, 25] \times [-2,2]$ beträgt.

Abbildung K.2 zeigt die Kameraaufnahmen der Beispiele. Es sind drei Szenen dargestellt, die häufig im Datensatz vorkommen:

- Wohngebiet,
- Parkplatz,
- Landstraße.

Die Dargestellten Punktwolken in den Abbildungen K.3 -K.5 beziehen sich jeweils auf diese drei Szenen.

Abbildung K.2: Beispiel Kamera aufnahmen aus dem Astyx HiRes Datensatz.

In Abbildung K.3 sind Beispiele für LiDAR-Punktwolken gegeben, wobei die Farbe den *Reflectivity*-Wert codiert. Dies zeigt deutlich, dass Nummernschilder oder Fahrbahnbeschilderungen gut reflektieren, wohingegen Straßen oder Vegetation schlechte Reflektoren sind. Alle Punkte befinden sich im Koordinatensystem des RADAR-Sensors.

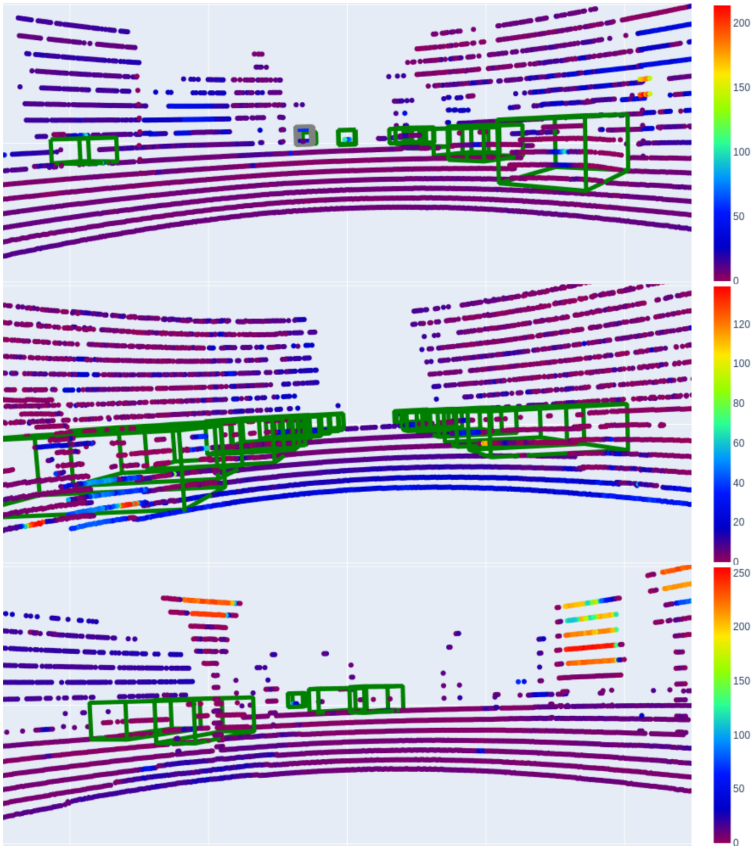


Abbildung K.3: Beispiel LiDAR Punktwolken aus dem Astyx HiRes Datensatz. Die Farbe codiert die *Reflectivity*.

In den Abbildungen K.4 und K.5 sind Beispiele für RADAR-Punktwolken dargestellt. Die Farbe codiert dabei den Wert der relativen radialen Geschwindigkeit bzw. die *Magnitude*.

Im obersten Bild von Abbildung K.4 fährt das Aufnahmefahrzeug, während die Szenerie statisch ist, wodurch sich der Wert der relativen radialen Geschwindigkeit gleichmäßig ändert. Im mittleren Bild bewegt sich das Aufnahmefahrzeug nicht und die Szenerie ist statisch, wodurch die Werte der relativen radialen

Geschwindigkeit nahe bei 0 sind. Im unteren Bild bewegt sich das Aufnahmefahrzeug nicht, während sich die anderen Fahrzeuge bewegen, wodurch sich diese Fahrzeuge sehr gut vom statischen Hintergrund separieren lassen.

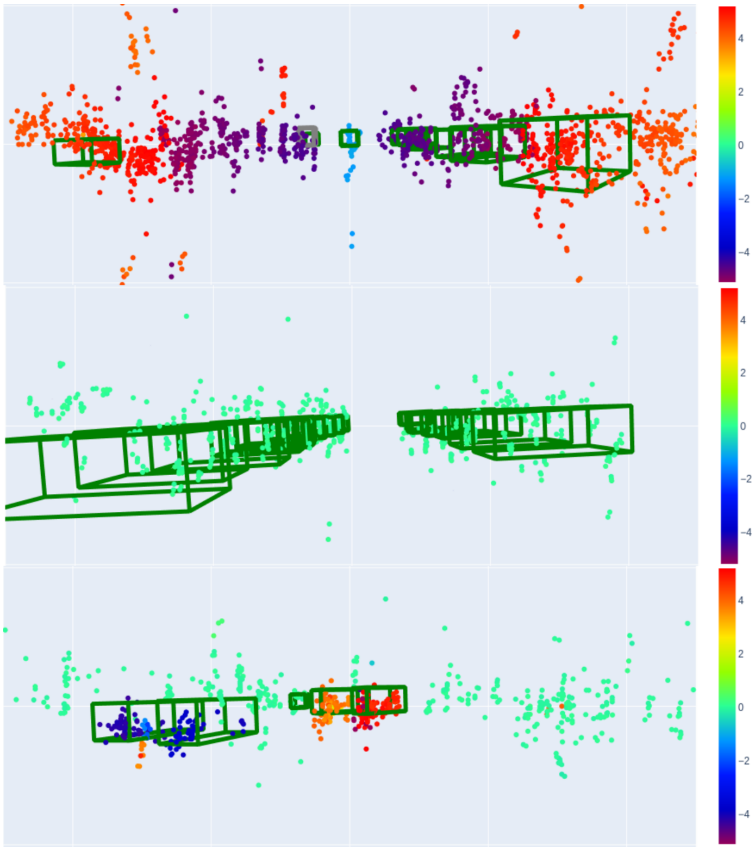


Abbildung K.4: Beispiel RADAR-Punktwolken aus dem Astyx HiRes Datensatz. Die Farbe codiert die relative radiale Geschwindigkeit.

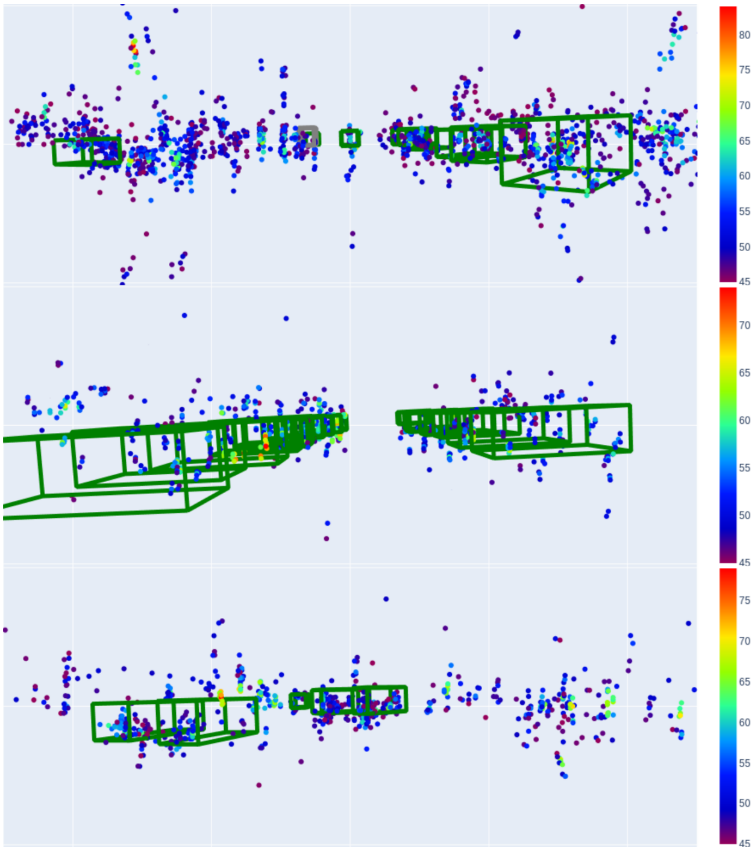


Abbildung K.5: Beispiel RADAR-Punktwolken aus dem Astyx HiRes Datensatz. Die Farbe codiert die *Magnitude*.

L Audi Autonomous Driving Dataset

Der A2D2-Datensatz (Audi Autonomous Driving Dataset) [Geye20] ist ein umfassender Datensatz, der von der Audi AG veröffentlicht wurde, um die Forschung im Bereich des autonomen Fahrens zu fördern. Dieser Datensatz besteht aus simultan aufgezeichneten Bildern und Punktwolken, ergänzt durch 3D-Bounding Boxen, semantische Segmentierung, Instanzsegmentierung und Daten, die vom Fahrzeugsystembus extrahiert wurden. Der verwendete Sensorsatz besteht aus sechs Kameras und fünf LiDAR-Einheiten, die eine vollständige 360-Grad-Abdeckung bieten. Insgesamt wurden fünf Velodyne VLP-16 LiDAR-Sensoren verwendet, die strategisch auf dem Fahrzeugdach eines Audi Q7 e-tron montiert wurden.

