Echtzeitfähige kamerabasierte Objektdetektion für automatisierte Fahrzeuge

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

Hendrik Königshof, M.Sc.

Tag der mündlichen Prüfung: 13.10.2021

Hauptreferent: Prof. Dr.-Ing. Christoph Stiller
Korreferent: Prof. Dr.-Ing. Klaus Dietmayer

Danksagung

Die vorliegende Arbeit resultiert aus meiner Tätigkeit als wissenschaftlicher Mitarbeiter am FZI Forschungszentrum Informatik und als Doktorand am Karlsruher Institut für Technologie (KIT). Sie wäre ohne die vielfältige Unterstützung meiner Kollegen, Familie und Freunde nicht möglich gewesen.

Mein erster Dank gilt Herrn Prof. Dr.-Ing. Christoph Stiller für die Betreuung dieser Arbeit, die konstruktiven Diskussionen und die Schaffung der ausgezeichneten Rahmenbedingungen für das wissenschaftliche Arbeiten. Weiterhin möchte ich mich bei Herrn Prof. Dr.-Ing. Klaus Dietmayer für die Übernahme des Korreferats und das damit verbundene Interesse an meiner Arbeit bedanken.

Ich danke allen Kolleginnen und Kollegen für die äußerst angenehme Arbeitsatmosphäre, die vielen Diskussionen während der Kaffeerunden, Dienstreisen und Sommerseminare, aber auch für die gemeinsamen Skiausflüge, Sport- und Pubquizabende und Social Tuesdays. Für das Korrekturlesen dieser Arbeit bedanke ich mich bei Ole Salscheider, Frank Holländer und vor allem Maximilian Naumann, der mich nicht nur durch das Korrekturlesen sondern auch durch viele hilfreiche Hinweise während der Erstellung dieser Arbeit außerordentlich unterstützt hat.

Abschließend möchte ich mich bei meinen Eltern und meiner Schwester bedanken, die den Grundstein zu dem legten, was ich heute bin und auf deren Unterstützung ich immer zählen kann.

Karlsruhe im März 2021, *Hendrik Königshof*

Kurzfassung

Automatisierte Fahrzeuge müssen ihre Umgebung schnell und präzise erfassen um in der Lage zu sein, mit ihr zu interagieren. Sie müssen dabei in einer Verkehrsumgebung zurecht kommen, welche speziell so gestaltet ist, dass sie für die visuelle Wahrnehmung des Menschens leicht zugänglich ist. Aus diesem Grund sind bildgebende Verfahren mit ähnlicher Verarbeitungsgeschwindigkeit, Zuverlässigkeit und Robustheit wie das menschliche Sehsystem, unabdingbar.

Diese Arbeit befasst sich deshalb mit der echtzeitfähigen kamerabasierten 3D-Objekterkennung. Im Gegensatz zu existierenden Arbeiten liegt der Fokus dabei nicht nur auf Fahrzeugen, sondern auf allen Arten von Verkehrsteilnehmern, wie zum Beispiel Fußgängern und Radfahrern. Das Ziel dieser Arbeit ist es, einen Ansatz zur Objektdetektion zu entwickeln, welcher mit möglichst geringem Rechenaufwand Ergebnisse liefert, die in ihrer Genauigkeit und Zuverlässigkeit für die in automatisierten Fahrzeugen auf die Umfelderfassung folgenden Module wie die Prädiktion, Verhaltensgenerierung und Trajektorienplanung ausreichend sind.

Dafür werden zwei binokulare Methoden vorgestellt, welche Rasterkarten in der Vogelperspektive als abstrahierte Darstellung des Fahrzeugumfelds verwenden. Durch die damit verbundene direkte räumliche Information und die Dimensionsreduktion gegenüber Punktwolken verringert sich die Komplexität und somit die Laufzeit der Objektdetektion. Da die Verwendung eines binokularen Kamerasystems nicht nur höhere Kosten als eine einzige Kamera verursacht, sondern vor allem auch eine aufwendigere Kalibrierung benötigt, wird in dieser Arbeit ebenfalls untersucht, welche Genauigkeiten mit einer monokularen 3D-Objektdetektion erreicht werden können.

Untersuchungen auf dem öffentlich verfügbaren KITTI-Datensatz [GLU12] zeigen, dass die vorgestellten Ansätze den Stand der Technik für echtzeitfähige kamerabasierte 3D-Objekterkennung übertreffen. Darüber hinaus zeigen

Experimente auf dem Testfahrzeug BerthaOne gute Objektdetektionen hinsichtlich Position und Abmessung auch für Objekte in 100 m Entfernung.

Abstract

Autonomous vehicles need to sense their environment quickly and accurately in order to be able to interact with it. They must be able to understand and effortlessly act within a traffic environment that has been specifically designed to be easily accessible for humans. For this reason, imaging techniques with similar processing speed, reliability and robustness as the human visual system, are essential.

Therefore, this work investigates real-time camera-based 3D object recognition. In contrast to existing approaches, it focuses not only on vehicles, but on all types of road users, including pedestrians and cyclists. The goal of this work is to develop an approach for object detection that provides results which are sufficient in their accuracy and reliability for high level scene understanding and trajectory planning, with a minimum of computational effort.

For this purpose, two binocular methods are presented that use grid maps in bird's-eye view as an abstracted representation of the vehicle environment. The direct spatial information and dimensionality reduction compared to point clouds reduces the complexity and thus the runtime of the object detection. Since stereo camera systems not only cause higher costs than a single camera, but also require a more complex calibration, this thesis also investigates which accuracies can be achieved with a monocular 3D object detection.

Evaluations on the publicly available KITTI dataset [GLU12] show that the presented approaches outperform the state of the art for real-time camera-based 3D object detection. Furthermore, experiments on the test vehicle BerthaOne demonstrate reasonable object detections regarding object position and dimensions even for objects at distances of 100 m.

Inhaltsverzeichnis

Abkürzungen und Symbole

Abkürzungen

2D/3D	**2/3-D**imensional
AOS	**A**verage **O**rientation **S**imilarity
BEV	**B**ird's **E**ye **V**iew
CCL	**C**onnected **C**omponent **L**abeling
CNN	**C**onvolutional **N**eural **N**etwork
IoU	**I**ntersection **o**ver **U**nion
KNN	**K**ünstliches **N**euronales **N**etz
LiDAR	**Li**ght **D**etection **A**nd **R**anging
LRD	**L**eft **R**ight **D**ifference
MLP	**M**ehr**L**agige **P**erzeptronen
Radar	**Ra**dio **d**etection **a**nd **r**anging
ReLU	**Re**ctified **L**inear **U**nit
RoI	**R**egion **o**f **I**nterest

Allgemeine Notationen

x, y, z, X, Y, Z	Skalare
$\mathbf{x}, \mathbf{y}, \mathbf{z}$	Vektoren
$\mathbf{X}, \mathbf{Y}, \mathbf{Z}$	Matrizen
$\hat{x}, \hat{\mathbf{y}}, \hat{\mathbf{Z}}$	Unsicherheitsbehaftete Größen

Symbole

$\mathbf{O}$	Ursprung des weltfesten Koordinatensystems
b	Basisbreite
$B_{i,j}(x)$	Basis-Spline
$\mathcal{B}$	Rasterkartenbox
$(c_\mathrm{u}, c_\mathrm{v})$	Bildhauptpunkt
C	Konfidenz
$\mathbf{C}$	Optisches Zentrum einer Kamera
d	Disparität
f	Brennweite
h_C	Einbauhöhe der Kamera
$\mathcal{H}_1()$	Huber-Verlustfunktion
$\mathbf{J}$	Jakobi-Matrix
L	Verlustfunktion
$p_\mathrm{o}(x, y)$	Belegtheit einer Rasterzelle
$\mathbf{P}$	Projektionsmatrix
r	Rasterkartenauflösung

$\mathbf{R}$	Rotationsmatrix
s	Rasterkartengröße
$\mathbf{t}$	Translationsvektor
u	Bildspalte
v	Bildzeile
θ_{C}	Kamera-Nickwinkel
θ	Globale Orientierung
θ_l	Lokale Orientierung
θ_{ray}	Sichtstrahlrichtung
μ	Mittelwert
σ	Standardabweichung
Φ	Rollwinkel

1 Einleitung

Auf dem Gebiet der Fahrerassistenzsysteme hat es in den letzten Jahren enorme Fortschritte gegeben. Dazu gehören beispielsweise Spurhalteassistenten, Notbremssysteme, Totwinkel-Überwachung und viele weitere. Diese Systeme führen neben dem Komfortgewinn auch zu einer erhöhten Sicherheit im Straßenverkehr. Nichtsdestotrotz sind nach Angaben des Statistischen Bundesamtes [Sta17] weiterhin 88 % der Verkehrsunfälle in Deutschland auf ein Fehlverhalten des Fahrers zurückzuführen. Der nächste Schritt werden deshalb vollständig automatisierte Fahrzeuge sein, die nicht nur die Zahl der Unfälle reduzieren, sondern auch Staus verhindern und die Schadstoffbelastung verringern werden.

Wie wir bereits in der Fahrschule lernen, ist die korrekte Überwachung der Umgebung des eigenen Fahrzeugs unerlässlich um sicher durch dichten Verkehr navigieren zu können und Kollisionen mit anderen Verkehrsteilnehmern zu vermeiden. Natürlich ist diese Umfelderfassung auch für moderne Fahrerassistenzsysteme oder eben automatisierte Fahrzeuge essentiell, da die Entscheidung über eine Vollbremsung, ein Ausweichmanöver oder einen Spurwechsel oder die Planung der zukünftigen Trajektorie des Fahrzeugs, ohne Kenntnis der Umgebung nicht realisiert werden können.

Eines der wichtigsten Gebiete ist dabei die Erkennung anderer beweglicher Verkehrsteilnehmer. Neben Ultraschallsensorik, welche ausschließlich für den Nahbereich des Fahrzeuges, zum Beispiel in Parksensoren zum Einsatz kommt, liegt der Fokus aktueller Systeme zur Objektdetektion stark auf Radarsensoren. Mithilfe von Radar können die Position und vor allem Geschwindigkeit anderer Fahrzeuge zuverlässig bestimmt werden. Auch ist es sehr robust gegen ungünstige Witterungs- und Lichtverhältnisse, weshalb es für sicherheitskritische Fahrerassistenzsysteme wie Abstandsregeltempomat und Spurwechselassistent Verwendung findet.

Aufgrund der genauen Tiefeninformation basieren die meisten Objekterkennungsmethoden in automatisierten Forschungsfahrzeugen, wie dem in Abbildung 1.1 dargestellten Fahrzeug BerthaOne, stark auf LiDAR-Systemen. Der Nachteil dieser Systeme sind die hohen Kosten und kürzere Wahrnehmungsreichweiten. Eine kostengünstige Alternative bietet das ebenfalls dargestellte binokulare Kamerasystem, welches aus zwei Kameras mit überlappendem Sichtbereich besteht und durch Triangulation der Punktkorrespondenzen zwischen den Kameras ebenfalls eine Tiefenrekonstruktion der Umgebung ermöglicht. Diese Abstandsinformation wird als Disparitätsbild bezeichnet und kann unter anderem zur Detektion von Objekten genutzt werden. Da automatisierte Fahrzeuge in einer Verkehrsumgebung zurecht kommen müssen, die speziell so gestaltet ist, dass sie für die visuelle Wahrnehmung des Menschen leicht zugänglich ist, sind bildgebende Verfahren unabdingbar. Dies ist einer der Gründe, warum Kameras in den meisten modernen Fahrzeugen bereits vorhanden sind. Monokulare Kameras finden dabei hauptsächliche Verwendung bei der Erkennung von Verkehrszeichen, Ampeln und Fahrbahnmarkierungen. Binokulare bzw. Stereokamerasysteme werden aktuell hauptsächlich für Notbremssysteme eingesetzt.

Um vollständig automatisiertes Fahren zu ermöglichen, sind Objekterkennungsalgorithmen mit ähnlicher Geschwindigkeit, Zuverlässigkeit und Robustheit wie das menschliche Sehsystem eine Grundvoraussetzung. Diese Arbeit befasst sich deshalb mit der echtzeitfähigen kamerabasierten 3D-Objektdetektion. Da gerade die Verkehrsteilnehmer, welche am verletzlichsten sind, am schwierigsten zu detektieren sind, liegt der Fokus in dieser Arbeit nicht nur auf der Erkennung von Fahrzeugen, sondern auch auf der Detektion von Fußgängern und Fahrradfahrern. Die Bestimmung des Objekttyps trägt wesentlich dazu bei sowohl die Dimension als auch die zukünftige Bewegung eines Objektes besser einschätzen zu können.

1.1 Ziele der Arbeit

Die Ziele dieser Arbeit liegen vor allem darin einen möglichst schnellen Ansatz zur kamerabasierten 3D-Objektdetektion zu entwickeln, welcher Ergebnisse liefert, die in ihrer Genauigkeit für die in automatisierten Fahrzeugen auf die Umfelderfassung folgenden Module wie die Prädiktion, Verhaltensgenerie-

Abbildung 1.1: Sensorik des Forschungsfahrzeugs BerthaOne. Für die linke Kamera des binokularen Kamerasystems steht eine Graustufen- oder eine Farbkamera zur Verfügung.

rung und Trajektorienplanung ausreichend sind. Um dies zu erreichen werden zwei binokulare Methoden vorgestellt, welche Rasterkarten als abstrahierte Darstellung des Fahrzeugumfelds verwenden. Durch die damit verbundene Dimensionsreduktion verringert sich nicht nur die Komplexität und somit die Laufzeit der Objektdetektion, auch wird der Fokus auf die für die Anwendung im automatisierten Fahrzeug wichtigere Positionsgenauigkeit entlang der Bodenoberfläche gelegt.

Da die Verwendung eines binokularen Kamerasystems nicht nur höhere Kosten als eine einzige Kamera verursacht, sondern vor allem auch eine aufwendigere Kalibrierung benötigt, soll in dieser Arbeit ebenfalls untersucht werden, welche Genauigkeiten mit einer monokularen 3D-Objektdetektion erreicht werden können.

1.2 Beiträge der Arbeit

Die Beiträge dieser Arbeit ergeben sich zusammengefasst zur:

- Vorstellung eines Ansatzes zur Schätzung der Bodenoberfläche auf Basis von V-Disparitäten und deren Modellierung mithilfe von B-Splines.
- Fusion von semantischer und Tiefeninformation zur Bestimmung von Objektclustern im Bild.
- Projektion dieser Objektcluster in eine Rasterkartendarstellung und Aufstellen einer Optimierungsfunktion zur Bestimmung der Position, Orientierung und Dimension des jeweiligen Objektes, wodurch 3D Bounding Boxen für alle im Bild befindlichen Objekte in Echtzeit bestimmt werden können.
- Erweiterung dieses Ansatzes durch die Schätzung der Objektparameter mithilfe eines Convolutional Neural Networks (CNN) innerhalb von Rasterkartenausschnitten, wodurch nicht nur nochmal verbesserte Ergebnisse erzielt werden, sondern auch die Gesamtlaufzeit der Objektdetektion auf etwa 60 ms verringert wird. Auf dem öffentlich verfügbaren KITTI-Datensatz [GLU12] ist dieser Ansatz damit der beste echtzeitfähige und schnellste binokulare Objektdetektor mit Genauigkeiten im Bereich von anderen Arbeiten, welche eine bis zu sechsfache Rechenzeit benötigen.
- Entwicklung eines CNNs, welches die 3D-Information von Objekten direkt aus einem einzigen Kamerabild schätzt und auf dem KITTI-Datensatz den besten echtzeitfähigen monokularen Ansatz darstellt.

1.3 Struktur der Arbeit

Nach einer Übersicht zu verwandten Arbeiten im Bereich der kamerabasierten 3D-Objektdetektion in Kapitel 2, werden in Kapitel 3 einige fachliche Grundlagen dargelegt, welche für das Verständnis der weiteren Kapitel hilfreich sind. Dabei wird das Modell einer idealen Lochkamera und der Prozess der Tiefenrekonstruktion mithilfe binokularer Kamerasysteme beschrieben. Neben einer

kurzen Herleitung von B-Splines wird auf die Methode der kleinsten Fehlerquadrate für lineare und nichtlineare Modellfunktionen eingegangen. Das Kapitel schließt mit einer Einführung zu Künstlichen Neuronalen Netzen (KNN).

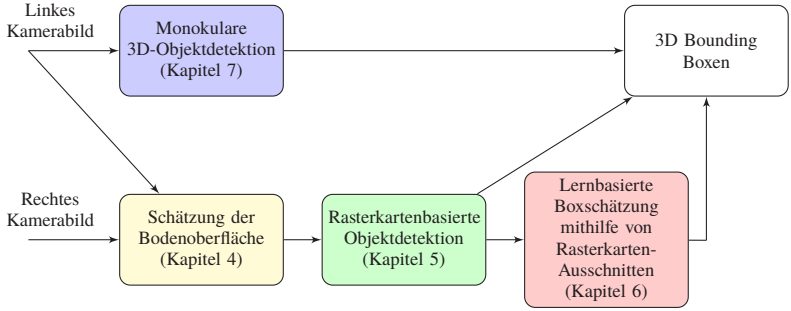


Abbildung 1.2: Übersicht über den Zusammenhang der in dieser Arbeit enthaltenen Kapitel.

Kapitel 4 erläutert die Schätzung des in der Kamera sichtbaren Fahrbahnverlaufs auf Basis sogenannter V-Disparitäten und deren Modellierung mithilfe von B-Splines. Wie in Abbildung 1.2 dargestellt bildet dies die Grundlage für die in den beiden darauf folgenden Kapiteln genutzte Rasterkartendarstellung. Durch die Kenntnis des Verlaufs der Bodenoberfläche können zum Boden gehörende Punkte bei der Erstellung dieser Karten herausgefiltert und auch die Höhe und Position detektierter Objekte genauer abgebildet werden, da sich diese auf der Bodenoberfläche befinden müssen.

Kapitel 5 beschreibt, wie die mithilfe von Tiefenschätzung und semantischer Information im Bild bestimmten Objektcluster in eine Rasterkartendarstellung überführt und daraus die Orientierung und Dimension des jeweiligen Objekts unter Verwendung klassenspezifischer Objektgrößen optimiert werden.

In Kapitel 6 wird dieser Ansatz erweitert und ein Ausschnitt der erstellten Rasterkarten pro Objekt verwendet um mit einem CNN direkt dessen 3D Bounding Box Parameter zu schätzen.

Kapitel 7 erläutert, wie die 3D Bounding Box direkt aus einem einzigen Kamerabild geschätzt werden kann. Dafür werden ebenfalls semantische und Tiefeninformation verwendet, für jede 2D-Box konkateniert und mithilfe eines CNNs die Position, Größe und Orientierung des Objektes bestimmt.

Die in den vorherigen drei Kapiteln beschriebenen Ansätze werden in Kapitel 8 anhand des KITTI Benchmark-Datensatzes [GLU12] quantitativ evaluiert und mit den Ergebnissen verwandter Arbeiten verglichen. Außerdem werden Ergebnisse des bereits erwähnten Forschungsfahrzeugs BerthaOne präsentiert.

In Kapitel 9 werden die wesentlichen Beiträge dieser Arbeit zusammengefasst und ein Ausblick über mögliche fortführende Forschungsarbeiten gegeben.

2 Stand der Technik

Dieses Kapitel gibt eine Übersicht zu aktuellen Arbeiten, die sich mit kamerabasierter 3D-Objektdetektion beschäftigen. Zunächst werden in Abschnitt 2.1 Objektrepräsentationen aufgezeigt, welche neben der klassischen Bounding Box in aktuellen monokularen und binokularen Objektdetektoren zum Einsatz kommen. Abschnitt 2.2 geht auf aktuelle Ansätze zur monokularen Objektdetektion ein und teilt diese je nach verwendeten Eingangsdaten in fünf Kategorien ein. In Abschnitt 2.3 werden Methoden zur Objekterkennung vorgestellt, welche auf binokularen Kamerasystemen beruhen.

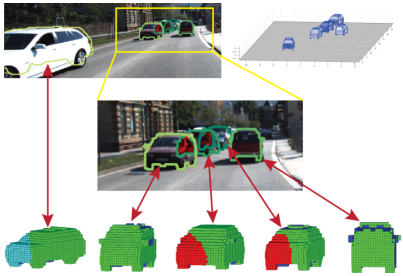
2.1 Objektrepräsentationen

Seit den frühen Tagen der Bildverarbeitung wurden verschiedenste Modelle entwickelt um die Objekterkennung aus Bildern zu unterstützen [Sze11]. Hier soll nur kurz auf vier Repräsentationen eingegangen werden, welche neben der klassischen 2D und 3D Bounding Box in aktuellen Ansätzen Verwendung finden und in Abbildung 2.1 visualisiert sind.

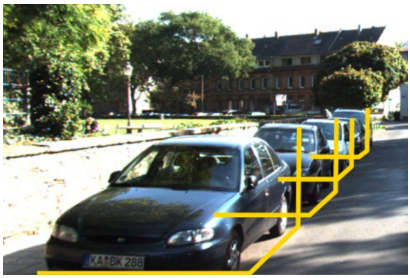
Zia et al. [ZSSS13] schlagen Drahtmodelle (engl. wire frames) vor um die Form verschiedenster Objekte besser darzustellen und deren Pose so genauer bestimmen zu können. Zur Erzeugung dieser verformbaren Drahtmodelle verwendet man die Eckpunkte dreidimensionaler CAD-Modelle und führt eine Hauptkomponentenanalyse der 3D-Koordinaten durch. Die endgültige Darstellung basiert dann auf dem mittleren Drahtmodell zuzüglich der Hauptkomponentenrichtungen und deren Standardabweichungen.

Um ebenfalls eine genauere Repräsentation von Objekten zu erhalten, nutzen Xiang et al. [XWLS15] 3D-Voxel-Modelle, welche die Schlüsseleigenschaften von Objekten wie Erscheinung, 3D-Form, Blickwinkel, Verdeckung und Trunkierung kodieren können.

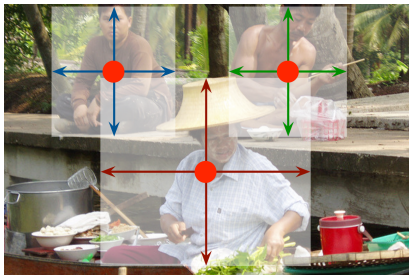
(a) Drahtmodell [ZSSS13]



(b) 3D-Voxel-Modell [XWLS15]



(c) Bounding Shapes [GWW+19]



(d) Objekt als Punkt [ZWK19]

Abbildung 2.1: Visualisierung aller in diesem Abschnitt vorgestellten Objektrepräsentationen.

BS^{3}D [GWW+19] schlägt als Abwandlung von klassischen 2D-Boxen sogenannte Bounding Shapes vor. Dies sind geometrische Objekte mit Manhattan-Struktur, welche vier im Bild garantiert sichtbare Merkmalspunkte enthalten. Im Gegensatz zu 2D Bounding Boxen kodiert die Bounding-Shape-Darstellung wichtige Informationen des Objekts wie die Orientierung. Sie kann sehr effizient in aktuelle 2D-Objektdetektoren wie YOLO [RDGF16], SSD [LAE+16] oder Faster R-CNN [RHGS17] integriert werden und durch Annahme einer flachen Bodenebene oder zusätzlicher Verwendung einer Tiefenschätzung kann daraus direkt die 3D-Box berechnet werden.

Zhou et al. [ZWK19] reduzieren die Anzahl der Merkmalspunkte noch weiter und repräsentieren Objekte als einen einzigen Punkt und zwar den Mittelpunkt der Bounding Box. Größe, Position, Orientierung und weitere Objekteigenschaften können dann von dessen Merkmalsvektor abgeleitet werden.

2.2 Monokulare Objektdetektion

Aktuelle Ansätze zur monokularen Objektdetektion lassen sich nach den hauptsächlich verwendeten Eingangsdaten in fünf Kategorien einteilen. Zuerst werden in Abschnitt 2.2.1 Methoden vorgestellt, welche auf Tiefeninformationen beruhen und diese größtenteils in eine Punktwolkenstruktur bringen. Aber auch in den darauf folgenden drei Kategorien finden sich einzelne Ansätze, welche zusätzlich eine Tiefenschätzung verwenden. Dabei werden in Abschnitt 2.2.2 Ansätze erläutert, welche anhand von Konturinformationen, wie Draht- oder CAD-Modellen eine Posen- und Formschätzung durchführen, die den Projektionsfehler zwischen Modell und Bild minimieren. Anschließend wird in Abschnitt 2.2.3 auf Arbeiten eingegangen, welche anhand von semantischer und Instanz-Information eine Optimierung der 3D-Boxschätzung durchführen. Weitere in Abschnitt 2.2.4 vorgestellte Ansätze bestimmen die 3D-Boxparameter in der Vogelperspektive, welche entweder durch eine Tiefenschätzung oder durch eine direkte orthografische Merkmalstransformation erzeugt wird. Dadurch können viele unerwünschte Eigenschaften von Bildern vermieden werden, welche es schwierig machen auf die 3D-Geometrie der Umgebung zu schließen. Ein Großteil aktueller Methoden versucht ohne die Verwendung von 3D-Modellen, Semantik oder Tiefe, die 3D-Box nur anhand des geometrischen Zusammenhangs mit der 2D-Box zu schätzen. Auf diese Arbeiten wird in Abschnitt 2.2.5 näher eingegangen.

2.2.1 Ansätze basierend auf Tiefeninformationen

Einige Ansätze lösen das Problem der Objektdetektion direkt im 3D-Raum, wodurch existierende LiDAR-basierte Detektionsalgorithmen verwendet werden können. [WK19] konvertiert dafür die monokulare Tiefenschätzung von DORN [FGW+18] in eine Pseudo-LiDAR genannte Punktwolkenrepräsentation um mithilfe des Frustum PointNets [QLW+18] 3D-Boxen zu bestimmen. Außerdem werden Instanzmasken anstatt 2D-Boxen als Region of Interest (RoI) verwendet um robuster gegen Rauschen in der Punktwolke zu sein und ein „2D-3D Bounding-Box Consistency Constraint“ vorgeschlagen, welches die 3D Bounding Box so anpasst, dass sie nach der Projektion auf das Bild eine hohe Überlappung mit dem entsprechenden 2D-Vorschlag aufweist. RefinedMPL [VAL19] verbessert die Qualität von Pseudo-LiDAR-

Punktwolken durch überwachte und unüberwachte Ausdünnung und verwendet PointRCNN [SWL19] als 3D-Detektor. AM3D [MWL+19] berechnet eine Punktwolke für jede RoI und unterstützt die 3D-Detektion mit zusätzlichen Merkmalen aus dem RGB-Bild. MonoFENet [BXC20] verfolgt einen ähnlichen Ansatz, dünnt die Punktwolken allerdings noch zusätzlich durch Mean Pooling aus. Xu und Chen [XC18] schätzen die 3D-Position des Objektes ebenfalls direkt aus einer Punktwolke, generieren aber aus der Tiefenschätzung zusätzliche Merkmalskarten welche mit dem Eingangsbild verknüpft werden um die 2D-Box Schätzung zu verbessern.

D^4LCN [DHY+20] nutzt die Tiefenschätzung um die Filterkernel des lokalen Faltungsnetzes und ihre Dilatation dynamisch anpassen zu können. ROI-10D [MKG19] erzeugt mithilfe monokularer Tiefe eine vollständig differenzierbare Zuordnung von 2D-Detektionen und Orientierungsschätzung in den 3D-Raum, welche als RoI-Lifting bezeichnet wird.

2.2.2 Ansätze basierend auf Konturinformationen

Zia et al. [ZSS14] nutzen die Bodenebene als gemeinsame Szenengeometrie mehrerer Objekte, für welche dann die 3D-Form und Position gleichzeitig geschätzt werden. Dabei wird ein Drahtmodell verwendet um die 3D-Geometrie und Abhängigkeiten zwischen den einzelnen Objekten wie Verdeckungen detailliert beschreiben zu können. [MKCK17] verwendet ebenfalls ein Drahtmodell und optimiert den Abstand zu Merkmalspunkten auf den Objekten. Die Posen- und Formschätzung findet allerdings getrennt voneinander statt, da bei einer gemeinsamen Schätzung eine Minimierung des Projektionsfehlers zu willkürlichen Verformungen der 3D-Gestalt führen kann. Deep MANTA [CCR+17] nutzt einen CAD-Fahrzeugmodelldatensatz um Drahtmodelle zu erstellen. Dabei wird eine iterative Verfeinerung von Objektvorschlägen im Eingangsbild mithilfe des Deep MANTA-Netzwerks durchgeführt und das ähnlichste 3D-Modell aus dem Datensatz gesucht. Durch einen Abgleich können die Fahrzeugausrichtung und 3D-Position rekonstruiert und sogar verdeckte Fahrzeugteile erkannt werden. Mono3D++ [HS19] verwendet zusätzlich eine monokulare Tiefenschätzung und führt eine gemeinsame Optimierung des Projektionsfehlers der 3D Bounding Box, der Tiefe, der Bodenebene und der Abweichung von einem anpassbaren Drahtmodell durch. 3D-RCNN [KLR18] nutzt die Faster-RCNN Architektur und fügt dieser zwei Ausgänge zur Prä-

diktion der Form und Pose hinzu. Es wird ein niedrigdimensionaler Formen-Raum aus CAD-Modellen gelernt und mit einer differenzierbaren Render-und-Vergleichs-Verlustfunktion die Schätzung der 3D-Form und 3D-Pose mithilfe von 2D-Daten wie Instanzsegmentierung oder Tiefe überwacht. Beker et al. [BKM+20] rendern nicht nur die Silhouette wie in 3D-RCNN sondern auch die Textur, da diese wertvolle Informationen über die Orientierung des Objektes liefert. In die Verlustfunktion geht neben der Instanz und Tiefe auch der Pixelfehler des gerenderten Bildes, die Bounding Box Differenz sowie die Abweichung der Ausdehnung des Objektes vom Mittelwert ein.

2.2.3 Ansätze basierend auf semantischen Informationen

Frühe Methoden wie [CKZ+16] generieren potentielle 3D-Objektboxen in der Nähe einer geschätzten Bodenebene, welche dann zurück ins Bild projiziert werden und anhand semantischer Klasse und Instanz, Kontur, Objektform, Kontext und Lage bewertet werden. MonoGRNet [QWL19a] besteht aus vier Teilnetzen für 2D-Detektion, Schätzung der Tiefe einzelner Instanzen, 3D-Positionsschätzung und lokaler Eckenregression. Anhand der erkannten 2D Bounding Box schätzt das Netzwerk zunächst die Tiefe und die 2D-Projektion des Zentrums der 3D-Box um die globale 3D-Position zu erhalten und ermittelt dann die Eckkoordinaten im lokalen Kontext. Die endgültige 3D Bounding Box wird abschließend auf Grundlage der geschätzten 3D-Position und der lokalen Ecken im globalen Kontext optimiert.

2.2.4 Ansätze basierend auf der Transformation in die Vogelperspektive

Cai et al. [CLJ+20] bestimmen ein strukturiertes Polygon in 2D, welches der Projektion der 3D-Box entspricht und nutzen die geschätzte Objekthöhe um dessen Tiefe zu bestimmen. Die Tiefenschätzung wird für die Erzeugung einer Vogelperspektive genutzt, mithilfe derer die 3D-Boxen verfeinert werden. Anhand einer ausführlichen Evaluation wird gezeigt, dass die Vogelsperspektive mit direkter räumlicher Information besser für die Bestimmung der 3D-Boxparameter geeignet ist als 2D-Bildmerkmale oder Disparitätsbilder. OFT-Net [RKC19] führt eine orthografische Merkmalstransformation ein, bei der

bildbasierte Merkmale direkt in die Vogelperspektive konvertiert werden. Dies ist gerade dann hilfreich, wenn der Maßstab der einzelnen Objekte drastisch variiert.

2.2.5 Ansätze basierend auf geometrischen Merkmalen

Einige Methoden versuchen ohne die Verwendung weiterer Informationen, wie Semantik, 3D-Modellen oder Tiefe, die 3D-Box nur anhand des geometrischen Zusammenhangs mit der 2D-Box zu schätzen. Einer der ersten Ansätze in diese Richtung ist [MAFK17], welcher die vollständige 3D-Pose und Objektdimension aus der 2D-Box durch Ausnutzung von Randbedingungen aus der projektiven Geometrie bestimmt. Die Kernidee besteht darin, dass die perspektivische Projektion einer 3D Bounding Box genau in die 2D-Detektion passen sollte, deshalb wird gefordert, dass jede Seite der 2D Bounding Box die Projektion von mindestens einer 3D-Box-Ecke berührt.