Die genauen Positionen der LiDAR-Sensoren sind wie folgt:

- Front-Center: Ein LiDAR-Sensor befindet sich in der Mitte der Vorderseite des Fahrzeugs.
- Front-Left und Front-Right: Zwei weitere LiDAR-Sensoren sind links und rechts vorne auf dem Fahrzeugdach montiert. Diese seitlichen Sensoren erweitern das Sichtfeld und ermöglichen eine bessere Erkennung von Objekten, die sich seitlich zum Fahrzeug befinden
- Rear-Left und Rear-Right: Diese beiden LiDAR-Sensoren sind links und rechts hinten auf dem Fahrzeugdach positioniert. Diese Sensoren überwachen den hinteren Bereich des Fahrzeugs.

Die Sensoren wurden symmetrisch entlang der x-z-Ebene des Fahrzeugs platziert, um eine gleichmäßige Verteilung des Sichtfeldes und eine maximale Überlappung der Sensorabdeckungen zu erreichen. Die genaue Ausrichtung und Positionierung der Sensoren wurden manuell mithilfe von CAD-Software

optimiert, um die toten Winkel zu minimieren und eine maximale Überlappung der LiDAR- und Kamera-Sichtfelder zu gewährleisten. Für ein Beispiel der Daten siehe Abbildung L.1.

Die Anmerkungen im A2D2-Datensatz umfassen nicht-sequenzielle Rahmen: Insgesamt sind 41.277 Rahmen mit semantischer Segmentierung und Punktwolkenbeschriftungen versehen, von denen 12.497 Rahmen auch 3D-Bounding Boxen für Objekte im Sichtfeld der Frontkamera enthalten. Zusätzlich werden 392.556 sequentielle Rahmen unannotierter Sensordaten aus Aufnahmen in drei Städten in Süddeutschland (München, Ingolstadt, Gaimersheim) bereitgestellt.

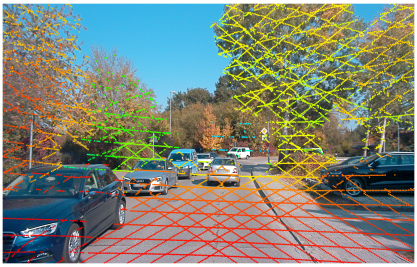


Abbildung L.1: Die Bilder zeigen ein Beispiel des A2D2 Datensatz [Geye20]. Dabei wurden sowohl die LiDAR als auch RADAR Sensordaten auf das Kamerabild projiziert.

M View of Delft Datensatz

Der “’View of Delft” Datensatz, auch bekannt als ”Delft” Datensatz, wurde 2021 von Palffy et al. [Palf22] eingeführt und ist ein multisenorielle Datensatz im Bereich der automobilen Objekterkennung. Dieser Datensatz zeichnet sich durch die Verwendung hochauflösender Sensoren aus, einschließlich eines Velodyne HDL-64 S3 LiDAR, Stereo-Kameras und eines ZF FRGen21 3+1D RADARs. Diese sind synchronisiert und kalibriert, wobei der 3+1D RADAR neben den räumlichen Dimensionen (Reichweite, Azimut und Höhe) auch die Doppler-Dimension erfasst. Für ein Beispiel der Daten siehe Abbildung M.1.

(a) View of Delft LiDAR Beispiel **(b)** View of Delft RADAR Beispiel

Abbildung M.1: Die Bilder zeigen ein Beispiel des View of Delft Datensatz [Palf22]. Dabei wurden sowohl die LiDAR als auch RADAR Sensordaten auf das Kamerabild projiziert.

Die Datenerfassung erfolgte in der Stadt Delft in den Niederlanden, in verschiedenen urbanen Verkehrsszenarien, einschließlich Vororte und Altstadtgebiete, mit einem besonderen Augenmerk auf verletzliche Verkehrsteilnehmer wie Fußgänger und Radfahrer. Der Datensatz umfasst 8.693 Frames mit insgesamt 123.106 3D-Bounding-Box-Annotationen, die verschiedene Objektklassen wie

Fußgänger, Radfahrer und Autos abdecken. Diese umfassenden und detaillierten Annotationen machen ihn zu einem der größten Datensätze, der ein 3+1D RADAR einbezieht und sind besonders wertvoll für die Modellierung und Schulung im Bereich des RADAR-basierten Erkennungssystems.

N Sparse Fuse Dense

Sparse Fuse Dense (SFD) repräsentiert einen transformativen Ansatz in der 3D-Objekterkennung, der darauf ausgelegt ist, die durch spärliche LiDAR-Daten verursachten Limitationen anzugehen [Wu22]. Das Kernstück von SFD ist die Integration von pseudo Punktewolken, die durch Tiefenvervollständigung aus 2D-Bildern gewonnen wurden, mit LiDAR Sensor generierten Punktewolken, wodurch die Detektionsfähigkeiten durch eine reichhaltigere Datenrepräsentation verbessert werden.

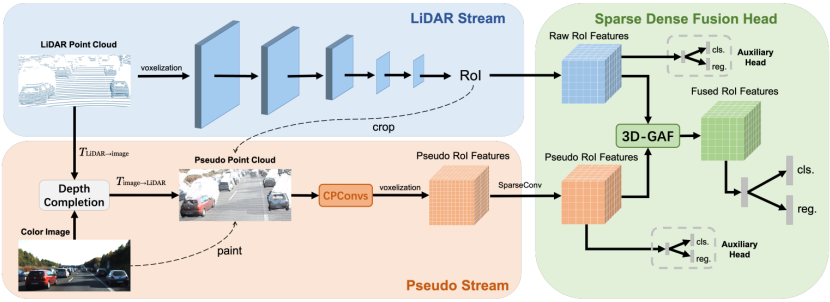
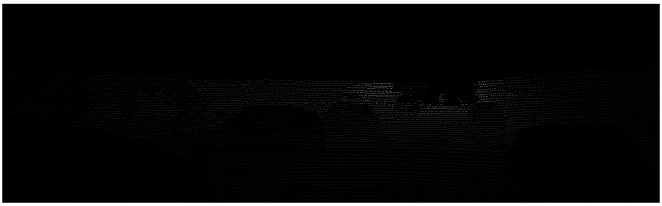


Abbildung N.1: Architektur von Sparse Fuse Dense, entnommen aus [Wu22]. SFD besteht aus drei Hauptkomponenten: LiDAR-Stream, Pseudo-Stream und Sparse Dense Fusion Head. Der LiDAR-Stream verwendet ausschließlich rohe Punktwolken, um 3D-Regionen von Interesse (RoIs) vorherzusagen. Diese RoIs werden dann genutzt, um rohe und pseudo Punktwolken zu beschneiden. Der Pseudo-Stream erzeugt aus rohen Wolken und Bildern pseudo Punktwolken. Durch die Einfärbung der pseudo Punktwolken mit RGB entstehen farbige pseudo Punktwolken. Anschließend werden mehrere CPConvs durchgeführt, um umfangreiche Informationen aus den pseudo Punktwolken in den RoIs zu extrahieren. Am Ende des Pseudo-Streams werden die pseudo Punktwolken in den RoIs voxelisiert und 3D-Sparse-Konvolutionen angewendet. Im Sparse Dense Fusion Head werden die RoI-Merkmale aus rohen und pseudo Punktwolken mittels 3D-GAF fusioniert. Die fusionierten RoI-Merkmale werden dann verwendet.

In Abbildung N.1 ist die Architektur von SFD dargestellt. Die Methodik beginnt mit dem Prozess der Tiefenvervollständigung, bei dem spärliche LiDAR-Daten auf eine 2D-Ebene projiziert werden, um eine spärliche Tiefenkarte zu erstellen. Hierfür verwendet SFD Twin-Surface Extrapolation (TWISe) [Imra21]. TWISe modelliert sowohl die sichtbaren als auch die verdeckten Oberflächen, um eine dichtere Tiefenkarte vorherzusagen. Abbildung N.2 zeigt ein Beispiel für die spärliche Tiefenkarte, die durch direkte LiDAR-Messungen erzeugt wird und die dichte Tiefenkarte, welche durch die Anwendung von TWISe entwickelt wird.



(a) Spärlich Tiefenkarte

(b) Dicht Tiefenkarte

(c) Kamera Daten

Abbildung N.2: N.2a zeigt die spärliche Tiefenkarte, die durch direkte LiDAR-Messungen erzeugt wird. N.2b stellt die dichte Tiefenkarte dar, welche durch die Anwendung von TWISe aus der spärlichen Tiefenkarte entwickelt wird. Zudem ist in N.2c die zugehörige RGB-Bild abgebildet.

Parallel zur Tiefenvervollständigung verarbeitet der LiDAR-Stream die rohen Punktewolken durch Voxelisierung, indem der Raum in Voxel segmentiert und Merkmale innerhalb jedes Voxels aggregiert werden. Gleichzeitig nutzt der Pseudo-Stream, bestehend aus den angereicherten pseudo Punktewolken, Color Point Convolution (CPConv), um geometrische und Farbinformationen zu integrieren.