MonoDIS [SBP+19] verwendet die gleiche Lifting-Transformation wie ROI-10D, führt aber als zusätzliche Verlustfunktion noch eine Konfidenzbewertung der 3D-Boxen ein. Außerdem werden bestimmte Parameter in der Verlustfunktion individuell behandelt um deren Abhängigkeiten aufzulösen und eine Verbesserung der 2D-Detektion durch eine *signed* Intersection over Union (IoU) als Verlustfunktion vorgeschlagen. Diese verhindert das Verschwinden von Gradienten bei nicht überlappenden Bounding Boxen, da in diesen Fällen negative IoUs ausgegeben werden. M3D-RPN [BL19] erstellt direkt Vorschläge für 2D- und 3D-Boxen durch geometrische Zusammenhänge zwischen 2D-Maßstab und 3D-Tiefe. Um Schätzung der 3D-Parameter zu verbessern werden ähnlich wie bei D^4LCN verschiedene Filterkernel verwendet, da aber keinerlei Tiefeninformation vorliegt, hängen diese nur von der Bildzeile ab.

SHIFT R-CNN [NPK+19] erweitert Faster R-CNN um zusätzlich die 3D-Orientierung und die 3D-Maße zu schätzen und kombiniert es mit einem Least-Square-Ansatz zur Lösung des Problems der inversen geometrischen Abbildung von 2D auf 3D unter Verwendung der Kameraprojektionsmatrix. Abschließend wird das Ergebnis mithilfe des ShiftNets basierend auf einem „Volume Displacement Loss“ verfeinert. MonoPSR [KPW19] nutzt die Kameramatrix um Vorschläge für 3D-Boxen zu generieren und prädiziert parallel dazu eine Punktwolke pro Objekt um Form und Maße des Objekts zu bestim-

men und die 2D-3D-Konsistenz zu verbessern. Während des Trainingsprozesses werden dafür LiDAR-Punktwolken verwendet, wodurch die Inflexibilität von CAD-Modellen umgangen werden kann.

RTM3D [LZLC20] ermittelt die 3D-Box durch 9 Merkmalspunkte pro Objekt und geometrische Randbedingungen. Zwei ähnliche Methoden nutzen die in Abschnitt 2.1 vorgestellte Objektrepräsentation mit einem einzigen Merkmalspunkt pro Objekt zur direkten Regression der 3D-Box aus dem Eingangsbild. Bei SMOKE [LWT20] wird dabei das in MonoDIS vorgestellte Konzept der individuellen Behandlung von Parametergruppen in der Verlustfunktion erweitert. MonoPair [CTSL20] wiederum ergänzt den Merkmalspunkt um räumliche Zusammenhänge zwischen den einzelnen Objekten, wie die 3D-Distanz zwischen zwei Objekten.

GS3D [LOS+19] projiziert aus der 2D-Box und Orientierung einen Vorschlag für eine 3D-Box und nutzt sichtbare Flächen des Objektes als Merkmale um diese Schätzung zu verfeinern. Ein ähnlicher Ansatz, der die im Bild sichtbaren Flächen jedoch auch schon zur Initalisierung der 3D-Box verwendet, ist 3D-GCK [GWJ+20]. Dieser erweitert die Bounding Shape Repräsentation um zusätzliche Regressionsparameter, wie das Seitenverhältnis der sichtbaren Seiten und binäre Klassifikatoren wie Front/Heck oder Links/Rechts, die ebenfalls die Sichtbarkeit beschreiben.

FQNet [LLX+19] führt ein Qualitätskriterium ein, welches die 3D-IoU zwischen einer Vielzahl ins Bild zurückprojizierter potentieller 3D-Boxen und dem eigentlichen Objekt allein auf Grundlage von 2D-Informationen abschätzen kann. MoVi-3D [SBP+20] führt die Detektion nicht auf dem Originalbild, sondern auf virtuellen Bildern, welche eine skalierte Version eines Ausschnitts des Originalbilds darstellen, durch. Dadurch erhält man Objekte gleichen Ausmaßes unabhängig von der Tiefe in der Szene und vereinfacht so das Training des 3D-Detektors. SS3D [JZK19] ermittelt einen 26-dimensionalen Parametervektor pro Objekt im Eingangsbild und stellt die Bestimmung der 3D-Boxparameter als gewichtetes nichtlineares Least Squares Problem auf. Dabei werden verschiedene Ansätze untersucht die Gewichte für die einzelnen Parameter der Regression zu lernen.

2.3 Binokulare Objektdetektion

Aktuelle Methoden binokularer Objektdetektion fokussieren sich hauptsächlich auf zwei verschiedene Ansätze. Entweder werden Disparitäten in eine Punktwolkenstruktur gebracht, mit welcher bestehende LiDAR-basierte Detektionsalgorithmen zufriedenstellende Ergebnisse liefern oder es werden Merkmale aus dem linken und rechten Bild extrahiert und damit versucht die 3D-Information direkt zu gewinnen.

Einer der bekanntesten auf Punktwolken basierenden Ansätze ist Pseudo-LiDAR [WCG+19], welches kamerabasierte Tiefenschätzungen in eine LiDAR-Punktwolken Repräsentation konvertiert und darauf verschiedene Detektionsalgorithmen wie AVOD [KML+18] oder Frustum-PointNet [QLW+18] anwendet. Pseudo-LiDAR++ [YWC+20] erweitert diesen Ansatz durch Hinzufügen eines vierzeiligen LiDARs um den Stereotiefenfehler speziell für weit entfernte Objekte korrigieren zu können.

Pon et al. [PKLW20] führen eine Instanzsegmentierung aus und berechnen Disparitäten nur für die jeweiligen Objekte, um das Problem der ungenauen Tiefenschätzung an Objektgrenzen zu umgehen. Aus den daraus berechneten Punktwolken werden mit AVOD die 3D-Objekte bestimmt. ZoomNet [XZY+20] berechnet ebenfalls nur die Disparitäten für einzelne Instanzen, skaliert diese aber zuvor auf eine einheitliche Größe um den Tiefenfehler speziell für weit entfernte Objekte zu verringern. Die Pose jedes Objektes wird dann mithilfe einer CAD-Modell basierten Verlustfunktion aus der instanzweisen Punktwolke geschätzt. Auch Disp R-CNN [SCX+20] bestimmt die Tiefe nur für einzelne Instanzen und verbessert die Disparitätsschätzung durch Erstellen von zusätzlichen Trainingsdaten mithilfe von CAD-Modellen. Dadurch werden selbst zum Training des Netzes keine zusätzlichen LiDAR-Daten benötigt, wie es bei Pseudo-LiDAR der Fall ist. CG-Stereo [LKW20] nutzt semantische Segmentierung um die Disparitätsberechnung für den Vorder- und Hintergrund zu trennen. Dabei wird aus den Ergebnissen des Stereomatchers eine Punktwolke mit Konfidenzwerten berechnet, wobei von den Hintergrundpixeln nur diejenigen verwendet werden, die zur Bodenoberfläche gehören. Beide Ansätze verwenden anschließend PointRCNN zur 3D-Detektion.

Auf die in der Disparität oft fehlerhaften Bereiche an den Objektkanten legt auch CDN [GWH+20] seinen Fokus. In diesen Bereichen entsteht ei-

ne multimodale Wahrscheinlichkeitsverteilung der Disparität, für die CDN den Wasserstein-Abstand zur Ground Truth minimiert. Die Architektur wird auf bestehende stereobasierte neuronale Netze zur Tiefenschätzung wie PSMNet [CC18] oder GANet [ZPYT19] und 3D-Objektdetektoren wie Pseudo-LiDAR++ oder DSGN [CLSJ20] angewendet und verbessert die beiden letztgenannten Ansätze um etwa zwei Prozentpunkte in der Genauigkeit der 3D-Detektion.

Im Gegensatz zu diesen Ansätzen verwendet TLNet [QWL19b] keine pixelgenaue Tiefenschätzung sondern 3D-Merkmalspunkte um Korrespondenzen zwischen RoIs im rechten und linken Bild direkt auf Objektebene zu konstruieren. Mit deren Hilfe kann das Objekt im 3D-Raum detektiert und trianguliert werden. DSGN [CLSJ20] extrahiert Merkmale aus dem linken und rechten Bild und konstruiert eine differenzierbare 3D-Repräsentation, die dann in einem dreidimensionalen neuronalen Netz zur Erkennung der 3D-Objekte verwendet wird. Stereo R-CNN [LCS19] erweitert Faster R-CNN für Stereobilder um Objekte gleichzeitig im linken und rechten Bild detektieren und assoziieren zu können. Anschließend wird die 3D Bounding Box durch einen photometrischen Abgleich auf Grundlage der RoIs bestimmt.

Über eine Vielzahl älterer stereobasierter Objektdetektoren liefern Bernini et al. [BBC+14] einen guten Überblick. Des weiteren sei noch 3DOP [CKZ+15] erwähnt, welcher eine Optimierungsfunktion aufstellt, die Objektgröße, Bodenoberfläche und verschiedene Tiefenmerkmale beinhaltet um Objektvorschläge direkt im 3D-Raum zu behandeln.

3 Grundlagen

In diesem Kapitel werden die für das weitere Verständnis der kamerabasierten Objektdetektion benötigten theoretischen Grundlagen vorgestellt. Dabei wird das Modell einer idealen Lochkamera in Abschnitt 3.1 und im Weiteren der Aufbau und die Eigenschaften binokularer Kamerasysteme beschrieben. Anschließend wird in Abschnitt 3.3 dargelegt, auf welche Weise die 3D-Information der beobachteten Szene mit solchen Systemen rekonstruiert werden kann. In Abschnitt 3.4 werden die für die Schätzung der Bodenoberfläche in Kapitel 4 verwendeten B-Splines definiert. Abschnitt 3.5 geht auf die Methode der kleinsten Fehlerquadrate für lineare und nichtlineare Modellfunktionen ein, welche in Kapitel 5 benötigt wird. Eine kurze Einführung in die für Kapitel 6 und 7 benötigten KNNs bildet mit Abschnitt 3.6 den Abschluss dieses Kapitels.

3.1 Kameramodellierung

Eine Kamera bildet einen Objektpunkt aus der dreidimensionalen Welt auf eine zweidimensionale Bildebene ab. Dieser Abbildungsprozess wird im Allgemeinen durch eine perspektivische Projektion mithilfe des Lochkameramodells beschrieben. Dabei passieren alle in die Kamera einfallenden Sichtstrahlen eine infinitesimal kleine Lochblende, das sog. optische Zentrum $\mathbf{C}$. Dieses Modell genügt um Kameraoptiken mit gewöhnlichen Öffnungswinkeln ausreichend gut annähern zu können.

Zur mathematischen Beschreibung dieses Abbildungsprozesses spielen homogene Koordinaten eine wesentliche Rolle. Jeder Punkt im n-dimensionalen euklidischen Raum kann durch $n+1$ homogene Koordinaten beschrieben werden. Für einen 3D-Punkt $\mathbf{x}_\mathrm{W} = [X_\mathrm{W}, Y_\mathrm{W}, Z_\mathrm{W}]^\mathrm{T}$ liegen die homogenen Koordinaten beispielsweise als 4D-Vektor der Form $\tilde{\mathbf{x}}_\mathrm{W} = \lambda[X_\mathrm{W}, Y_\mathrm{W}, Z_\mathrm{W}, 1]^\mathrm{T}$ mit $\lambda \in \mathbb{R}\setminus\{0\}$ vor. An diesem Beispiel lassen sich direkt zwei wichtige Eigenschaften homo-

gener Koordinaten veranschaulichen. Zum einen erhalten wir durch Division der ersten n Elemente durch das $(n+1)$-te Element des homogenen Koordinatenvektors die ursprünglichen euklidischen Koordinaten. Zum anderen sind zwei Punkte in homogenen Koordinaten genau dann identisch, wenn ein Skalierungsfaktor $\lambda \in \mathbb{R} \setminus \{0\}$ existiert, mit welchem sie ineinander überführt werden können. Für diese Identität wird in dieser Arbeit die Schreibweise „$\simeq$“ verwendet. Hauptvorteil homogener Koordinaten ist, dass sich alle Kollineationen einheitlich durch lineare Abbildungen und damit durch Matrizen beschreiben lassen.

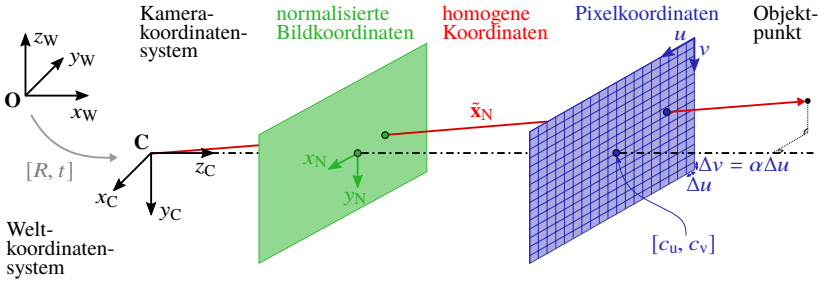


Abbildung 3.1: Perspektivischer Abbildungsprozess einer Lochkamera. Die Z_C-Achse wird als optische Achse bezeichnet und entspricht der Lotgeraden zur Bildebene durch das optische Zentrum.

Betrachten wir zunächst die 3D-Transformation von Welt- in Kamerakoordinaten aus Abbildung 3.1. Das Kamerakoordinatensystem, dessen Ursprung sich im optischen Zentrum der Kamera befindet, ist eindeutig durch eine Rotationsmatrix $\mathbf{R} \in \mathbb{R}^{3\times3}$ und einen Translationsvektor $\mathbf{t} \in \mathbb{R}^{3\times1}$ relativ zum weltfesten Koordinatensystem W beschrieben:

$$\mathbf{x}_C = \mathbf{R}\mathbf{x}_W + \mathbf{t}. \tag{3.1}$$

Bei Verwendung von homogenen Koordinaten lässt sich diese Transformation in einer einzigen Multiplikation mit der Matrix $\mathbf{M} \in \mathbb{R}^{4\times4}$ zusammenfassen:

$$\tilde{\mathbf{x}}_C = \mathbf{M}\tilde{\mathbf{x}}_W\,, \text{ mit } \mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix}. \tag{3.2}$$

Diese affine Transformation stellt bei der Verkettung von Koordinatentransformationen eine Vereinfachung dar und ist deshalb im Bereich des maschinellen Sehens [HZ04] weit verbreitet. Die Parameter $(\mathbf{R}, \mathbf{t})$ sind für das Abbildungsmodell der Kamera von großer Bedeutung und werden als extrinsische Parameter bezeichnet.

Auch das bereits erwähnte ideale Lochkameramodell lässt sich mithilfe von homogenen Koordinaten elegant formulieren. Der vom, in Kamerakoordinaten transformierte, Objektpunkt $\mathbf{x}_\mathrm{C}$ ausgehende Lichtstrahl durch das optische Zentrum trifft die normalisierte Bildebene, dadurch ergeben sich die sog. normalisierten Bildkoordinaten

$$x_\mathrm{N} = X_\mathrm{C}/Z_\mathrm{C} \tag{3.3}$$
$$y_\mathrm{N} = Y_\mathrm{C}/Z_\mathrm{C}. \tag{3.4}$$

Daraus ergibt sich in homogenen Koordinaten die einfache Matrixmultiplikation

$$\tilde{\mathbf{x}}_\mathrm{N} \simeq \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{x}}_\mathrm{C}. \tag{3.5}$$

Diese normalisierten Bildkoordinaten werden schlussendlich über eine affine Transformation der Form

$$\tilde{\mathbf{x}}_\mathrm{P} = \mathbf{K}\tilde{\mathbf{x}}_\mathrm{N} \tag{3.6}$$

in Pixelkoordinaten überführt, wobei $\mathbf{K}$ durch

$$\mathbf{K} = \begin{bmatrix} f & 0 & c_\mathrm{u} \\ 0 & \alpha f & c_\mathrm{v} \\ 0 & 0 & 1 \end{bmatrix} \tag{3.7}$$

gegeben ist. Die Parameter der Matrix **K** werden auch als intrinsische Parameter der Kamera bezeichnet und beschreiben:

- die Brennweite f in Pixeln,
- das Seitenverhältnis α zwischen der horizontalen und vertikalen Seitenlänge eines Pixels,
- den Bildhauptpunkt (c_u, c_v).