Für eine Aufnahme von Pseudo-Punktwolken P werden die RGB-Werte (r, g, b) und Koordinaten (u, v) jedes Pixels des Bildes den entsprechenden Pseudo-Punkten zugeordnet. Somit kann der i-te Pseudo-Punkt p_i als $(x_i, y_i, z_i, r_i, g_i, b_i, u_i, v_i)$ dargestellt werden. Ein naiver Ansatz zur Merkmalsextraktion aus Pseudo-Wolken ist die direkte Voxelisierung der Pseudo-Wolken und die Durchführung von 3D-Sparse-Konvolutionen, wobei diese Methode die reichhaltigen semantischen und strukturellen Informationen in Pseudo-Wolken nicht vollständig ausnutzt. CPConv (*Color Point Convolution*) wird zur Merkmalsextraktion verwendet, das Nachbarn im Bildbereich sucht. Für den i-ten pseudo Punkt p_i wird das Merkmal von p_i als $f_i = (x_i,y_i,z_i,r_i,g_i,b_i)$ bezeichnet, das aus 3D-geometrischen Merkmalen (x_i,y_i,z_i) und 2D-semantischen Merkmalen (r_i,g_i,b_i) besteht. Für K Nachbarn von p_i werden deren Positionen gesammelt und Positionsresiduen berechnet. Dann wird eine vollständig verbundene Schicht auf Positionsresiduen angewendet, wobei deren Kanäle auf C_3 angehoben werden, um mit den pseudo Punktmerkmalen übereinzustimmen.Gegeben eine Menge von Nachbarmerkmalen $F_i = \{f_{i,k} \in \mathbb{R}^{C_3}, k \in \{1, \dots, K\}\}$ und eine Menge von Nachbarpositionsresiduen $H_i = \{h_{i,k} \in \mathbb{R}^{C_3}, k \in \{1, \dots, K\}\}$, wird jedes $f_{i,k}$ mit dem entsprechenden $h_{i,k}$ gewichtet. Die gewichteten Nachbarmerkmale werden konkateniert. In der Multi-Level-Feature-Fusion werden drei CPConvs gestapelt, um tiefere Merkmale von Pseudo-Wolken zu extrahieren. In Anbetracht dessen, dass Merkmale auf höherem Niveau ein größeres rezeptives Feld und reichere semantische Informationen bieten, während Merkmale auf niedrigerem Niveau feinere Strukturinformationen liefern, wird der Ausgang jeder CPConv konkateniert, um eine umfassendere und diskriminierendere Darstellung für Pseudo-Wolken zu erhalten.

Die Fusion dieser Streams erfolgt im Sparse Dense Fusion Head, der die 3D Grid-wise Attentive Fusion (3D-GAF) Methode verwendet. Diese Fusionsstrategie kombiniert die Merkmale aus beiden Streams innerhalb eines 3D-Gitterframeworks. Jede Gitterzelle enthält Merkmale aus den entsprechenden räumlichen Regionen der beiden Streams. Durch einen Aufmerksamkeitsmechanismus bewertet und gewichtet 3D-GAF dynamisch die Bedeutung der Merkmale aus jeder Quelle, optimiert die Fusion basierend auf der kontextuellen Relevanz und Qualität der Daten. Dieser Prozess stellt sicher, dass Merkmale aus dem Pseudo-Stream in Regionen, in denen LiDAR-Daten spärlich oder verdeckt sind, hervorgehoben werden.

Für das 3D-GAF sei b ein einzelnes 3D-RoI. $F^{\text{raw}} \in \mathbb{R}^{n \times C}$ und $F^{\text{pse}} \in \mathbb{R}^{n \times C}$ die Merkmale der RoI der Rohwolke bzw. der Pseudowolke in b bezeichnen, wobei n (standardmäßig $6 \times 6 \times 6$, basierend auf Voxel-RCNN) die Gesamtzahl der Gitter in einer 3D-RoI und C der Gittermerkmalskanal ist. Das i-te RoI-Gittermerkmal von F^{raw} und F^{pse} wird als $F^{\text{raw}}_{i,}$ und $F^{\text{pse}}_{i,}$ bezeichnet. Für ein Paar von RoI-Gittermerkmalen $(F^{\text{raw}}_{i,}, F^{\text{pse}}_{i,})$ werden $F^{\text{raw}}_{i,}$ und $F^{\text{pse}}_{i,}$ verbunden. Dann wird das Ergebnis auf eine vollständig verbundene Schicht und eine sigmoide Schicht angewendet, wodurch ein Paar von Gewichten $(w^{\text{raw}}_{i,}, w^{\text{pse}}_{i,})$ für das Paar von Gittermerkmalen erzeugt wird, wobei $w^{\text{raw}}_{i,}$ und $w^{\text{pse}}_{i,}$ alle Skalare sind. Schließlich wird $(F^{\text{raw}}_{i,}, F^{\text{pse}}_{i,})$ mit $(w^{\text{raw}}_{i,}, w^{\text{pse}}_{i,})$ gewichtet, um das fusionierte Gittermerkmal F_i zu erhalten. Formal ergibt sich F_i wie folgt:

$$(w^{\text{raw}}_{i,}, w^{\text{pse}}_{i,}) = \sigma(\text{MLP}(\text{CONCAT}(F^{\text{raw}}_{i,}, F^{\text{pse}}_{i,})))$$

$$F_i = \text{MLP}(\text{CONCAT}(w^{\text{raw}}_{i,} F^{\text{raw}}_{i,}, w^{\text{pse}}_{i,} F^{\text{pse}}_{i,})).$$

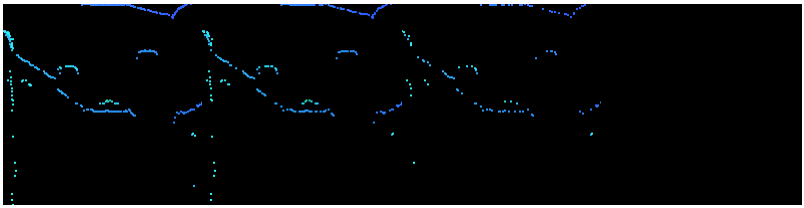
Als Verlustfunktion wird der RPN-Verlust und der RoI-Kopf-Verlust L_{rpn} von Voxel-RCNN L_{roi} gefolgt. Zusätzlich wird der RoI-Kopf-Verlust sowohl für den LiDAR-Stream L_{aux1} als auch für den Pseudo-Stream L_{aux2} hinzugefügt. L_{aux1} und L_{aux2} sind konsistent mit L_{roi}, einschließlich Verlust der Klassenkonfidenz und Regressionsverlust. Der Verlust des Tiefenvervollständigungsnetzwerks L_{depth} folgt der Definition von [12]. Dann ist der Gesamtverlust:

$$L = L_{\text{rpn}} + L_{\text{roi}} + \lambda_1 L_{\text{aux1}} + \lambda_2 L_{\text{aux2}} + \beta L_{\text{depth}},$$

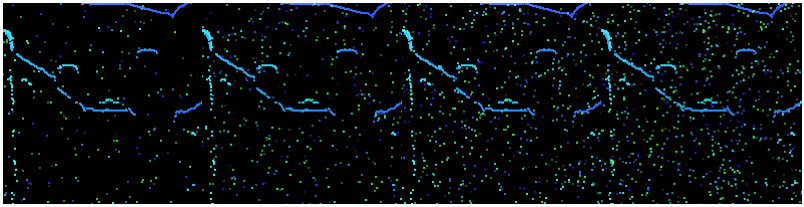
wobei λ_1, λ_2 und β die Gewichte von L_{aux1}, L_{aux2} und L_{depth}. Die Gewichte sind standardmäßig ($\lambda_1 = 0.5$, $\lambda_2 = 0.5$, $\beta = 1$).

O Beispiele für die betrachteten Punktwolkestörungen

Im folgenden Anhang werden Beispiele für die Punktwolkestörungen {"Punkte verlieren Grad 1", ..., "Punkte verlieren Grad 4", "Punkte hinzufügen Grad 1", ..., "Punkte hinzufügen Grad 4", "Punkte verschieben Grad 1", ..., "Punkte verschieben Grad 4", "Information verrauscht Grad 1", ..., "Information verrauscht Grad 4", "Cluster Grad 1", ..., "Cluster Grad 4", "Nebel Grad 1", ..., "Nebel Grad 4", "Schnee"} präsentiert.

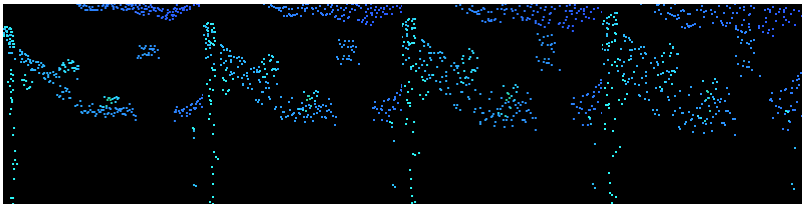


(a) Punkte verlieren Grad 1 **(b)** Punkte verlieren Grad 2 **(c)** Punkte verlieren Grad 3 **(d)** Punkte verlieren Grad 4

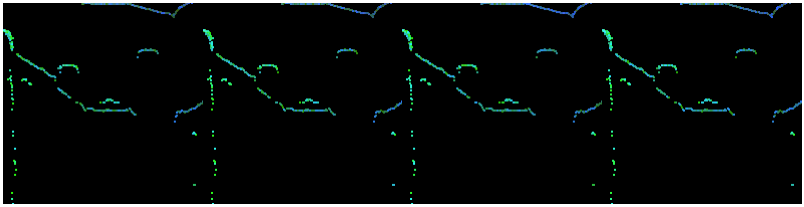


(e) Punkte hinzufügen Grad 1 **(f)** Punkte hinzufügen Grad 2 **(g)** Punkte hinzufügen Grad 3 **(h)** Punkte hinzufügen Grad 4

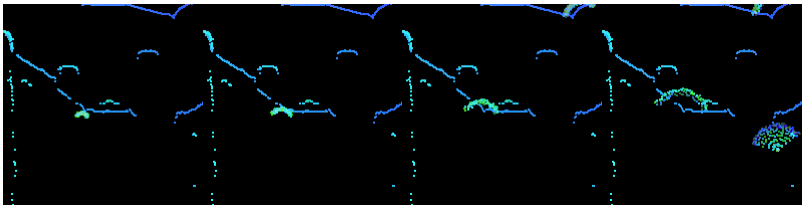
Abbildung O.1: Vergrößerung in einem Beispiel für die Störungen: "Punkte verlieren" und "Punkte hinzufügen" mit den Schwierigkeitsgraden: 1,2,3 und 4 auf LiDAR Daten aus dem Astyx Datensatz. Die Punktwolken sind in BEV dargestellt und in der gleichen Weise wie bei Complex-YOLO sind die Farbkanäle kodiert.



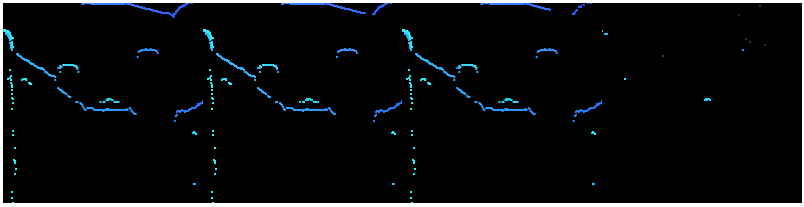
(a) Punkte verschieben Grad 1 **(b)** Punkte verschieben Grad 2 **(c)** Punkte verschieben Grad 3 **(d)** Punkte verschieben Grad 4



(e) Information verrauscht Grad 1 **(f)** Information verrauscht Grad 2 **(g)** Information verrauscht Grad 3 **(h)** Information verrauscht Grad 4

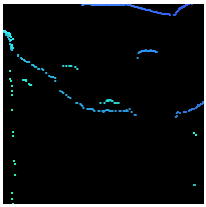


(i) Cluster Grad 1 **(j)** Cluster Grad 2 **(k)** Cluster Grad 3 **(l)** Cluster Grad 4



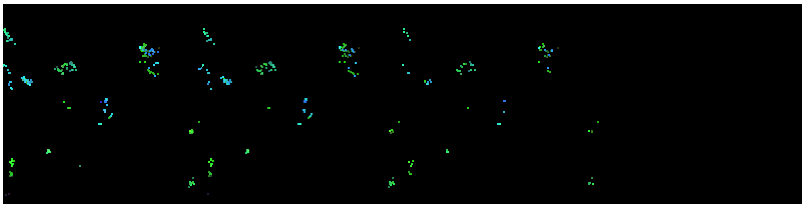
(m) Nebel Grad 1 **(n)** Nebel Grad 2 **(o)** Nebel Grad 3 **(p)** Nebel Grad 4

Abbildung O.2: Vergrößerung in einem Beispiel für die Störungen: ”Punkte verschieben”, ”Informationen verrauscht”, ”Cluster”, und ”Nebel” mit den Schwierigkeitsgraden: 1,2,3 und 4 auf LiDAR Daten aus dem Astyx Datensatz. Die Punktwolken sind in BEV dargestellt und in der gleichen Weise wie bei Complex-YOLO sind die Farbkanäle kodiert.

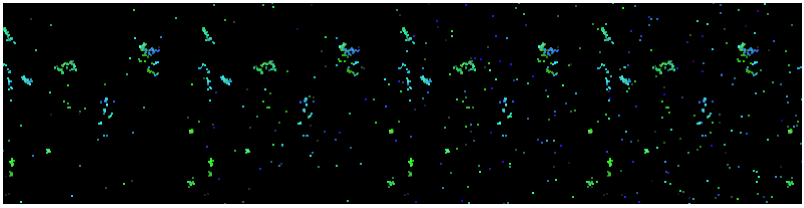


(a) Schnee

Abbildung O.3: Vergrößerung in einem Beispiel für die Störungen: ”Schnee” auf LiDAR Daten aus dem Astyx Datensatz. Die Punktwolken sind in BEV dargestellt und in der gleichen Weise wie bei Complex-YOLO sind die Farbkanäle kodiert.

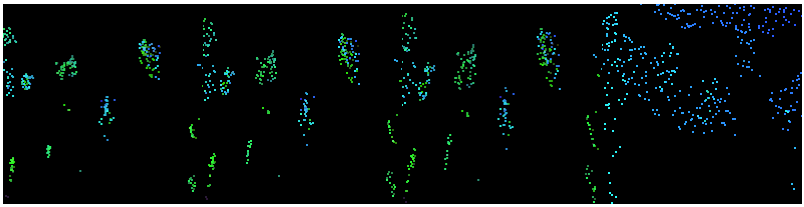


(a) Punkte verlieren Grad 1 **(b)** Punkte verlieren Grad 2 **(c)** Punkte verlieren Grad 3 **(d)** Punkte verlieren Grad 4

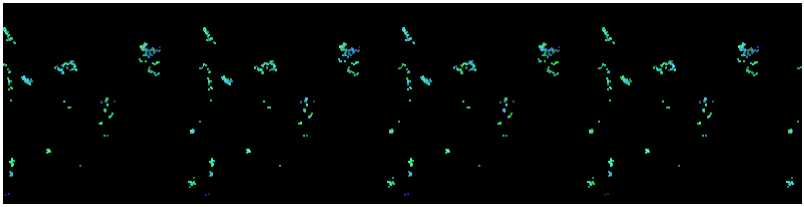


(e) Punkte hinzufügen Grad 1 **(f)** Punkte hinzufügen Grad 2 **(g)** Punkte hinzufügen Grad 3 **(h)** Punkte hinzufügen Grad 4

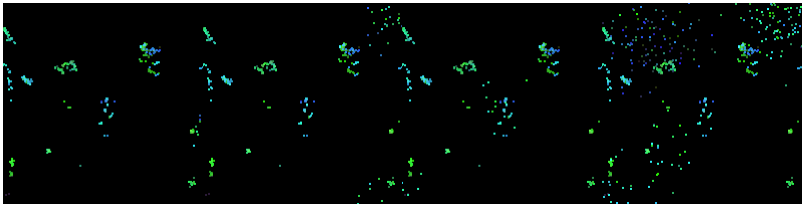
Abbildung O.4: Vergrößerung in einem Beispiel für die Störungen: ”Punkte verlieren” und ”Punkte hinzufügen” mit den Schwierigkeitsgraden: 1,2,3 und 4 auf RADAR Daten aus dem Astyx Datensatz. Die Punktwolken sind in BEV dargestellt und in der gleichen Weise wie bei Complex-YOLO sind die Farbkanäle kodiert.



(a) Punkte verschieben Grad 1 **(b)** Punkte verschieben Grad 2 **(c)** Punkte verschieben Grad 3 **(d)** Punkte verschieben Grad 4



(e) Information verrauscht Grad 1 **(f)** Information verrauscht Grad 2 **(g)** Information verrauscht Grad 3 **(h)** Information verrauscht Grad 4

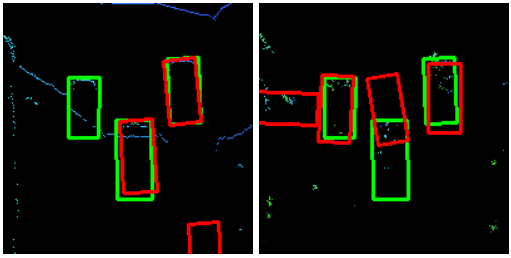


(i) Cluster Grad 1 **(j)** Cluster Grad 2 **(k)** Cluster Grad 3 **(l)** Cluster Grad 4

Abbildung O.5: Vergrößerung in einem Beispiel für die Störungen: ”Punkte verschieben”, ”Informationen verrauscht” und ”Cluster” mit den Schwierigkeitsgraden: 1,2,3 und 4 auf RADAR Daten aus dem Astyx Datensatz. Die Punktwolken sind in BEV dargestellt und in der gleichen Weise wie bei Complex-YOLO sind die Farbkanäle kodiert.