Fassen wir nun die Gleichungen 3.2, 3.5 und 3.6 zusammen, erhalten wir das projektive Kameramodell für die Abbildung eines 3D-Weltpunktes $\tilde{\mathbf{x}}_W$ auf die Bildposition $\tilde{\mathbf{x}}_P$:

$$\tilde{\mathbf{x}}_P = \begin{bmatrix} u \\ v \\ w \end{bmatrix} \simeq \mathbf{K}\mathbf{M}\tilde{\mathbf{x}}_W = \mathbf{P}\tilde{\mathbf{x}}_W. \tag{3.8}$$

Die 3×4-Matrix **P** wird als Projektionsmatrix bezeichnet und beinhaltet sowohl intrinsische als auch extrinsische Kameraparameter. Daraus ergibt sich in euklidischen Koordinaten:

$$\mathbf{x}_P = \begin{bmatrix} u/w \\ v/w \end{bmatrix}. \tag{3.9}$$

3.2 Modell eines Stereokamerasystems

Bei der zuvor beschriebenen Projektion eines dreidimensionalen Objektpunktes auf die zweidimensionale Bildebene handelt es sich um eine nicht bijektive Abbildung, das heißt es existiert keine eindeutige inverse Abbildung der Bildinformation in die beobachtete Szenengeometrie. Betrachtet man dieselbe Szene jedoch aus zwei unterschiedlichen Blickwinkeln, kann die Tiefeninformation für alle Bildpunkte bestimmt werden, welche in beiden Ansichten existieren. Zur Beschreibung eines solchen Kamerasystems wird das Modell einer monokularen Lochkamera um eine zweite Kamera erweitert. Die resultierende

Stereoanordnung besteht aus zwei Kameras mit den optischen Zentren $\mathbf{C}_\mathrm{L}$ und $\mathbf{C}_\mathrm{R}$, die denselben Szenenpunkt $\mathbf{x}_\mathrm{W}$ abbilden. Zur Unterscheidung der beiden Kameras werden die Indizes *L* (links) und *R* (rechts) verwendet. Aus praktischen Gründen wird das Weltkoordinatensystem häufig so gewählt, dass es mit einem der beiden Kamerakoordinatensysteme übereinstimmt. In dieser Arbeit wird dazu die linke Stereokamera gewählt, d.h. die extrinsischen Parameter der linken Kamera ergeben sich zu $\mathbf{R}_\mathrm{L} = \mathbf{I}$, $\mathbf{t}_\mathrm{L} = \mathbf{0}$ und die der rechten zu $\mathbf{R}_\mathrm{R} = \mathbf{R}$, $\mathbf{t}_\mathrm{R} = \mathbf{t}$. Die jeweiligen Projektionsmatrizen lauten somit $\mathbf{P}_\mathrm{L} = \mathbf{K}_\mathrm{L}[\mathbf{I}, \mathbf{0}]$ bzw. $\mathbf{P}_\mathrm{R} = \mathbf{K}_\mathrm{R}[\mathbf{R}, \mathbf{t}]$. Die Positions- und Orientierungsänderung der rechten Kamera kann somit durch eine starre Transformation ausgedrückt werden. Ein wichtiger Designparameter eines Stereosystems ist die Basisbreite $b = |\mathbf{t}|$, welche den Abstand zwischen den optischen Zentren beider Kameras ausdrückt. Die richtige Wahl von *b* ist ein Kompromiss aus zwei gegensätzlichen Anforderungen. Eine große Basisbreite bewirkt eine höhere Tiefenauflösung, erschwert allerdings die Suche nach korrespondierenden Punkten im Stereobild, da sich perspektivische Effekte stärker auswirken und Verdeckungen häufiger auftreten. Vor allem im Nahbereich ist so zumeist kein Stereosehen möglich.

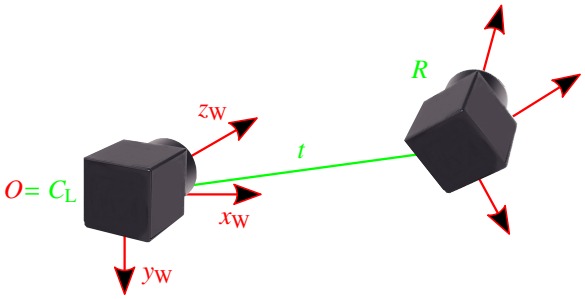


Abbildung 3.2: Extrinsische Modellierung einer Stereokamera.

Insgesamt weist dieses Modell eines Stereokamerasystems zehn intrinsische und sechs extrinsische Parameter auf. Um die praktische Bedeutung dieser Parameter zu analysieren, soll zunächst die Hauptaufgabe eines Stereosystems, die 3D-Rekonstruktion, genauer beschrieben werden.

3.3 3D-Rekonstruktion aus Stereobildern

Zur Gewinnung der Tiefeninformation aus einem Stereobild müssen zwei Problemstellungen gelöst werden [Fau93]:

1. Das Korrespondenzproblem: Wie lassen sich Paare von korrespondierenden Bildpunkten finden, die dem gleichen Objektpunkt entsprechen?
2. Das Rekonstruktionsproblem: Wie kann aus einem gegebenen Korrespondenzpaar die 3D-Position des zugehörigen Objektpunktes $\mathbf{x}_W$ bestimmt werden?

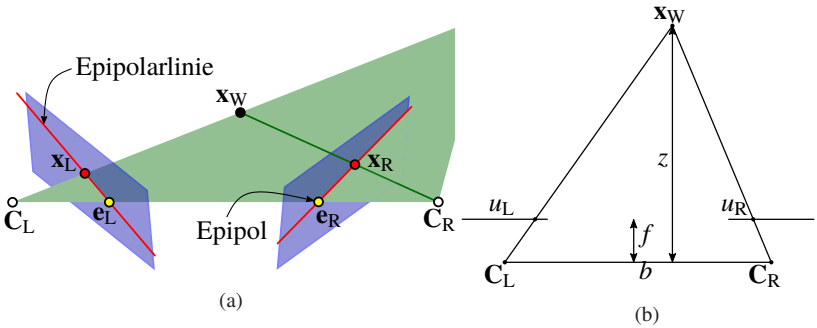


Abbildung 3.3: a) Epipolargeometrie einer Stereokamera. b) Draufsicht einer rektifizierten Stereoanordnung.

Mithilfe der in Abbildung 3.3a dargestellten Epipolargeometrie vereinfacht sich das Korrespondenzproblem erheblich. Jeder Bildpunkt $\mathbf{x}_L$ spannt mit den beiden optischen Zentren $\mathbf{C}_L$ und $\mathbf{C}_R$ eine Ebene auf, die sowohl den Objektpunkt $\mathbf{x}_W$ als auch den zugehörigen Punkt $\mathbf{x}_R$ im rechten Bild enthält. Die Schnittgerade dieser Ebene mit den jeweiligen Bildebenen wird Epipolarlinie genannt. Die Gerade, welche die optischen Zentren der Kameras verbindet, durchstößt die beiden Bildebenen in jeweils einem Punkt, dem Epipol. Dort treffen sich alle Epipolarlinien einer Kamera. Der Epipol eines Bildes ist dementsprechend die Abbildung des optischen Zentrums der jeweils anderen Kamera. Die so gegebene Geometrie vereinfacht die Korrespondenzanalyse signifikant, indem sie den Suchbereich für korrespondierende Punkte auf nur eine Linie, die Epipolarlinie, beschränkt.

Eine weitere Vereinfachung der Korrespondenzsuche liegt vor, wenn beide Kameras identische intrinsische Parameter aufweisen, die Bildebenen der Kameras koplanar und die Kamerakoordinatensysteme achsparallel sind. Dadurch verschieben sich die Epipole ins Unendliche und die Epipolarlinien werden zu parallelen Geraden. Man spricht dann von einem rektifizierten Stereosystem. In der Praxis sind solche Anordnungen zwar kaum umsetzbar, allerdings können bei bekannten intrinsischen und extrinsischen Kameraparametern reale Stereobilder in ideale, rektifizierte Bilder transformiert werden. Detaillierte Beschreibungen gängiger Rektifizierungsverfahren finden sich zum Beispiel in [PKVG99] und [FTV00]. Nach der Rektifizierung ergeben sich Bilder, die eine Stereoanordnung mit $\mathbf{R}_\mathrm{R} = \mathbf{I}$, $\mathbf{t}_\mathrm{R} = [b, 0, 0]^T$ aufgenommen hätte. Wichtigste Eigenschaft rektifizierter Stereobilder ist, dass korrespondierende Bildpunkte stets in der gleichen Bildzeile liegen [TV98].

Zur Lösung des Korrespondenzproblems rektifizierter Stereosysteme existieren in der Literatur viele unterschiedliche Ansätze. Scharstein und Szeliski [SS02] und Brown et al. [BBH03] geben eine ausführliche Übersicht sowohl zu lokalen als auch globalen Verfahren.

Bei lokalen (fensterbasierten) Algorithmen hängt die Disparitätsberechnung an einem bestimmten Punkt nur von Intensitätswerten innerhalb eines endlichen Fensters ab. Neben dem *Feature Matching*, welches Korrespondenzen nur für einige Merkmalspunkte liefert und gradientenbasierten Verfahren zur Berechnung des optischen Flusses gehören zu den lokalen Verfahren vor allem *Block Matching* Algorithmen. Diese suchen für den Bereich um den jeweiligen Punkt den maximalen Matchingwert oder die minimalen Kosten durch Nutzung von Varianten der Kreuzkorrelation oder der Rank- oder Census-Transformation. Bei letzteren handelt es sich um nicht-parametrische Transformationen, die nur von der relativen Anordnung der Intensitäten und nicht von den tatsächlichen Intensitätswerten abhängen. Dadurch ergeben sich Vorteile bei monotonen Helligkeitsschwankungen und in Bereichen von multimodalen Intensitätsverteilungen zum Beispiel an Objektgrenzen [ZW94].

Globale Methoden stellen Nebenbedingungen an die Glattheit über einzelne Zeilen oder das gesamte Bild und lösen ein globales Optimierungsproblem. Dadurch sind sie weniger anfällig für mehrdeutige Regionen in Bildern wie zum Beispiel Verdeckungen oder Bereiche mit einheitlicher Textur. Häufigste Anwendung finden dabei dynamische Programmierung und *Graph Cuts*. Ein großer Nachteil ist allerdings der höhere Rechenaufwand.

Ein weit verbreitetes Verfahren, welches Techniken von lokalen und globalen Ansätzen kombiniert, ist das Semi-global Matching [Hir07]. Dabei findet die Korrespondenzsuche ebenfalls durch fensterbasierte Korrelation statt, wird aber durch eine globale Kostenfunktion unterstützt, welche entlang von acht Richtungen über das Bild optimiert wird. Neben der guten Parallelisierbarkeit und Robusheit gegen Helligkeitsschwankungen im Bild, zeichnet sich dieses Verfahren auch dadurch aus, dass die Eingangsbilder zwar eine bekannte Epipolargeometrie besitzen, jedoch nicht rektifiziert sein müssen.

Aber auch das Rekonstruktionsproblem kann im Fall rektifizierter Stereobilder leicht gelöst werden. Aus Abbildung 3.3b ergibt sich durch Anwendung des Strahlensatzes:

$$\frac{z}{b} = \frac{f}{u_\mathrm{L} - u_\mathrm{R}} \Rightarrow z = \frac{bf}{d}. \tag{3.10}$$

Die Größe $d = u_\mathrm{L} - u_\mathrm{R}$ wird als Disparität bezeichnet und repräsentiert die Verschiebung eines Objektpunktes zwischen linkem und rechtem Bild in Pixeln. Zu jeder Pixelposition $(u, v)^\mathrm{T}$ ergibt sich dementsprechend eine Disparität, mit welcher die 3D-Position des Objektpunktes rekonstruiert werden kann:

$$\mathbf{x}_\mathrm{W} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{b}{d} \begin{bmatrix} u - c_\mathrm{u} \\ v - c_\mathrm{v} \\ f \end{bmatrix}. \tag{3.11}$$

Für den Fall nicht-rektifizierter Stereobilder sei auf das Triangulationsverfahren von Hartley und Sturm [HS97] verwiesen, welches bei bekannten Kameraparametern die 3D-Position eines Objektpunktes iterativ aus fehlerbehafteten Punktkorrespondenzen schätzt.

3.4 B-Splines

Splines d-ten Grades sind Funktionen, die aus stückweise definierten Polynomen höchstens d-ten Grades zusammengesetzt sind. Dabei wird an die Verbindungspunkte an denen zwei Polynomstücke zusammenstoßen, auch Knoten

genannt, unter anderem die Bedingung gestellt, dass der Spline $d-1$ mal stetig differenzierbar ist. Durch die stückweise Definition sind Splines flexibler als Polynome und sorgen dafür, dass sich komplexe Formen relativ einfach mathematisch beschreiben lassen.

Jeder Spline einer bestimmten Ordnung lässt sich als Linearkombination von Basis-Splines derselben Ordnung darstellen [DB78]. Die daraus entstehenden Splines werden B-Spline-Kurven genannt. Eine B-Spline-Kurve der Ordnung d hat die Form

$$s(x) = \sum_{i=0}^{n} B_{i,d}(x) c_i = \mathbf{B_d}(x)^T \mathbf{c} \tag{3.12}$$

mit den Kontrollpunkten (auch de Boor-Punkte genannt)

$$\mathbf{c} = [c_0, \ldots, c_n]^T. \tag{3.13}$$

Die Basis-Splines $B_{i,j}(x)$ können ausgehend von einem Knotenvektor mit monoton steigenden Werten $\mathbf{t} = [t_0, \ldots, t_{n+d+1}]$ nach der Cox-de Boor-Rekursionsformel effektiv berechnet werden [DB78]:

$$B_{i,0}(x) = \begin{cases} 1 & \text{wenn } x \in [t_i, t_{i+1}) \\ 0 & \text{sonst} \end{cases} \tag{3.14}$$

$$B_{i,j}(x) = \frac{x - t_i}{t_{i+j} - t_i} B_{i,j-1}(x) + \frac{t_{i+j+1} - x}{t_{i+j+1} - t_{i+1}} B_{i+1,j-1}(x). \tag{3.15}$$

Splines deren Knotenpunkte auf dem gesamten Gültigkeitsbereich äquidistant angeordnet sind, nennt man uniform. Die Verwendung solcher B-Splines bringt den Vorteil, dass für jedes Segment dieselbe Basis genutzt und deshalb vorberechnet werden kann, was zu Laufzeitvorteilen führt.

3.5 Methode der kleinsten Fehlerquadrate

Die Methode der kleinsten Fehlerquadrate (engl. Least Squares) ist das Standardverfahren in der Ausgleichsrechnung zur Approximation der Lösung überbestimmter Systeme. Das Ziel ist es den n-dimensionalen Parametervektor $\boldsymbol{\beta}$ einer Modellfunktion f so an die m Messungen y_i anzupassen, dass die Summe der quadratischen Residuen (Abweichungen) von f zu jeder Messung minimal wird:

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \sum_i (y_i - f(x_i, \boldsymbol{\beta}))^2 \tag{3.16}$$

$$= \arg\min_{\boldsymbol{\beta}} \|\mathbf{r}\|^2. \tag{3.17}$$

Hierbei bezeichnet x_i den Punkt der Modellfunktion, an welchem y_i beobachtet wurde und $\mathbf{r}$ den Residuenvektor über alle Messungen. Wie genau dieses Minimierungsproblem gelöst wird, hängt von der Art der Modellfunktion ab.