P Qualitative Ergebnisse der Punktwolkenanalyse

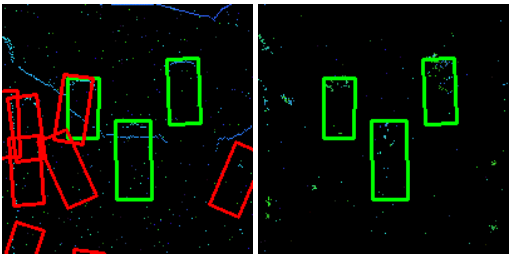
Die Abbildung P.1 zeigt die qualitativen Ergebnisse für Complex-YOLO LiDAR (Abb. P.1a) und Complex-YOLO RADAR (Abb. P.1b). Es zeigt sich, dass Complex-YOLO LiDAR weniger falsch-positive Ergebnisse liefert als Modelle, die mit RADAR-Daten trainiert wurden. Außerdem ist die Orientierung und Lokalisierung der detektierten Objekte mit LiDAR Daten genauer, während ein Objekt überhaupt nicht detektiert wird. Die Complex-YOLO Modelle für RADAR Daten schätzen zwar für alle drei Objekte ein Objekt, jedoch ist die Orientierung und Lokalisierung sehr ungenau und liefern jeweils zwei falsch positive Ergebnisse.



(a) Qualitatives Ergebnis Complex-YOLO LiDAR Originaldaten

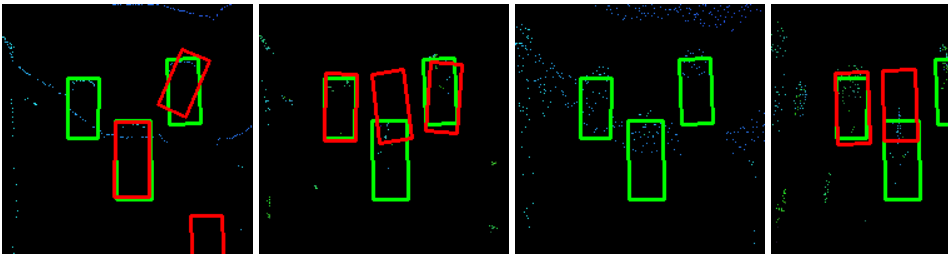
(b) Qualitatives Ergebnis Complex-YOLO RADAR Originaldaten

Abbildung P.1: Qualitative Ergebnisse für Complex-YOLO Modelle ohne Fusion auf Originaldaten ohne zusätzliche Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.



(a) Qualitatives Ergebnis Complex-YOLO LiDAR "Punkte hinzufügen Grad 2"

(b) Qualitatives Ergebnis Complex-YOLO RADAR "Punkte hinzufügen Grad 2"



(c) Qualitatives Ergebnis Complex-YOLO LiDAR "Punkte verlieren Grad 2"

(d) Qualitatives Ergebnis Complex-YOLO RADAR "Punkte verlieren Grad 2"

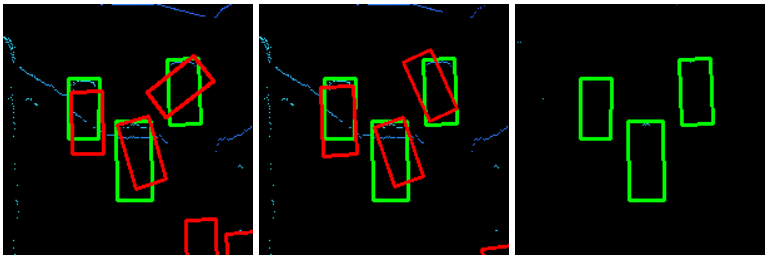
(e) Qualitatives Ergebnis Complex-YOLO LiDAR "Punkte verschieben Grad 2"

(f) Qualitatives Erge Complex-YOLO "Punkte verschie Grad 2"

Abbildung P.2: Qualitative Ergebnisse für Complex-YOLO Modelle ohne Fusion auf Daten mit zusätzlicher Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.

Die Abbildungen P.2 bis P.3 präsentieren die qualitativen Ergebnisse verschiedener Modelle, die auf gestörte Daten angewendet werden. Es wird deutlich, dass Störungen in den Daten zu einer Zunahme von falsch positiven sowie falsch negativen Ergebnissen führt. Dies tritt insbesondere bei den Störungen "Punkte hinzufügen Grad 2" und "Punkte verschieben Grad 2" hervor. Ein wesentlicher Faktor dabei ist, dass bei der Störung "Punkte verlieren Grad 2" viele der ursprünglichen Punkte erhalten bleiben. Diese Punkte tragen dazu bei, dass die Merkmale, welche der Detektor zur Identifizierung von Objekten nutzt – wie etwa die L-Form der Fahrzeuge in LiDAR-Daten –, größtenteils ungestört bleiben. Im Kontrast dazu führt das Hinzufügen neuer Punkte zu

einer Veränderung der Merkmale, während das Verschieben von Punkten dazu führen kann, dass die ursprünglichen Merkmale nicht mehr erkennbar sind.



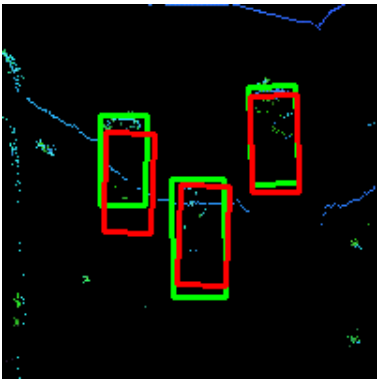
(a) Qualitatives Ergebnis Complex-YOLO LiDAR "Nebel Grad 2"

(b) Qualitatives Ergebnis Complex-YOLO LiDAR "Nebel Grad 3"

(c) Qualitatives Ergebnis Complex-YOLO LiDAR "Nebel Grad 4"

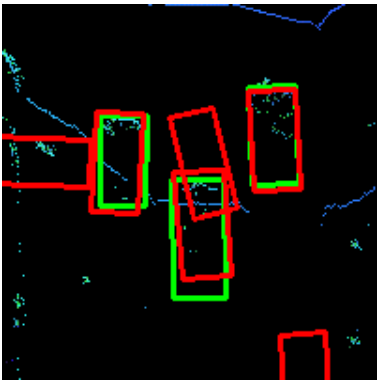
Abbildung P.3: Qualitative Ergebnisse für Complex-YOLO LiDAR auf Daten, welche durch Nebel ohne zusätzliche Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.

Auch die verschiedenen Schwierigkeitsgrade an Störung durch Nebel auf den LiDAR Daten führen zu einer Verschlechterung der Lokalisierung und Orientierung bei "Nebel Grad 2" und "Nebel Grad 3" (siehe Abb. P.3a und P.3b) und schließlich bei "Nebel Grad 4" dazu, dass überhaupt keine Objekte mehr erkannt werden konnten (Abb. P.3c).



(a) Qualitatives Ergebnis Complex-YOLO Low Level Originaldaten

Abbildung P.4: Qualitative Ergebnisse für Complex-YOLO Low Level auf Daten ohne zusätzlicher Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.

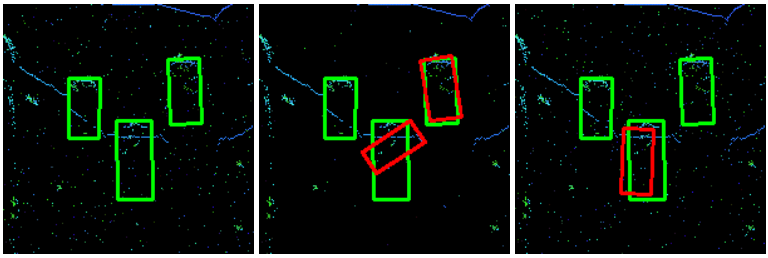


(a) Qualitatives Ergebnis Complex-YOLO High Level Originaldaten

Abbildung P.5: Qualitative Ergebnisse für Complex-YOLO Modelle mit High Level Fusion auf Daten ohne zusätzlicher Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.

Die Abbildungen P.4 und P.5 zeigen die qualitativen Ergebnisse für Complex-YOLO Modelle mit Low Level bzw. High Level Fusion auf Daten ohne zusätzliche Störungen. Sie verdeutlichen die verbesserte Detektionsleistung durch Low Level Sensorfusion, da Complex-YOLO Low Level alle drei Objekte detektieren konnten. Auch im qualitativen Beispiel in Abbildung P.5 werden alle drei Objekte durch die High Level Sensorfusion erkannt, allerdings ist die falsch positive Detektion bei Complex-YOLO High Level deutlich erhöht.

Bei Datenstörungen zeigen die Low Level-Fusion-Modelle in Abbildung P.6 dagegen ähnliche Schwächen wie die Modelle ohne Fusion zuvor. So sind in den qualitativen Ergebnissen bei der Verzerrung ”Punkte hinzufügen Grad 2” insbesondere fehlende Detektionen, also falsch negative, ein Problem. Im Vergleich dazu zeigen die High Level Fusion Modelle in Abbildung P.7 besonders viele falsch positive Ergebnisse, was damit zusammenhängt, dass die verwendeten Detektionsmodelle (Complex-YOLO LiDAR und Complex-YOLO RADAR Mag) mit Daten, die die Störung ”Punkte hinzufügen Grad 2” aufweisen, ebenfalls zu falsch positiven Ergebnissen führen (vgl. Abbildung P.7). Abbildung P.2). Das Modell Complex-YOLO RADAR hat dagegen keine falsch positiven Ergebnisse unter der Störung ”Punkte hinzufügen Grad 2”, aber auch keine richtig positiven Ergebnisse (vergl. P.2b), und somit hat Complex-YOLO High Level auch vergleichsweise weniger falsch positive Ergebnisse bei der Störung ”Punkte hinzufügen Grad 2” auf den RADAR-Daten.

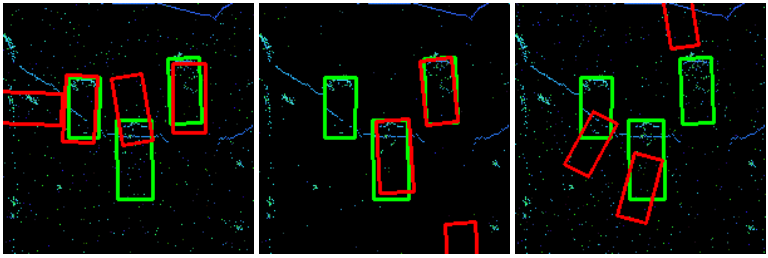


(a) Qualitatives Ergebnis Complex-YOLO Low Level ”Punkte hinzufügen Grad 2” auf den LiDAR Daten

(b) Qualitatives Ergebnis Complex-YOLO Low Level ”Punkte hinzufügen Grad 2” auf den RADAR Daten

(c) Qualitatives Ergebnis Complex-YOLO Low Level ”Punkte hinzufügen Grad 2” auf beiden Daten

Abbildung P.6: Qualitative Ergebnisse für Complex-YOLO Modelle mit Low Level Fusion auf Daten mit zusätzlicher Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.



(a) Qualitatives Ergebnis Complex-YOLO High Level ”Punkte hinzufügen Grad 2” auf den LiDAR Daten

(b) Qualitatives Ergebnis Complex-YOLO High Level ”Punkte hinzufügen Grad 2” auf den RADAR Daten

(c) Qualitatives Ergebnis Complex-YOLO High Level ”Punkte hinzufügen Grad 2” auf beiden Daten

Abbildung P.7: Qualitative Ergebnisse für Complex-YOLO Modelle mit High Level Fusion auf Daten mit zusätzlicher Störung. Dargestellt wird ein Ausschnitt der BEV-Projektion mit der wahre Bounding Box in grün und die geschätzte Bounding Box in rot.

Q Ergebnisse des Trainingskonzept bei einer Störung für Complex-YOLO

Dieser Abschnitt zeigt die Ergebnisse von Complex-YOLO Low Level, wenn es mit KISA und einer speziellen Störung trainiert. Welche Störung auf welchem Sensor einfließt, steht in der Bildunterschrift.

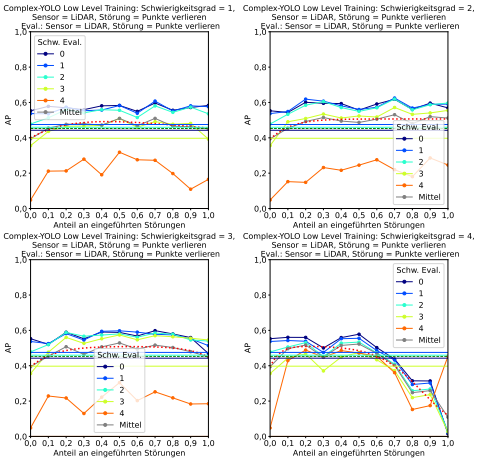


Abbildung Q.1: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte verlieren" mit verschiedenen Schwierigkeitsgraden auf den Daten vom LiDAR-Sensor.

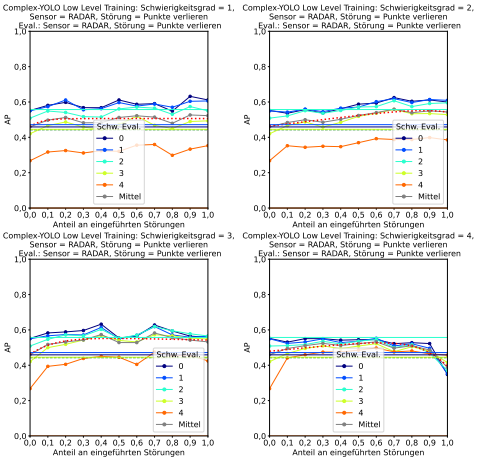


Abbildung Q.2: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte verlieren" mit verschiedenen Schwierigkeitsgraden auf den Daten vom RADAR-Sensor.

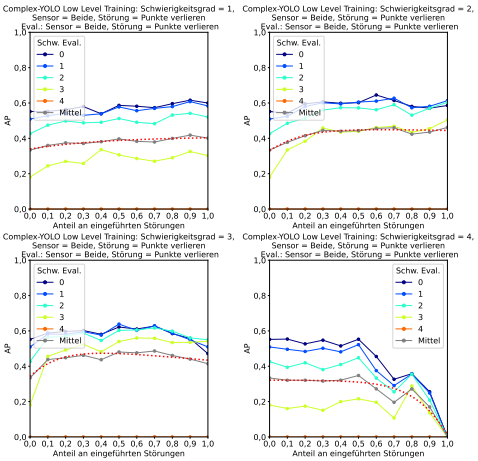


Abbildung Q.3: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte verlieren" mit verschiedenen Schwierigkeitsgraden auf den Daten beider Sensoren.

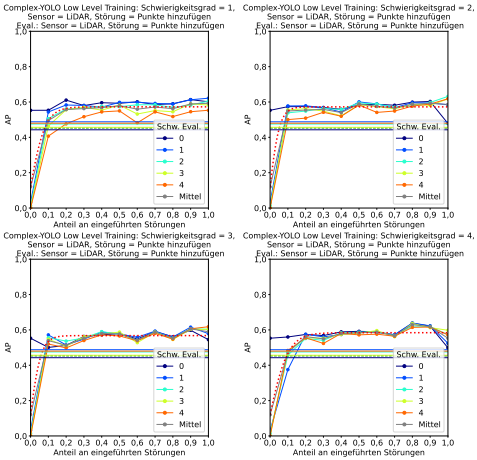


Abbildung Q.4: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte hinzufügen" mit verschiedenen Schwierigkeitsgraden auf den Daten vom LiDAR-Sensor.

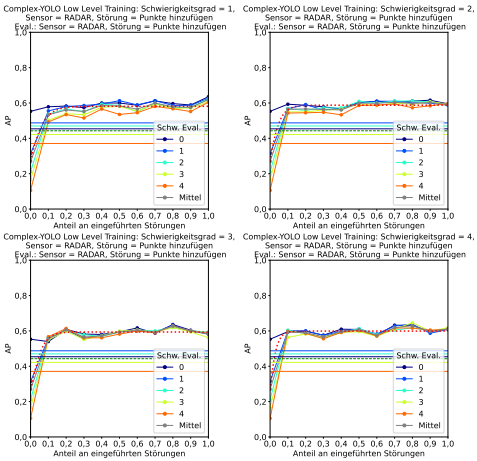


Abbildung Q.5: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte hinzufügen" mit verschiedenen Schwierigkeitsgraden auf den Daten vom RADAR-Sensor.

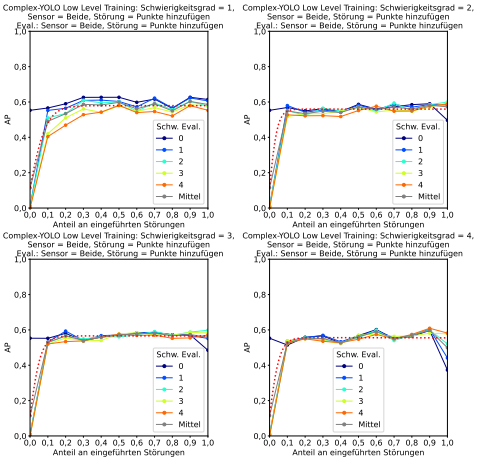


Abbildung Q.6: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte hinzufügen" mit verschiedenen Schwierigkeitsgraden auf den Daten beider Sensoren.

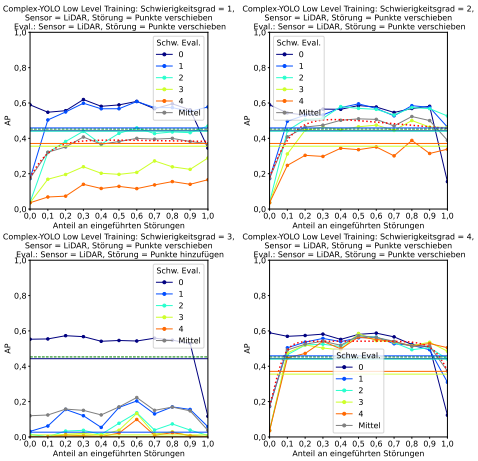


Abbildung Q.7: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte verschieben" mit verschiedenen Schwierigkeitsgraden auf den Daten vom LiDAR-Sensor.

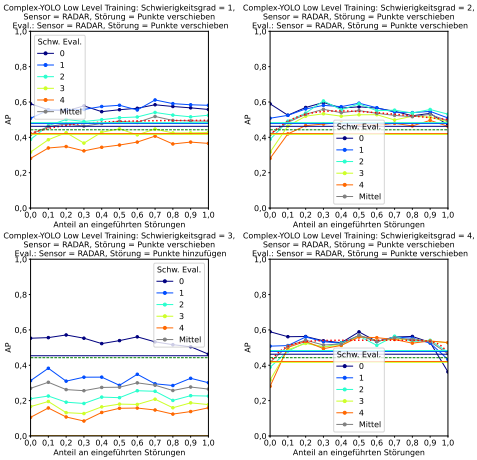


Abbildung Q.8: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte verschieben" mit verschiedenen Schwierigkeitsgraden auf den Daten vom RADAR-Sensor.