3.5.1 Lineare Modellfunktion

Betrachten wir zunächst den einfachsten Fall eines linearen Beobachtungsmodells

$$\mathbf{y} = \mathbf{H}\boldsymbol{\beta} \tag{3.18}$$

mit der Beobachtungsmatrix $\mathbf{H} \in \mathbb{R}^{m \times n}$, welche den Zusammenhang zwischen den idealen Beobachtungen $\mathbf{y}$ und den „wahren" Parametern $\boldsymbol{\beta}$ beschreibt. Der Residuenvektor ergibt sich durch die Differenz aus den unsicherheitsbehafteten Beobachtungen und der Schätzung des Modells:

$$\hat{\mathbf{r}} = \hat{\mathbf{y}} - \mathbf{H}\hat{\boldsymbol{\beta}}. \tag{3.19}$$

Die Methode der kleinsten Fehlerquadrate bestimmt die Schätzwerte für $\hat{\boldsymbol{\beta}}$, welche die quadratische euklidische Norm des Residuenvektors in Form der Gütefunktion

$$J(\hat{\boldsymbol{\beta}}) = \|\mathbf{r}\|^2 = \hat{\mathbf{y}}^T\hat{\mathbf{y}} - 2\hat{\mathbf{y}}^T\mathbf{H}\hat{\boldsymbol{\beta}} + \hat{\boldsymbol{\beta}}^T\mathbf{H}^T\mathbf{H}\hat{\boldsymbol{\beta}} \tag{3.20}$$

minimiert. Da J eine in $\hat{\boldsymbol{\beta}}$ quadratische Funktion ist, gibt es genau einen Wert von $\hat{\boldsymbol{\beta}}$ für den J optimal wird. Das Nullsetzen der Ableitung

$$\frac{dJ(\hat{\boldsymbol{\beta}})}{d\hat{\boldsymbol{\beta}}} = -2\mathbf{H}^T\hat{\mathbf{y}} + 2\mathbf{H}^T\mathbf{H}\hat{\boldsymbol{\beta}} = \mathbf{0} \tag{3.21}$$

liefert die Lösung

$$\hat{\boldsymbol{\beta}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\hat{\mathbf{y}}. \tag{3.22}$$

Die Matrix $(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T$ wird auch als Pseudoinverse bezeichnet [PLB15].

3.5.2 Nichtlineare Modellfunktion

Oftmals sind die Residuen allerdings nichtlinear in der Optimierungsvariable $\boldsymbol{\beta}$. In diesen Fällen wird versucht sich iterativ einem lokalen Minimum anzunähern indem beginnend von einem Anfangswert $\boldsymbol{\beta_0}$ die Lösung mit jedem Schritt $\mathbf{s}$ durch $\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k} + \mathbf{s_k}$ verbessert wird.

Ein weit verbreiteter Ansatz ist das Gauß-Newton-Verfahren. Die Grundidee dieses Verfahrens besteht darin, den Residuen-Vektor $\mathbf{r}(\boldsymbol{\beta})$ mit einer Taylorentwicklung 1.Ordnung an der Stelle der aktuellen Lösung $\boldsymbol{\beta_k}$ zu linearisieren:

$$\mathbf{r}(\boldsymbol{\beta}) \approx \mathbf{r}(\boldsymbol{\beta_k}) + \mathbf{J}(\boldsymbol{\beta_k})(\boldsymbol{\beta} - \boldsymbol{\beta_k}). \tag{3.23}$$

Die Matrix J ist hier die Jakobi-Matrix

$$\mathbf{J}(\boldsymbol{\beta}) = \begin{bmatrix} \frac{\partial r_1}{\partial \beta_1} & \cdots & \frac{\partial r_1}{\partial \beta_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial r_m}{\partial \beta_1} & \cdots & \frac{\partial r_m}{\partial \beta_n} \end{bmatrix}. \tag{3.24}$$

In jeder Iteration lösen wir also ein lineares Least Squares Problem für die Korrektur

$$\mathbf{s_k} = \arg\min_{\mathbf{s}} \|\mathbf{r}(\boldsymbol{\beta_k}) + \mathbf{J}(\boldsymbol{\beta_k})\mathbf{s}\|^2. \tag{3.25}$$

Das Gauss-Newton-Verfahren konvergiert sehr schnell für mäßig nichtlineare Residuen. Liegt der Anfangswert allerdings weit entfernt von der Lösung oder handelt es sich um ein stark nichtlineares Problem, kann es passieren das diese Methode gar nicht konvergiert [Bjö96]. Abhilfe bietet eine modifizierte Variante dieses Verfahrens, bei welcher im Update-Schritt eine Schrittweite α_k eingeführt wird:

$$\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k} + \alpha_k \mathbf{s_k}. \tag{3.26}$$

Es gibt einige Möglichkeiten eine geeignete Schrittweite zu finden. Eine einfache, aber weit verbreitete ist es, bei $\alpha_k = 1$ zu beginnen und dieses zu halbieren, sobald

$$\|\mathbf{r}(\boldsymbol{\beta_k})\|^2 - \|\mathbf{r}(\boldsymbol{\beta_k} + \alpha_k \mathbf{s_k})\|^2 \geq \alpha_k \|\mathbf{J}(\boldsymbol{\beta_k})\mathbf{s}\|^2 \tag{3.27}$$

nicht mehr gilt [Bjö96]. Diese Ungleichung ist ein unmittelbares Maß dafür, wie gut die Linearisierung den nichtlinearen Residuenvektor für die Schrittweite α_k entlang der Schrittrichtung $\mathbf{s_k}$ approximiert.

Beide Versionen des Gauß-Newton-Verfahrens können Probleme verursachen, wenn die Jakobi-Matrix in einer Iteration nicht oder nur annähernd vollen Rang hat [NW06]. Das Levenberg-Marquardt-Verfahren löst dieses Problem und ist deutlich robuster als das Gauß-Newton-Verfahren. Das heißt, es konvergiert mit einer hohen Wahrscheinlichkeit auch bei schlechten Startbedingungen. Es

löst das gleiche Minimierungsproblem wie Gauß-Newton, legt aber zusätzlich eine Beschränkung der Schrittweite fest, welche sich durch

$$\|\mathbf{D_k s_k}\| \leq \Delta_k \tag{3.28}$$

mit einer Diagonalmatrix $\mathbf{D_k}$ und dem Trust Region-Radius Δ_k ergibt. Die Matrix $\mathbf{D_k}$ kann dabei so gewählt werden, dass sie die Information über die verschiedenen Gradienten der Dimensionen von $\mathbf{s_k}$ enthält. Ist der Gradient in einer Dimension klein, so sollen in dieser Dimension größere Schritte zugelassen werden um eine langsame Konvergenz zu verhindern. Dies wird durch eine geringe Gewichtung für diese Dimension durch das entsprechende Diagonalelement in $\mathbf{D_k}$ erreicht. Gleichung 3.28 wird dem Minimierungsproblem als Nebenbedingung mit dem Lagrange-Multiplikator λ_k hinzugefügt. Dadurch ergibt sich das lineare Least Squares Problem

$$\mathbf{s_k} = \arg\min_{\mathbf{s}} \left\| \begin{bmatrix} \mathbf{J}(\boldsymbol{\beta_k}) \\ \sqrt{\lambda_k}\mathbf{D_k} \end{bmatrix} \mathbf{s} + \begin{bmatrix} \mathbf{r}(\boldsymbol{\beta_k}) \\ \mathbf{0} \end{bmatrix} \right\|^2 . \tag{3.29}$$

Der Radius der Trust Region wird iterativ korrigiert, wobei der Startwert Δ_0 möglichst groß gewählt wird. Sobald die Diskrepanz zwischen der Linearisierung und der nichtlinearen Funktion zu groß wird, wird der Radius verkleinert. Das Maß

$$\rho_k = \frac{\|\mathbf{r}(\boldsymbol{\beta_k})\|^2 - \|\mathbf{r}(\boldsymbol{\beta_k} - \mathbf{s_k})\|^2}{\|\mathbf{r}(\boldsymbol{\beta_k})\|^2 - \|\mathbf{r}(\boldsymbol{\beta_k}) + \mathbf{J}(\boldsymbol{\beta_k})\mathbf{s_k}\|^2} \tag{3.30}$$

wird zur Bewertung der berechneten Aktualisierung $\mathbf{s_k}$ verwendet. Liegt ρ_k nahe Eins, hat sich die lokale Approximation als gültig erwiesen, der Schritt wird akzeptiert und die Trust Region vergrößert. Ist ρ_k zu klein, wird die Trust Region verkleinert und $\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k}$ gesetzt [Bjö96].

3.6 Künstliche Neuronale Netze

KNNs sind Techniken des maschinellen Lernens, die vage von biologischen neuronalen Netzen inspiriert sind, aus denen Gehirne bestehen. Ein KNN basiert auf einer Sammlung miteinander verbundener Knoten, die als künstliche Neuronen bezeichnet werden. Wie die Synapsen in einem Gehirn kann jede Verbindung ein Signal an andere Neuronen übertragen.

3.6.1 Das Perzeptron

Das Perzeptron wurde 1958 von Rosenblatt [Ros58] vorgestellt und ursprünglich mit tatsächlichen Hardwareschaltungen implementiert. Es wurde als Gerät für die Bilderkennung konzipiert und verfügte über eine Anordnung von 400 Fotozellen. Die Gewichte waren in Potentiometern kodiert, und die Aktualisierung der Gewichte während des Lernens erfolgte durch Elektromotoren [Bis06]. Betrachtet man den zugrunde liegenden Algorithmus und nicht die Maschine, kann ein Perzeptron als das einfachste neuronale Netz betrachtet werden, das ein einzelnes Neuron repräsentiert.

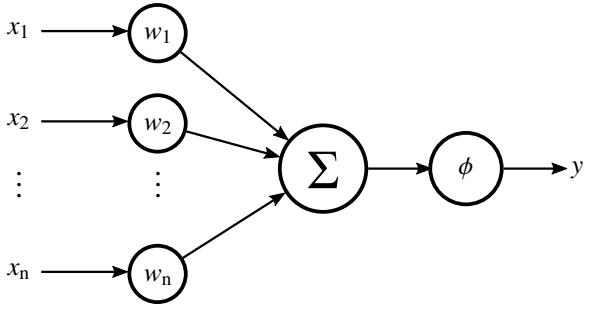


Abbildung 3.4: Schematische Darstellung eines einfachen Perzeptrons.

Das einfache Perzeptron, wie es in Abbildung 3.4 dargestellt ist, entspricht einem binären Klassifikator, welcher einem Eingang $\mathbf{x}$ mit n Merkmalsvariablen eine Klasse y zuordnet. Zunächst wird die gewichtete Summe aller Eingabewerte gebildet:

$$\sigma = \sum_{i=1}^{n} x_i w_i . \tag{3.31}$$

Die Ausgabe $y = \phi(\sigma)$ wird dann mit der Aktivierungsfunktion ϕ bestimmt. Für diese kommen eine Vielzahl unterschiedlicher Funktionen in Frage. Eine Möglichkeit ist die Verwendung einer einfachen Stufenfunktion der Form

$$\phi(\sigma) = \begin{cases} 1 & \sigma \geq 0 \\ -1 & \sigma < 0. \end{cases} \tag{3.32}$$

Diese Vorzeichenfunktion (engl. Sign function) führt zu einer „scharfen“ Aktivierung des Neurons. Durch ihre Nichtdifferenzierbarkeit kann sie jedoch nicht während des Trainings eingesetzt werden. Eine Alternative bietet unter anderem die differenzierbare Sigmoid-Funktion:

$$\mathrm{sig}(\sigma) = \frac{1}{1 + e^{-\sigma}}. \tag{3.33}$$

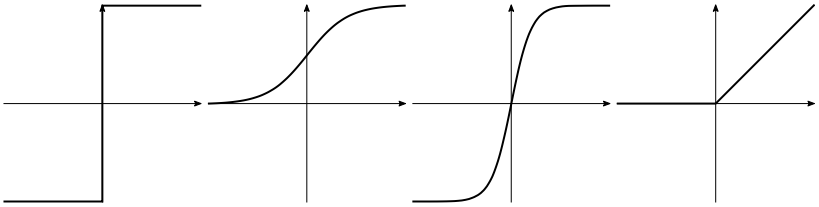
Wie in Abbildung 3.5 zu sehen, führt die sigmoid-Funktion zu einer weicheren Aktivierung des Neurons. Die ebenfalls dargestellte Tangens hyperbolicus Funktion

$$\tanh(\sigma) = \frac{e^{2\sigma} - 1}{e^{2\sigma} + 1} \tag{3.34}$$

besitzt eine ähnliche Form wie die Sigmoid-Funktion, ist dieser aber vorzuziehen, wenn der Ausgang sowohl positive als auch negative Werte liefern soll. Außerdem ist sie durch ihre Zentrierung um Null und den größeren Gradienten leichter trainierbar. Eine weitere populäre Aktivierungsfunktion ist die ReLU (engl. Rectified Linear Unit) Funktion:

$$\phi(\sigma) = \max(0, \sigma). \tag{3.35}$$

Durch die effiziente und schnelle Berechnung hat sich die ReLU-Funktion, insbesondere für die im nächsten Abschnitt vorgestellten mehrschichtigen neuronalen Netze, in den letzten Jahren gegenüber den zuvor genannten Aktivierungsfunktionen durchgesetzt.



(a) Sign (b) Sigmoid (c) Tanh (d) ReLU

Abbildung 3.5: Beispiele von Aktivierungsfunktionen.

3.6.2 Mehrschichtige neuronale Netze

Das Perzeptron enthält eine Eingabe- und eine Ausgabeschicht, von denen die Ausgabeschicht die einzige rechenausführende Schicht ist. Die Eingabeschicht überträgt die Daten an die Ausgabeschicht und alle Berechnungen sind für den Benutzer vollständig sichtbar. Mehrschichtige neuronale Netze, auch mehrlagige Perzeptronen (MLP) genannt, enthalten mehrere Rechenschichten. Die zusätzlichen Zwischenschichten werden als verdeckte Schichten bezeichnet, da die durchgeführten Berechnungen für den Benutzer nicht sichtbar sind. Sind die Ausgänge jeder Schicht nur mit Eingängen der nachfolgenden Schicht verknüpft, spricht man von einem Feed-Forward-Netz [Agg18], welches in Abbildung 3.6 beispielhaft dargestellt ist.

Die Ausgabe $\mathbf{f}_{\mathbf{w}}(\mathbf{x})$ hängt sowohl von den Eingabedaten als auch von der Gesamtheit der während des Trainings ermittelten Gewichte $\mathbf{w} = \cup_l \mathbf{W}_l$ ab. $\mathbf{W}_l$ bezeichnet dabei die Gewichtsmatrix der l-ten Schicht. Ziel des Trainings ist es, diese Gewichte so zu bestimmen, dass eine von der Prädiktion der Trainingseingangsdaten $\mathbf{x}$ und der Ground Truth $\mathbf{y}$ abhängige Verlustfunktion L minimiert wird:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} L(\mathbf{y}, \mathbf{f}_{\mathbf{w}}(\mathbf{x})). \tag{3.36}$$

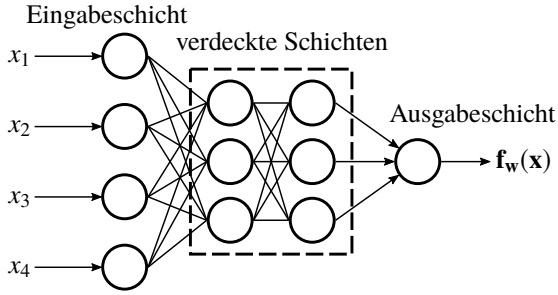


Abbildung 3.6: Schematische Darstellung eines Feed-Forward Netzes.

Aufgrund der großen Parameterzahl ist es meistens jedoch nicht möglich die Optimierung der Gewichte für den kompletten Trainingsdatensatz durchzuführen. Deshalb findet die Optimierung in mehreren tausend Schritten mit jeweils einem kleinen Teil der Daten statt, dem sogenannten *Mini-Batch*. Ein kompletter Durchlauf des Trainingsdatensatzes wird dabei als Epoche bezeichnet.

Am gebräuchlichsten für das Training neuronaler Netze ist das gradientenbasierte Lernen, bei dem der Gradient der Verlustfunktion zur Aktualisierung der Parameter verwendet wird. Für ein einzelnes Perzeptron ist dieser Trainingsprozess relativ einfach, da der Fehler als direkte Funktion der Gewichte berechnet werden kann, was eine einfache Gradientenberechnung ermöglicht. Im Falle von mehrschichtigen Netzen besteht das Problem darin, dass der Fehler eine Zusammensetzung von Funktionen der Gewichte aus vorherigen Schichten ist. Für die Berechnung des Gradienten dieser Kompositionsfunktion wird deshalb der Fehlerrückführungsalgorithmus (engl. backpropagation) angewendet. Dieser besteht aus zwei Schritten, der Prädiktion (Forward) und der Anpassung der Gewichte (Backward). Zuerst wird eine Trainingsinstanz in das Netz gegeben und die Ausgabe über alle Schichten hinweg mit den aktuellen Gewichten berechnet. Mithilfe der vorhergesagten Ausgabe und der Ground Truth der Trainingsinstanz wird die Verlustfunktion des Netzes berechnet. Die Ableitung dieser Funktion wird nun in Bezug auf die Gewichte aller Schichten zur Eingabeschicht zurück propagiert. Dabei werden die Gewichte entgegen des Gradientenanstiegs der Verlustfunktion gemäß

$$\mathbf{W}_l^{s+1} = \mathbf{W}_l^s - \alpha \cdot \frac{\partial L}{\partial \mathbf{W}_l^s} \tag{3.37}$$

angepasst, wobei s den aktuellen Schritt darstellt. Wie schon in Abschnitt 3.5.2, legt α auch hier die Schrittweite des Optimierungsschrittes fest und wird im Zusammenhang mit neuronalen Netzen als Lernrate bezeichnet.