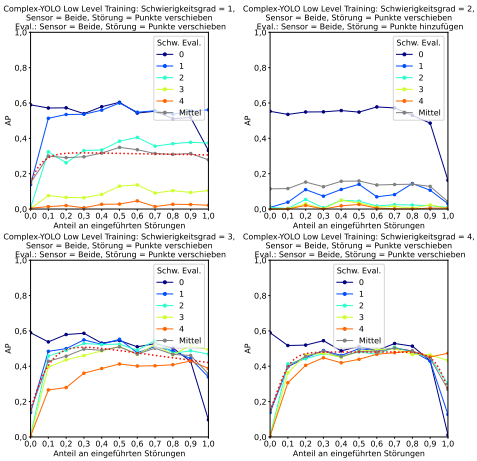


Abbildung Q.9: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Punkte verschieben" mit verschiedenen Schwierigkeitsgraden auf den Daten beider Sensoren.

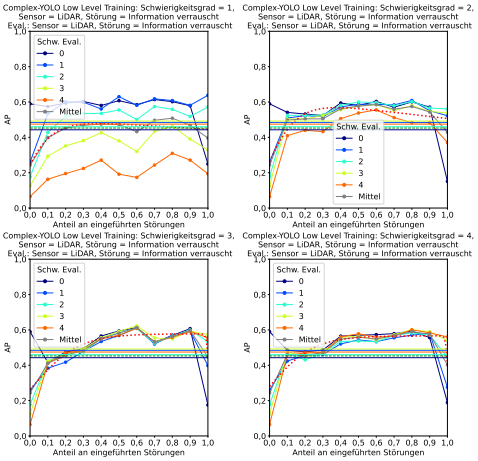


Abbildung Q.10: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Information verrauscht" mit verschiedenen Schwierigkeitsgraden auf den Daten vom LiDAR-Sensor.

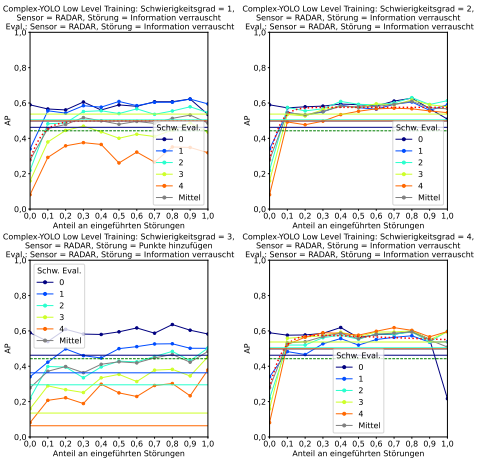


Abbildung Q.11: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Information verrauscht" mit verschiedenen Schwierigkeitsgraden auf den Daten vom RADAR-Sensor.

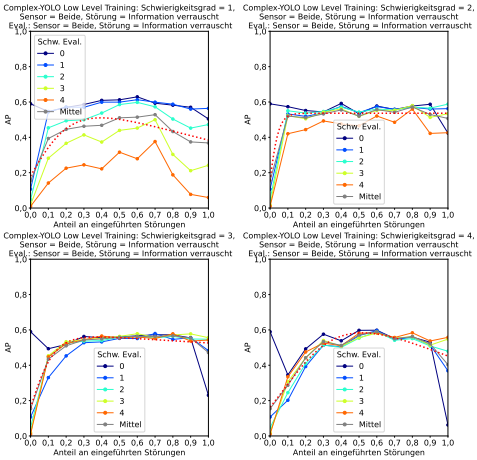


Abbildung Q.12: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Information verrauscht" mit verschiedenen Schwierigkeitsgraden auf den Daten beider Sensoren.

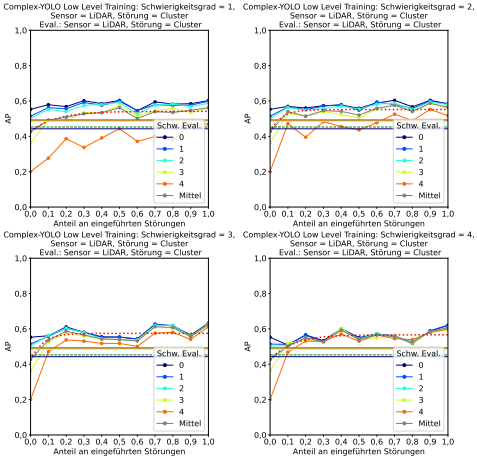


Abbildung Q.13: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Cluster" mit verschiedenen Schwierigkeitsgraden auf den Daten vom LiDAR-Sensor.

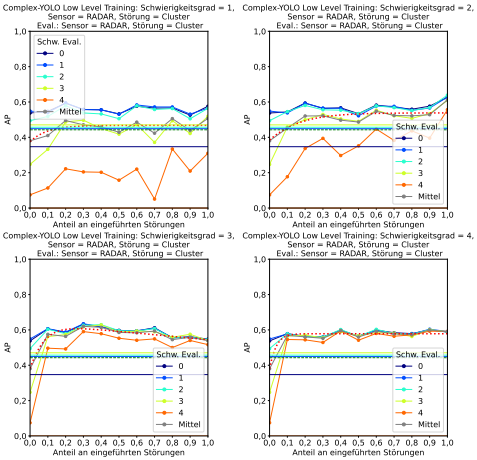


Abbildung Q.14: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Cluster" mit verschiedenen Schwierigkeitsgraden auf den Daten vom RADAR-Sensor.

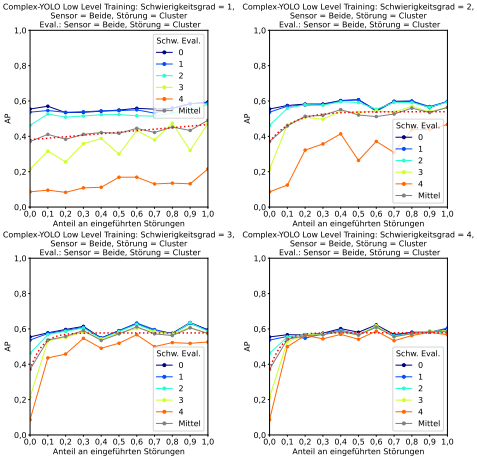


Abbildung Q.15: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Cluster" mit verschiedenen Schwierigkeitsgraden auf den Daten beider Sensoren.

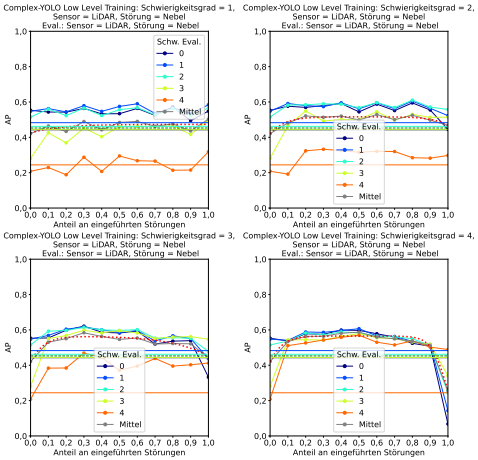


Abbildung Q.16: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Nebel" mit verschiedenen Schwierigkeitsgraden auf den Daten vom LiDAR-Sensor.

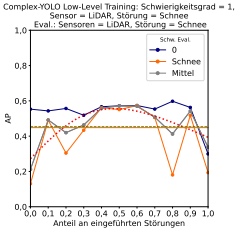
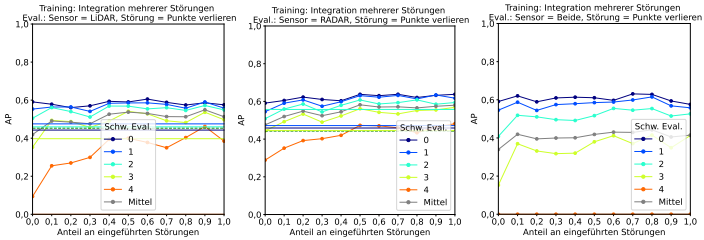


Abbildung Q.17: Performanz von Complex-YOLO Low Level trainiert mit KISA und der Störung "Schnee" auf den Daten vom LiDAR-Sensor.

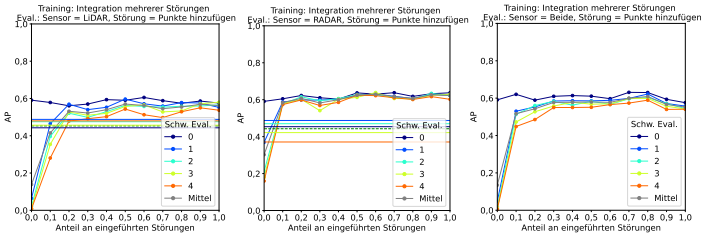
R Ergebnisse des Trainingskonzept bei mehreren Störungen für Complex-YOLO

Dieser Abschnitt zeigt die Ergebnisse von Complex-YOLO Low und High Level, wenn es mit KISA und mehreren Störungsarten und Störungsgraden trainiert wird. Auf welcher Störung evaluiert wird, steht in der Bildunterschrift.



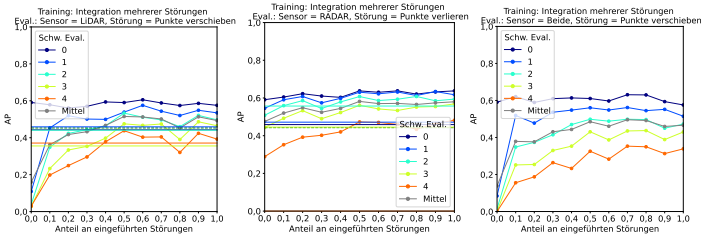
(a) "Punkte verlieren" auf den LiDAR Daten **(b)** "Punkte verlieren" auf den RADAR Daten **(c)** "Punkte verlieren" auf beiden Sensordaten

Abbildung R.1: Performanz von Complex-YOLO Low Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Punkte verlieren" auf den Daten des Astyx Datensatzes.



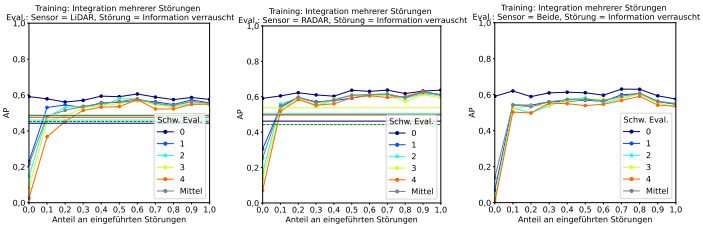
(a) "Punkte hinzufügen" auf den LiDAR Daten **(b)** "Punkte hinzufügen" auf den RADAR Daten **(c)** "Punkte hinzufügen" auf beiden Sensordaten

Abbildung R.2: Performanz von Complex-YOLO Low Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Punkte hinzufügen" auf den Daten des Astyx Datensatzes.



(a) "Punkte verschieben" auf den LiDAR Daten **(b)** "Punkte verschieben" auf den RADAR Daten **(c)** "Punkte verschieben" auf beiden Sensordaten

Abbildung R.3: Performanz von Complex-YOLO Low Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Punkte verschieben" auf den Daten des Astyx Datensatzes.

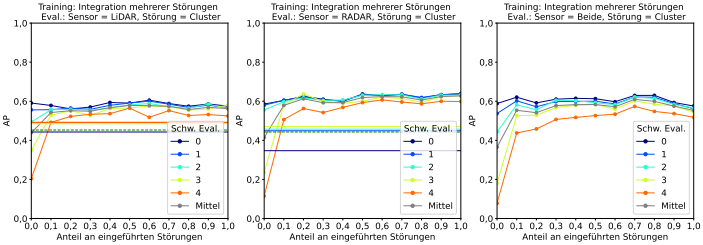


(a) "Information verrauscht" auf den LiDAR Daten

(b) "Information verrauscht" auf den RADAR Daten

(c) "Information verrauscht" auf beiden Sensordaten

Abbildung R.4: Performanz von Complex-YOLO Low Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Information verrauscht" auf den Daten des Astyx Datensatzes.

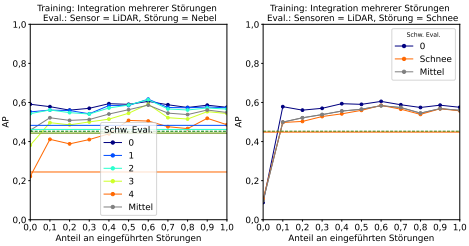


(a) "Cluster" auf den LiDAR Daten

(b) "Cluster" auf den RADAR Daten

(c) "Cluster" auf beiden Sensordaten

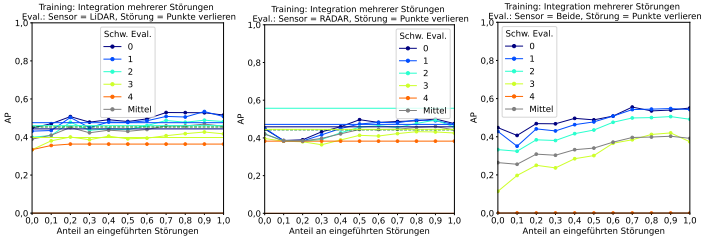
Abbildung R.5: Performanz von Complex-YOLO Low Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Cluster" auf den Daten des Astyx Datensatzes.



(a) "Nebel" auf den LiDAR Daten

(b) "Schnee" auf den LiDAR Daten

Abbildung R.6: Performanz von Complex-YOLO Low Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Nebel" bzw. "Schnee" auf den Daten des Astyx Datensatzes.

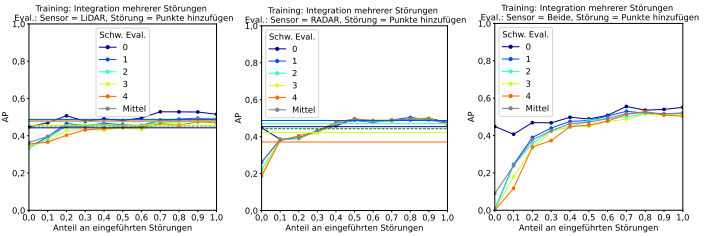


(a) "Punkte verlieren" auf den LiDAR Daten

(b) "Punkte verlieren" auf den RADAR Daten

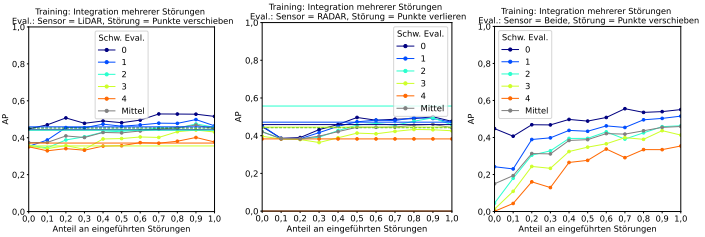
(c) "Punkte verlieren" auf beiden Sensordaten

Abbildung R.7: Performanz von Complex-YOLO High Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Punkte verlieren" auf den Daten des Astyx Datensatzes.



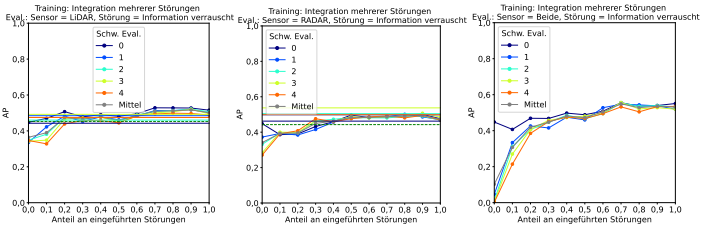
(a) "Punkte hinzufügen" auf den LiDAR Daten **(b)** "Punkte hinzufügen" auf den RADAR Daten **(c)** "Punkte hinzufügen" auf beiden Sensordaten

Abbildung R.8: Performanz von Complex-YOLO High Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Punkte hinzufügen" auf den Daten des Astyx Datensatzes.



(a) "Punkte verschieben" auf den LiDAR Daten **(b)** "Punkte verschieben" auf den RADAR Daten **(c)** "Punkte verschieben" auf beiden Sensordaten

Abbildung R.9: Performanz von Complex-YOLO High Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Punkte verschieben" auf den Daten des Astyx Datensatzes.

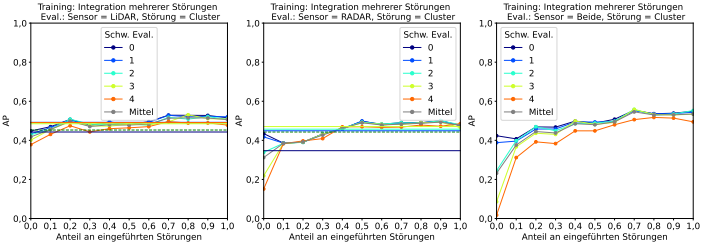


(a) "Information verrauscht" auf den LiDAR Daten

(b) "Information verrauscht" auf den RADAR Daten

(c) "Information verrauscht" auf beiden Sensordaten

Abbildung R.10: Performanz von Complex-YOLO High Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Information verrauscht" auf den Daten des Astyx Datensatzes.

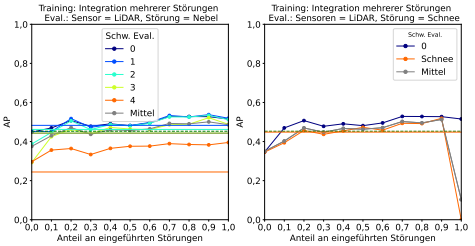


(a) "Cluster" auf den LiDAR Daten

(b) "Cluster" auf den RADAR Daten

(c) "Cluster" auf beiden Sensordaten

Abbildung R.11: Performanz von Complex-YOLO High Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Cluster" auf den Daten des Astyx Datensatzes.



(a) "Nebel" auf den LiDAR Daten

(b) "Schnee" auf den LiDAR Daten

Abbildung R.12: Performanz von Complex-YOLO High Level trainiert mit KISA mit allen Punktwolkestörungen und evaluiert bei "Nebel" bzw. "Schnee" auf den Daten des Astyx Datensatzes.