3.6.3 Convolutional Neural Networks

Bisher wurden ausschließlich vollständig verbundene (engl. fully connected) Netze betrachtet. Bei der Verarbeitung großer Bilder scheitern MLPs jedoch an der hohen Anzahl an Gewichten und somit zu optimierenden Parametern. Abhilfe schaffen hier CNNs, welche speziell für Daten entworfen wurden, die eine gitterartige Topologie mit räumlichen Abhängigkeiten aufweisen. CNNs sind neuronale Netze, die in mindestens einer ihrer Schichten die Faltung anstelle der allgemeinen Matrixmultiplikation verwenden [GBC16].

Faltungsschicht

Eine Faltung $f * g$ zweier Funktionen f und g ist definiert durch

$$(f * g)(x) = \int_{-\infty}^{\infty} f(\tau)g(x - \tau)d\tau. \tag{3.38}$$

Anschaulich ist das Ergebnis der Faltung ein gewichteter Mittelwert der Funktion f (Eingang) mit dem Filterkernel g. Als Eingabe eines CNNs liegt in der Regel eine zwei- oder dreidimensionale Matrix vor (z.B. die Pixel eines Grauwert- oder Farbbildes). Dementsprechend sind auch die Neuronen in der Faltungsschicht (engl. Convolutional Layer) angeordnet. Die Aktivität jedes Neurons wird über eine diskrete Faltung berechnet. Diese ergibt sich für den zweidimensionalen Fall durch die endliche Summe

$$(f * g)(u, v) = \sum_k \sum_l f(u - k, v - l)g(k, l). \tag{3.39}$$

Dabei wird ein vergleichsweise kleiner Filterkernel (Faltungsmatrix) über die Eingabe bewegt und an jeder Stelle eine gewichtete Summe aus Eingang und Filterkernel berechnet.

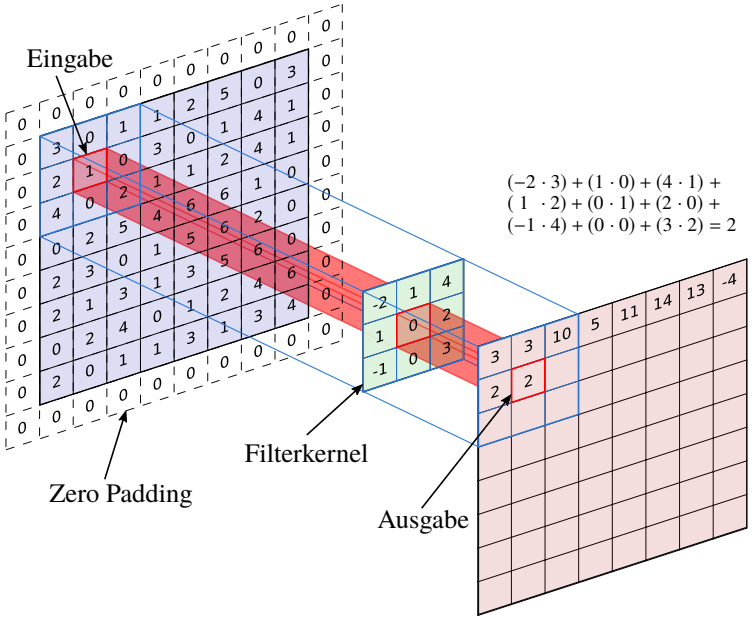


Abbildung 3.7: Visualisierung der Faltungsoperation mit bereits gespiegeltem Filterkernel. Durch das dargestellte Zero Padding behält die Ausgabe die gleiche Größe wie die Eingabe.

Dem biologischen Vorbild des rezeptiven Feldes folgend, reagiert ein Neuron in solch einer Schicht also nur auf Reize in einer lokalen Umgebung der vorherigen Schicht. Da derselbe Filterkernel für alle Neuronen einer Schicht verwendet wird, sind deren Gewichte identisch (geteilte Gewichte, engl. shared weights). Daraus folgt unmittelbar die Eigenschaft der Translationsinvarianz von CNNs. Sie besagt, dass das Ergebnis der Faltung für ein bestimmtes Merkmal unabhängig von translatorischen Verschiebungen des Merkmals identisch ist. Lediglich der Ort an dem das Merkmal in der Ausgabe der Schicht (auch *Feature Map* genannt) auftritt, ändert sich. Wie in Abbildung 3.7 zu sehen, reduziert sich je nach Größe des Filterkernels die Größe der Ausgabe gegenüber der Eingabe. Um dies auszugleichen kann das ebenfalls dargestellte *Padding* verwendet werden. Dadurch bleibt nicht nur die Ausgangsmatrix in der gleichen Größe, auch wird dafür gesorgt, dass die Informationen von den Rändern

des Eingangs ebenso gut repräsentiert werden, wie solche aus anderen Teilen. Ein weiterer Parameter für eine Faltungsschicht ist der sogenannte *Stride*. Hierbei handelt es sich um die Schrittweite des Filterkernels. Die Faltung wird also nicht bei jedem Pixel durchgeführt, was zu einer Verkleinerung der Ausgabe um ein Vielfaches führt.

Pooling Schicht

Eine weitere Möglichkeit die Dimension der Feature Map zugunsten einer schnelleren Berechnung zu reduzieren, stellen Pooling Schichten dar. Deren zugrunde liegende Idee ist es, dass benachbarte Pixel oftmals ähnliche Merkmale besitzen. Durch Pooling wird versucht solch redundanten Informationen zu filtern und eine kompaktere Repräsentation zu erhalten. Am weitesten verbreitet ist das in Abbildung 3.8 dargestellte *Max Pooling*. Dabei geht man davon aus, dass die wichtigste Information in einem Bild in dunkleren Pixeln bzw. im Falle von Feature Maps in Neuronen mit hoher Aktivität steckt [Ska18]. Dementsprechend wird in einer gewissen Nachbarschaft nur der Maximalwert in die Ausgabe übernommen. Pooling Schichten besitzen keine eigenen Gewichte und sind somit nicht unmittelbar am Lernprozess beteiligt.

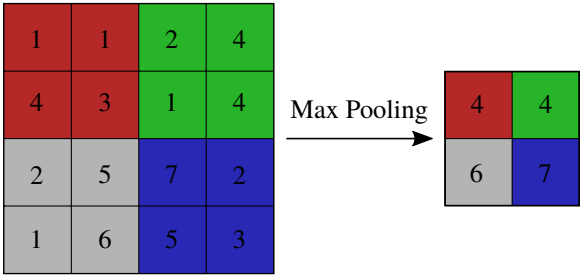


Abbildung 3.8: 2×2 Max-Pooling Operation mit Stride = 1. Die Breite und Höhe der Feature Map halbiert sich.

Batch-Normalisierung

Das Training eines neuronalen Netzes wird dadurch erschwert, dass sich die Verteilung der Eingangsdaten jeder Schicht ändert, sobald die Gewichte der

vorherigen Schicht geändert werden. Dieses, auch als interne Kovarianzverschiebung bezeichnete, Problem führt zu einer langsameren Konvergenz während des Trainings, da die Trainingsdaten für spätere Schichten nicht stabil sind. Des Weiteren führt die kontinuierliche Erhöhung oder Verringerung der Aktivierungsgradienten in aufeinanderfolgenden Schichten zu verschwindenden oder zu stark wachsenden Gradienten. Mithilfe der *Batch-Normalisierung* kann diesen beiden Problemen entgegengewirkt werden. Dabei werden zusätzliche Normalisierungsschichten hinzugefügt, die dafür sorgen, dass die Eingangsdaten jeder Schicht in etwa die gleiche Varianz besitzen. Um nicht den kompletten Trainingssatz nach jeder Parameteraktualisierung zu analysieren, wird die Normalisierung jeweils nur auf einem Mini-Batch angewendet. Dafür wird der Mittelwert $\mu_{\mathfrak{B}} = \frac{1}{m} \sum_{i=1}^{m} x_i$ und die Varianz $\sigma_{\mathfrak{B}}^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_{\mathfrak{B}})^2$ für den Mini-Batch $\mathfrak{B}$ berechnet. Die normalisierten Werte ergeben sich dann durch

$$\hat{x}_i = \frac{x_i - \mu_{\mathfrak{B}}}{\sqrt{\sigma_{\mathfrak{B}}^2 + \epsilon}}. \tag{3.40}$$

Für numerische Stabilität wird im Nenner eine willkürlich kleine Konstante ϵ hinzugefügt. Da eine einfache Normalisierung der Eingangsdaten jeder Schicht die Bedeutung der Schicht verändern kann, werden zusätzlich zwei Parameter γ und β eingeführt, welche die normalisierten Werte skalieren und verschieben können:

$$y_i = \gamma \hat{x}_i + \beta. \tag{3.41}$$

Diese Parameter werden zusammen mit den ursprünglichen Modellparametern gelernt und stellen die Darstellungsfähigkeit des Netzwerks sicher, sodass beispielsweise Sigmoid-Aktivierungen nicht auf ihren linearen Bereich begrenzt werden [IS15].

4 Schätzung der Bodenoberfläche

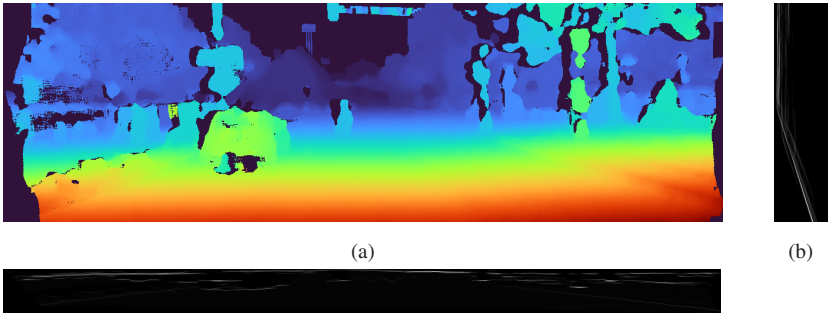
Die in den beiden folgenden Kapiteln genutzte Rasterkartendarstellung bietet eine orthografische Ansicht des Fahrzeugumfelds entlang der Bodenoberfläche. In dieser Darstellung sollen demnach einzig Punkte enthalten sein, welche nicht zur Bodenoberfläche gehören bzw. oberhalb dieser liegen. Durch die Verwendung semantischer Informationen kann bereits im Bild zwischen Boden- und anderen Bereichen unterschieden werden, bevor eine Projektion der einzelnen Punkte in die Rasterkarten erfolgt. Eine weitere Filterung anhand der Bodenoberfläche bietet hier also nur eine gewisse Robustheit gegen eine fehlerhafte Schätzung der Semantik. Nichtsdestotrotz liefert die Geometrie der Bodenoberfläche wertvolle Informationen um die Höhe und Position der detektierten Objekte korrekt abbilden zu können. Auch die dadurch zur Verfügung stehende Kenntnis über befahrbare Flächen trägt zu einem vollständigen Umfeldmodell bei und ist essentiell für das Szenenverstehen und die Verhaltensgenerierung automatisierter Fahrzeuge.

Die Schätzung des Fahrbahnverlaufs erfolgt auf Basis der so genannten V-Disparität, dessen Erstellung und Filterung in Abschnitt 4.1 genauer erläutert werden. Abschnitt 4.2 geht anschließend auf die Modellierung dieser Bodenoberfläche mithilfe von B-Spline-Kurven ein.

4.1 Erstellung und Filterung der V-Disparität

Der Verlauf der Bodenoberfläche in Längsrichtung wird mithilfe der V-Disparität $V(d, v)$ geschätzt, deren Eigenschaften von Labayrade et al. [LAT02] genauer beschrieben werden. Im Prinzip ist sie ein Histogramm der Disparitäten über jede Bildzeile eines Disparitätenbildes. Dafür wird die Häufigkeit jeder Disparität d pro Bildzeile v über den Bildzeilen aufgetragen und anhand des Helligkeitswerts des jeweiligen Pixels beschrieben. Das so entstehende Bild ist für das daneben zu sehende Disparitätenbild in Abbildung 4.1b dargestellt.

Speziell für weit entfernte Bereiche und demnach kleine Disparitäten macht es Sinn, das Histogramm nicht nur für ganzzahlige Disparitäten zu erstellen, sondern kleinere Klassenbreiten zu wählen um so später mehr Messpunkte für die Bodenoberflächenschätzung zur Verfügung zu haben.



(a) (b)

(c)

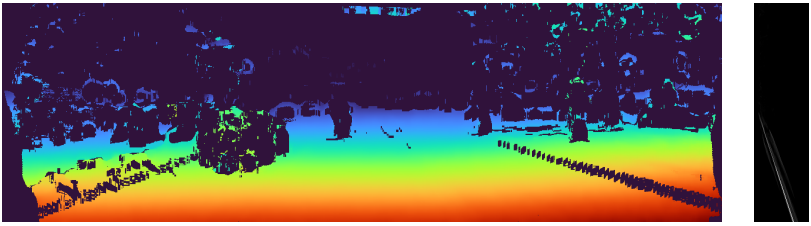
Abbildung 4.1: Veranschaulichung der Disparität (a), V-Disparität (b) und U-Disparität (c).

Unter der Voraussetzung, dass der Rollwinkel Φ zwischen Kamera und Boden null ist, bildet sich das Längsprofil der Bodenoberfläche in der V-Disparität als scharfe Linie ab, welche im Allgemeinen von links oben nach rechts unten verläuft. Ist der Boden im Bezug zur Kamera nicht mehr horizontal ausgerichtet, verwischt diese Linie und wird mit steigendem Rollwinkel zunehmend unschärfer. Zur Schätzung dieses Rollwinkels kann also die Schärfe der V-Disparität herangezogen werden. Dafür wird in einer Bildzeile das Verhältnis zwischen maximaler Intensität $\max_d\{V(\Phi, d, v)\}$ und dem Flächeninhalt unter der Intensitätsverteilung $\int V(\Phi, d, v)\mathrm{d}d$ über alle Disparitäten in Abhängigkeit des Rollwinkels betrachtet. Je größer dieser Quotient ist, desto schärfer ist der zur Bodenoberfläche gehörende Punkt in dieser Bildzeile. Der Rollwinkel zwischen Kamera und Boden entspricht dann demjenigen Winkel, für den die Summe dieses Schärfekriteriums über alle Bildzeilen maximal wird:

$$\Phi^* = \arg\max_{\Phi} \sum_{v} \frac{\max_d\{V(\Phi, d, v)\}}{\int V(\Phi, d, v)\mathrm{d}d}. \tag{4.1}$$

Ist das binokulare Kamerasystem zum Fahrzeug kalibriert, kann eine Schätzung des aktuellen Rollwinkels auch aus den Beschleunigungssensoren erfolgen.

Vertikale Flächen der im Bild befindlichen Objekte haben in jeder Bildzeile die gleiche Entfernung und bilden sich in der V-Disparität als senkrechte Linien ab, welche auf der Bodenoberflächenlinie enden. Bevor eine Schätzung der Bodenoberfläche erfolgt, sollen diese verlässlich heraus gefiltert werden. Dafür wird die in Abbildung 4.1c dargestellte U-Disparität verwendet. Diese lässt sich analog zur V-Disparität erzeugen. Zur Erstellung wird die Häufigkeit jeder Disparität pro Bildspalte bestimmt. Helle Punkte bedeuten demnach, dass diese Disparität in der entsprechenden Bildspalte häufig vorkommt und sind ein eindeutiges Merkmal für vertikale Objekte. Durch Festlegung eines geeigneten Grenzwertes können diese Punkte detektiert und verworfen werden, da sie offensichtlich nicht zur Bodenoberfläche gehören. Das auf diese Weise entstehende Disparitätenbild und die dazugehörige V-Disparität sind in Abbildung 4.2 wiedergegeben. Es ist deutlich zu erkennen, dass alle senkrechten Linien aus der V-Disparität heraus gefiltert wurden.



(a) (b)

Abbildung 4.2: Veranschaulichung der Filterung vertikaler Strukturen in der Disparität (a) und V-Disparität (b).

Dennoch sind weiterhin einige Punkte zum Beispiel an Objekträndern vorhanden, welche die Schätzung der Bodenoberfläche verfälschen würden. Da die Bodenoberfläche für eine feste Disparität den tiefsten Punkten im V-Disparitätenbild und für eine feste Bildzeile den Punkten mit der größten Entfernung entsprechen muss, werden zwei zusätzliche Gewichtungen angewendet um den Einfluss dieser Punkte zu reduzieren [Joo11]. Für die Gewichtung $w_v(d, v)$ wird das Verhältnis der überstrichenen Fläche zur Gesamtfläche entlang einer Spalte betrachtet:

$$w_v(d,v) = \frac{\int_0^v V(d,v)\mathrm{d}v}{\int_0^{v_{\max}} V(d,v)\mathrm{d}v}. \tag{4.2}$$

Die Gewichtung $w_d(d,v)$ lässt sich auf die gleiche Art beschreiben. Hier wird entlang einer Bildzeile ausgehend vom rechten Rand das Verhältnis von überstrichener Fläche zur Gesamtfläche gebildet:

$$w_d(d,v) = \frac{\int_d^{d_{\max}} V(d,v)\mathrm{d}d}{\int_0^{d_{\max}} V(d,v)\mathrm{d}d}. \tag{4.3}$$

Werden die Gewichte in dieser Form direkt auf die V-Disparität angewendet, sind sie sehr anfällig für Punkte welche sich unterhalb der zur relevanten Bodenoberfläche gehörenden Linie befinden, da diese das maximale Gewicht erhalten und so sogar noch stärker in die Schätzung des Fahrbahnprofils eingehen. Dies kann auftreten, wenn sich die Nachbarfahrbahn wie in Abbildung 4.2 etwas tiefer befindet oder zum Beispiel auf einer Landstraße neben der Fahrbahn Straßengräben vorliegen. Um dies zu verhindern werden beide Gewichtsverteilungen so transformiert, dass ihr Maximum nicht mehr bei 100 % sondern bei 95 % liegt. Bei Überschreitung von 95 % des Flächeninhalts werden die Gewichtungen linear gemäß Abbildung 4.3a zurückgenommen [Joo11]. Da die Bodenoberfläche in der V-Disparität für eine feste Bildzeile bzw. -spalte ein deutliches Maximum darstellt, liegt das 95 %-Kriterium selbst bei einigen Ausreißern nahe des tatsächlichen Maximums, während 100 % der Fläche erst sehr viel weiter entfernt erreicht werden würde. Abbildung 4.3b und 4.3c zeigen die V-Disparität vor und nach Anwendung dieser Gewichtungen.

4.2 Modellierung der Bodenoberfläche

Aus den auf diese Weise gefilterten V-Disparitäten wird mithilfe der in Abschnitt 3.4 vorgestellten B-Splines die Bodenoberfläche geschätzt. Durch die Verwendung von B-Splines können im Gegensatz zu stückweise planaren [LAT02], quadratischen [ONMT07] oder klothoidenbasierten [NDF+04] Modellen auch Oberflächenprofile kontinuierlich beschrieben werden, welche

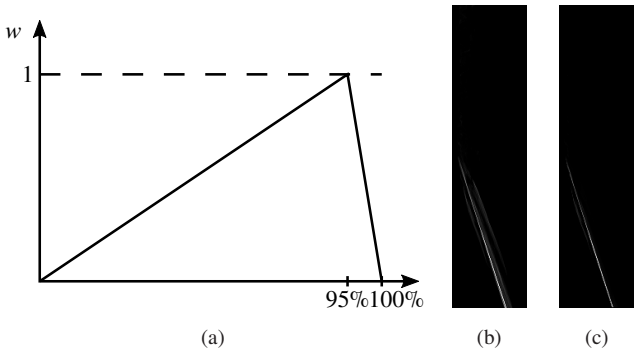


Abbildung 4.3: Verwendete Gewichtsverteilung (a), Ursprüngliche V-Disparität (b) und V-Disparität nach Anwendung der Gewichtungen (c).

mehr als eine Änderung in der Steigung aufweisen oder sogar wellenförmig verlaufen. Um möglichst geringe Laufzeiten zu erzielen, werden uniforme B-Splines benutzt. Dadurch bleibt die Basisfunktion für jedes Segment gleich und kann somit im Voraus berechnet werden. Der Spline wird dann nur noch durch den Kontrollpunktvektor **c** verändert. Da die Auflösung des V-Disparitätenbildes mit steigender Entfernung sinkt, würde eine Schätzung des Verlaufs der Bodenoberfläche mit uniformen B-Splines direkt in der V-Disparität dazu führen, dass der Nahbereich durch mehrere Splinesegmente repräsentiert wird, wohingegen weiter entfernte Oberflächen in einem großen Bereich einzig durch ein Segment modelliert werden. Letzteres würde dazu führen, dass bei mehreren Steigungsänderungen in diesem Bereich ein Spline höherer Ordnung nötig wäre. Um dies zu umgehen wird die Bodenoberfläche, wie bereits durch Wedel et al. [WBR+09] vorgeschlagen, direkt in Weltkoordinaten geschätzt. Dabei sei an dieser Stelle nochmal darauf hingewiesen, dass in dieser Arbeit, wie bereits in Abschnitt 3.2 erwähnt, als Weltkoordinatensystem das Kamerakoordinatensystem der linken Kamera verwendet wird. Der einzige Nachteil eines solchen Ansatzes ist, dass die nichtlineare Fehlerfortpflanzung der Disparitätsmessungen berücksichtigt werden muss.

Im Falle einer Ebene wird jeder Punkt auf dem Boden in Weltkoordinaten mit der Kamerahöhe h_C und dem Nickwinkel der Kamera θ_C durch

$$y_{\mathrm{W}}(z_{\mathrm{W}}) = h_{\mathrm{C}} + \tan(\theta_{\mathrm{C}}) z_{\mathrm{W}} \tag{4.4}$$

beschrieben. Durch die Annahme der Planarität in einem kleinen Bereich um das eigene Fahrzeug, kann dieser Zusammenhang mithilfe der V-Disparitätswerte für dementsprechend große Disparitäten dafür genutzt werden, den Nickwinkel der Kamera in Relation zur Bodenoberfläche zu schätzen. Durch Einsetzen der Rekonstruktionsfunktion aus Gleichung 3.11 in Gleichung 4.4 ergibt sich der Nickwinkel

$$\theta_{\mathrm{C}}(d, v) = \arctan\left(\frac{v - c_{\mathrm{v}}}{f} - \frac{h_{\mathrm{C}} d}{bf}\right) \tag{4.5}$$

in Abhängigkeit der Bildzeile und Disparität. Die Position des Maximums in der untersten Bildzeile der V-Disparität gibt an, welche Disparität die zum Boden gehörenden Punkte besitzen, die am nächsten zur Kamera liegen. Durch die in Abschnitt 4.1 vorgestellte Gewichtung wird die unterste Zeile allerdings zu Null gesetzt, weswegen die vorletzte Bildzeile verwendet werden muss. Beginnend bei der auf diese Weise gefundenen Disparität bis hin zu Disparitätswerten welche Punkten einige wenige Meter weiter entfernt entsprechen, wird für jede Spalte das Maximum der V-Disparität bestimmt und damit der Nickwinkel berechnet. Aus dem Mittelwert ergibt sich so die Schätzung für den Nickwinkel der Kamera in Relation zur Bodenoberfläche.

Ebenfalls ab dieser maximalen Disparität bis zu einer minimalen, welche sich ergibt, sobald das Spaltenmaximum einen bestimmten Schwellwert nicht mehr überschreitet, gehen die beiden größten Werte jeder Spalte in der V-Disparität in die Bodenoberflächenschätzung ein. Durch die Verwendung von zwei Messwerten pro Spalte und ihrer Gewichtung wird erreicht, dass eindeutige Maxima stärker in die Schätzung eingehen und in Spalten in denen das Maximum zwischen zwei Zeilen liegt implizit eine Interpolation erfolgt. Für jeden dieser Werte wird die Entfernung

$$z_m = \frac{fb}{d_m} \tag{4.6}$$

aus Brennweite, Basisbreite und Disparität bestimmt. Die Höhe

$$y_m = \frac{b(v_m - c_\mathrm{v})}{f} - \tan(\theta_\mathrm{C}) z_m \tag{4.7}$$

ergibt sich aus Basisbreite, der jeweiligen Zeile des Maximums v_m, Bildhauptpunkt, Brennweite und der entfernungsabhängigen Korrektur des Nickwinkels $\tan(\theta_\mathrm{C})z_m$. Außerdem wird jeweils eine Standardabweichung σ_m, welche die Unsicherheit der Stereo-Tiefenschätzung beschreibt, berechnet. Dazu wird die in der Kalibrierung und beim Matching auftretende Unsicherheit mit der Jakobi-Matrix der Rekonstruktionsfunktion in den 3D-Raum propagiert. Das genaue Vorgehen dafür wird in Abschnitt 5.3 erläutert. Als zusätzlicher Messwert wird die Randbedingung verwendet, dass das eigene Fahrzeug den Boden berühren und sich dieser demnach bei $s(0) = h_\mathrm{C}$ befinden muss.

Nun soll die B-Spline-Kurve $s(z)$ bestimmt werden, welche die relative Höhe der Bodenoberfläche in Abhängigkeit der Entfernung von der Kamera beschreibt. Als Kompromiss zwischen Laufzeit und Genauigkeit der Approximation werden kubische Splines mit äquidistanten Knotenpunkten verwendet. Das Ziel ist also, den optimalen Kontrollpunktvektor $\hat{\boldsymbol{c}}$ zu finden, sodass $s(z)$ am besten zu den Messwerten y passt. Die Güte dieser Anpassung wird durch die Summe der quadratischen Abweichungen von den Messwerten ausgedrückt:

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c}} \left\{ \sum_m \frac{V(d,v)}{\sigma_m^2} (s(z_m) - y_m)^2 \right\}. \tag{4.8}$$

Als Gewichtung geht neben der Unsicherheit der Stereo-Tiefenschätzung der jeweilige Wert der V-Disparität ein. Es ergibt sich ein Least Squares Problem, welches den optimalen Kontrollpunktvektor mit den in Abschnitt 3.5 beschriebenen Techniken effizient bestimmt:

$$\hat{\boldsymbol{c}} = \arg\min_{\boldsymbol{c}} \left\| \begin{bmatrix} \frac{V(d,v)}{\sigma_0^2} \mathbf{B}_\mathbf{d}(z_0)^T \\ \vdots \\ \frac{1}{\sigma_m^2} \mathbf{B}_\mathbf{d}(z_m)^T \end{bmatrix} \mathbf{c} - \begin{bmatrix} y_0 \\ \vdots \\ y_m \end{bmatrix} \right\|^2. \tag{4.9}$$

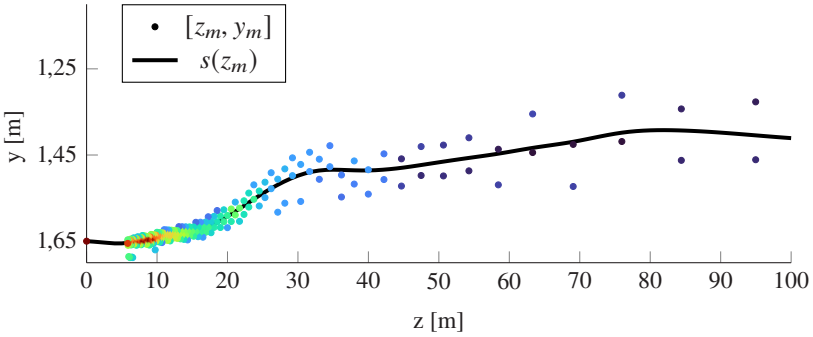
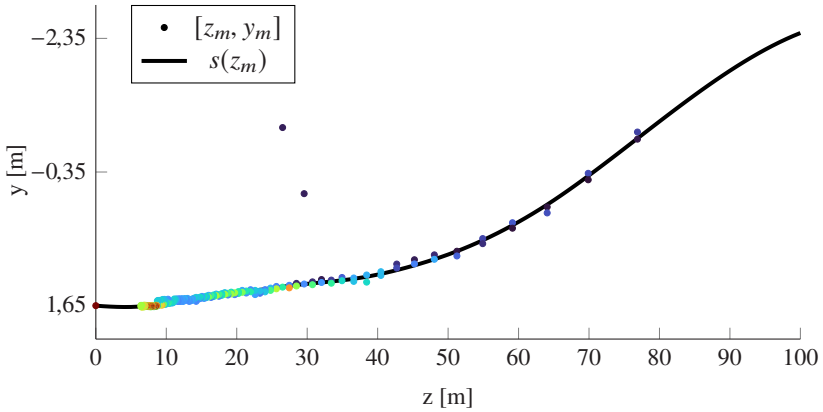
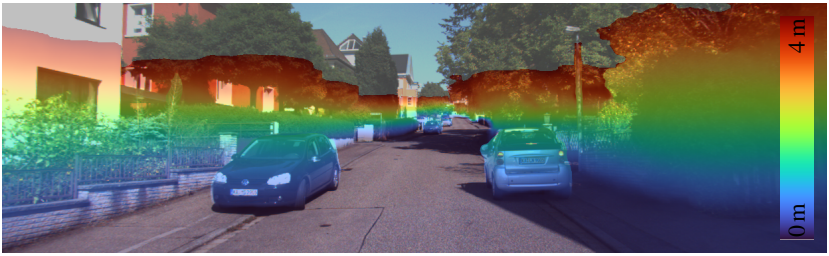


Abbildung 4.4: Messpunkte mit Gewichtung und resultierende B-Spline-Kurve für das Disparitätsbild aus Abbildung 4.1a bzw. die V-Disparität aus Abbildung 4.3c.

Die auf diese Weise bestimmte B-Spline-Kurve ist für das Beispiel aus Abbildung 4.1 gemeinsam mit den verwendeten Messwerten in Abbildung 4.4 dargestellt. Die farbliche Kodierung der Messwerte beschreibt die jeweilige Gewichtung von dunkelrot für hohe Werte hin zu dunkelblau für niedrige. Alle in diesem Kapitel verwendeten Beispiele stammen aus dem KITTI-Datensatz, welcher mit einer Kamerahöhe von $h_C = 1{,}65\,\text{m}$ aufgezeichnet wurde [GLSU13]. Es wird deutlich, wie durch die Verwendung der B-Splines der Verlauf des Fahrbahnprofils trotz veränderlicher Steigungen kontinuierlich beschrieben werden kann. Dennoch bewegt sich der Höhenverlauf der Bodenoberfläche in diesem Beispiel in einem Bereich, welcher selbst mit der Annahme einer planaren Fahrbahn in einem geringen Fehler resultieren würde. Deshalb ist in Abbildung 4.5a die B-Spline-Kurve für ein Fahrbahnverlauf mit ansteigendem Profil zu sehen. Um die Genauigkeit dieser Approximation besser einschätzen zu können ist in Abbildung 4.5b zusätzlich die dazugehörige Höhenkarte über das linke Kamerabild gelegt. Diese ergibt sich aus der Differenz der zum jeweiligen Bildpunkt gehörenden y-Koordinate und $s(z)$ an dieser Stelle. Bereiche, welche sich auf dem geschätzten Fahrbahnniveau befinden, sind farblich nicht überlagert, Punkte die darunter liegen sind dunkelblau dargestellt, Punkte darüber verlaufen von blau bis zu einer maximal dargestellten Höhe von vier Metern über dem Niveau der Fahrbahn in rot. Besonders an den Rändern der Fahrbahn zeigt sich die Genauigkeit der Bodenoberflächenschätzung, da die gleichen Farben über den gesamten sichtbaren Bereich der Straße etwa in einer Höhe relativ zur Fahrbahn verlaufen.



(a) Messpunkte mit Gewichtung und resultierende B-Spline-Kurve.



(b) Linkes Kamerabild mit überlagertem farblichem Verlauf der Höhe über dem geschätzten Fahrbahnniveau von blau bis zu einer maximal dargestellten Höhe von 4 m in rot.

Abbildung 4.5: Beispiel für einen nicht-ebenen Fahrbahnverlauf.

5 Rasterkartenbasierte Objektdetektion

In diesem Kapitel wird ein Ansatz zur binokularen Objektdetektion vorgestellt, welcher auf unserer Veröffentlichung [KSS19] basiert und diese um eine genauere Modellierung der Belegtheit der Rasterkartenzellen und eine explizite Behandlung von unbeobachteten Zellen erweitert. Zuerst wird in Abschnitt 5.1 auf die einzelnen Verarbeitungsschritte dieses Ansatzes und speziell die verwendeten Eingangsdaten eingegangen. Abschnitt 5.2 beschreibt, auf welche Weise diese Daten verwendet werden um Objektcluster im Bild zu finden. Die 3D Bounding Box Schätzung aus diesen Clustern bildet in Abschnitt 5.3 den Abschluss dieses Kapitels.

5.1 Übersicht

Die einzelnen Schritte des in diesem Kapitel vorgestellten Ansatzes zur binokularen Objektdetektion sind in Abbildung 5.1 zusammengefasst. Als Eingabe wird ein Stereobildpaar verwendet, welches entweder in Graustufen oder in Farbe sein kann, wobei von Ersterem eher die Disparitätenschätzung und von Zweiterem eher die Semantik profitiert. Eines der Bilder wird verwendet, um ebendiese pixelweise semantische Information zu erhalten. Dafür wurde sich für ein CNN entschieden, welches in Abschnitt 5.1.1 genauer beschrieben wird und ebenfalls 2D Bounding Boxen liefern kann. Letztere verändern die Ergebnisse dieses Ansatzes nur geringfügig, sorgen aber für eine bessere Parallelisierung des Clustering und damit für eine geringere Laufzeit. Parallel zur Schätzung der semantischen Informationen wird ein in Abschnitt 5.1.2 vorgestellter *Block Matching*-Algorithmus verwendet, um die Disparitäten zwischen linkem und rechtem Bild zu berechnen. Anschließend wird ein Clustering der Disparitäten mittels Connected Component Labeling (CCL) durchgeführt, um Objektvorschläge zu erhalten. Ein niedrigerer Schwellenwert wird verwendet,

wenn Pixel zur gleichen semantischen Klasse gehören und eine höhere Konfidenz der Disparität aufweisen. Eine genauere Erläuterung ist in Abschnitt 5.2 zu finden. Für jedes im Bildbereich gefundene Cluster wird die Position aller dazugehörigen Punkte in Weltkoordinaten errechnet und in eine lokale Rasterkarte in der xz-Ebene mit zusätzlicher Höheninformation, welche die y-Koordinate repräsentiert, projiziert. Danach werden die Orientierung und Dimension des Objekts unter Verwendung klassenspezifischer Objektgrößen optimiert und so die 3D Bounding Box geschätzt. Die Aufstellung dieser Optimierungsfunktion ist in Abschnitt 5.3 zu finden.

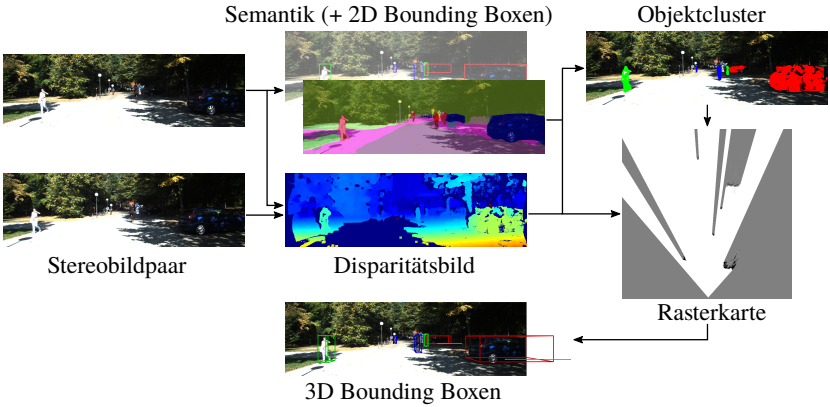


Abbildung 5.1: Übersicht über die einzelnen Schritte der Rasterkartenbasierten Objektdetektion.

5.1.1 Semantische Klassifizierung

Für die pixelweise semantische Segmentierung und die 2D-Objekterkennung wird ein CNN verwendet. Das Grundgerüst stellt ein Encoder dar, welcher auf dem ResNet-38 [WSVDH19] basiert. Da der Ausgang in zwei Zweige gespeist wird, löst dieses Netz die beiden Aufgaben gleichzeitig. Der erste Zweig dekodiert die Ausgabe des Encoders in eine semantische Segmentierungskarte mit der ursprünglichen Bildauflösung. Der zweite führt die Bounding Box Detektion und Regression durch. Hierfür wird ein Ansatz basierend auf Anchor Boxen verwendet, der Ideen von SSD [LAE+16] und RetinaNet [LGG+17] aufgreift. Die detaillierte Architektur dieses Netzwerks ist in [Sal20] beschrieben.

5.1.2 Disparitäts- und Konfidenzschätzung

Für die Disparitätsschätzung wird ein lokaler Stereo-Matcher basierend auf [RS14] verwendet. Dieser Ansatz verbessert die Fehleranfälligkeit anderer fensterbasierter Algorithmen in Bereichen mit großen räumlichen Unterschieden durch beliebig ausrichtbare Scharen von Ebenen. Durch die Verwendung der in Abschnitt 3.3 erläuterten Census-Transformation ist er robust gegen Helligkeitsschwankungen und liefert bessere Ergebnisse an den für die exakte Objektdetektion wichtigen Objektgrenzen, an denen sowohl Pixelintensitäten des Objekts als auch der Szene und somit eine multimodale Intensitätsverteilung vorliegen. Die durch die Census-Transformation resultierende Kodierung der relativen Anordnung der Intensitäten innerhalb eines Matching-Fensters als Bitfolge und das daraus folgende Matching mithilfe der Hamming-Distanz, führt außerdem zu einer guten Parallelisierbarkeit und dementsprechend geringer Laufzeit.

Um eine zwar in dieser Arbeit nicht betrachtete aber durchaus denkbare Fusion der erkannten Objekte mit Detektionen aus anderen Sensoren zu ermöglichen, ist es notwendig, die Genauigkeit der einzelnen Detektionen einschätzen zu können. Dafür wird aus den Kosten, welche beim Matchen der jeweiligen Bildbereiche entstehen, ein Konfidenzwert zu jeder Disparität berechnet, welcher ebenfalls für das Clustering verwendet wird. Hu und Mordohai [HM12] evaluieren 17 verschiedene Konfidenzmaße und kommen zu dem Ergebnis, dass die Left Right Difference (LRD) die beste Metrik für binokulare Systeme bietet, weshalb diese auch hier genutzt wird. LRD hängt sowohl von dem Unterschied des kleinsten und zweitkleinsten Wert der Matchingkosten im linken Bild c_1 und c_2 ab, als auch von der Konsistenz der minimalen Kosten des linken und rechten Disparitätenbildes und ergibt sich zu

$$C_{\mathrm{LRD}}(u, v) = \frac{c_2 - c_1}{\left| c_1 - \min_{d_\mathrm{R}} c_\mathrm{R}(u - d_\mathrm{L}, v, d_\mathrm{R}) \right|}. \tag{5.1}$$

Dabei bezeichnen d_L und d_R die jeweilige Disparität des linken bzw. rechten Bildes und c_R die Kosten des rechten Disparitätenbildes. Die Idee hinter dieser Metrik ist, dass wirklich übereinstimmende Matching-Fenster zu ähnlichen Kostenwerten und damit zu kleinen Werten des Nenners führen sollten. Sie bietet eine Absicherung gegen zwei verschiedene Fehlermöglichkeiten. Ist die

Differenz $c_2 - c_1$ groß, aber nicht die richtigen korrespondierenden Pixel gefunden, ist der Nenner groß und somit die Konfidenz gering. Im Falle einer kleinen Differenz, ist der Match wahrscheinlich mehrdeutig und die Größe des Nenners gibt an, ob es sich um zwei ähnliche Pixel handelt.

5.2 Clustering

Die berechneten Disparitäten werden nun zu Clustern mit ähnlichen Werten zusammengefasst. Zu diesem Zweck wird ein ungerichteter Graph aufgebaut, in dem jedes Pixel als Knoten betrachtet wird und benachbarte Knoten i und j, die innerhalb eines semantik- und konfidenzabhängigen Schwellenwertes liegen, durch Kanten verbunden werden, falls

$$\frac{1}{d_i} - \frac{1}{d_j} < \frac{t_{\text{ass}} C_i C_j}{kbf} \quad \text{mit} \quad k = \begin{cases} 1 & \text{, gleiche semantische Klasse} \\ 5 & \text{, sonst.} \end{cases} \tag{5.2}$$

Mit der Basisbreite b und der Brennweite f ergibt sich durch Umstellen die metrische Differenz der Entfernung der beiden Pixel in z-Richtung, welche kleiner als ein Schwellwert sein soll, der von t_{ass}, k und den Konfidenzen der beiden Pixel C_i und C_j abhängt. Dabei ist t_{ass} ein anpassbarer Assoziationsschwellwert, welcher dementsprechend angibt bis zu welchem Abstand in Metern zwei Pixel der gleichen semantischen Klasse zu einem gemeinsamen Cluster gehören sollen.

Aus diesem Graphen werden die verbundenen Komponenten mit einer Tiefensuche (engl. depth-first search) berechnet. Jede verbundene Komponente ist eine Menge von Eckpunkten in diesem Graphen, welche alle untereinander erreichbar sind. Dieser Schritt wird ebenfalls parallelisiert, indem das Bild in einzelne Blöcke unterteilt und dann das CCL innerhalb jedes Blocks parallel durchgeführt wird. Anschließend müssen Cluster, die über die Grenzen der Blöcke hinausgehen, kombiniert werden. Dazu werden alle Blockgrenzen nacheinander auf äquivalente Label überprüft und anschließend die Beschriftungen in jedem Block parallel aktualisiert. Die Tatsache, dass der zweite Schritt in einem einzigen Thread ausgeführt werden muss, führt jedoch vor

und nach diesem Schritt zu einer Synchronisationsbarriere und damit zu Laufzeitverlusten. Wie bereits oben erwähnt, können diese durch die Verwendung von 2D Bounding Boxen als zusätzliche Eingabe vermieden werden. In diesem Fall wird das CCL parallel in allen Bounding Boxen durchgeführt. Dann werden nur die wenigen benachbarten oder überlappenden Bounding Boxen nach Clustern durchsucht, die kombiniert werden müssen. Der letztgenannte Schritt wird nur verwendet, um robust gegenüber falschen 2D-Box-Vorschlägen zu sein. Darüber hinaus wird in allen Bildbereichen ohne Bounding Boxen ein grobes Clustering durchgeführt, um eine zu starke Abhängigkeit von den 2D-Boxen zu verhindern und auch Objekte zu erkennen, welche vom 2D-Detektor nicht erkannt wurden.

5.3 3D Bounding Box Schätzung

Da durch das Clustering schon bekannt ist, welche Punkte zu welchem Objekt gehören, werden alle Punkte jedes Clusters nicht in eine gemeinsame Rasterkarte sondern jeweils in eine eigene lokale Rasterkarte mit der Auflösung $[r_\mathrm{x}, r_\mathrm{z}]$ in der x-z-Ebene des Kamerakoordinatensystems projiziert:

$$x = \frac{(u - c_\mathrm{u}) \cdot b}{d \cdot r_\mathrm{x}}, \tag{5.3}$$

$$z = \frac{f \cdot b}{d \cdot r_\mathrm{z}}. \tag{5.4}$$

Die Belegtheit jeder Rasterzelle ergibt sich durch die Anzahl an Punkten n_p, welche in diese Zelle projiziert wurden zu

$$p_\mathrm{o}(x, z) = \min\left\{1, \frac{\log(n_\mathrm{p} + 1)}{\log(n_\mathrm{o})}\right\}. \tag{5.5}$$

n_o definiert, ab wie vielen Detektionen in einer Zelle diese als belegt gilt und ergibt sich dementsprechend aus der Auflösung des Eingangsbildes (der Disparitätenkarte) und der Rasterkarte. Unbeobachteten Zellen, das heißt Zellen, die sich hinter belegten Zellen in der Richtung des Beobachtungswinkels oder

außerhalb des aktuellen Sichtfeldes befinden, wird ein Wert von $p_o(x, z) = 0{,}5$ zugewiesen. Zusätzlich wird pro Rasterzelle noch eine maximale Höhe, durch alle

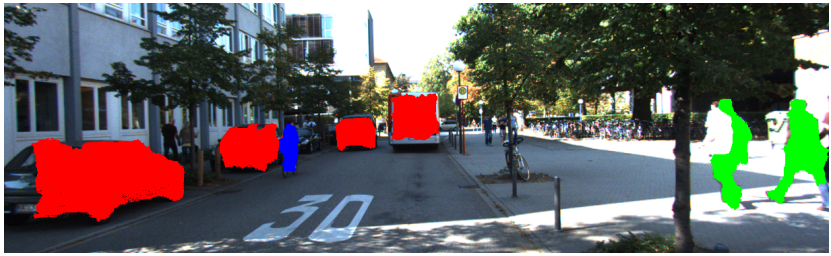
$$y_i = \frac{(v_i - c_v) \cdot b}{d(u_i, v_i)} \tag{5.6}$$

die in dieser Zelle liegen, berechnet. Die minimale Höhe der jeweiligen Zelle ergibt sich entweder aus der in Kapitel 4 vorgestellten Schätzung der Bodenoberfläche oder ebenfalls aus allen in dieser Zelle liegenden Punkten. In Abbildung 5.2 sind alle in einem Beispielbild bestimmten Cluster gemeinsam mit einer Rasterkarte zu sehen, welche durch die Kumulation der auf diese Weise berechneten Rasterkarten aller Cluster entstanden ist.

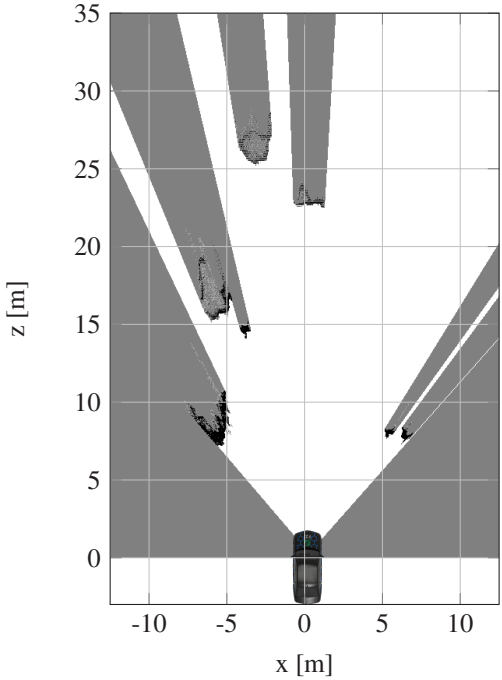
Da auch die Objektklasse jedes Clusters durch die semantische Information bekannt ist, können die typischen mittleren Dimensionen dieses Objekttyps $[\mu_h, \mu_w, \mu_l]$ und deren Standardabweichungen $[\sigma_h, \sigma_w, \sigma_l]$ verwendet werden um auch die genauen Ausmaße von Objekten schätzen zu können, welche nur von einer Seite sichtbar sind, wie zum Beispiel vorausfahrende Fahrzeuge.

Außerdem soll die Unsicherheit der Stereo-Tiefenschätzung in der Optimierungsfunktion Beachtung finden. Die korrespondierenden Punkte $[u, v]^T$ im linken Bild und $[u_R, v_R]^T$ im rechten Bild werden dafür als normalverteilte Vektoren mit den Kovarianzmatrizen $\mathbf{\Sigma}_L$ und $\mathbf{\Sigma}_R$ angenommen. Die Unsicherheiten in u berücksichtigen Fehler in der Kalibrierung und beim Matching, Abweichungen in v treten ausschließlich durch Ersteres auf.

Im Falle isotroper Kovarianzen liegen die Punkte mit einer Wahrscheinlichkeit von η innerhalb der in Abbildung 5.3a in den beiden Bildebenen dargestellten Kreise. Der durch diese beiden Punkte rekonstruierte 3D-Punkt liegt dann mit einer Wahrscheinlichkeit von η^2 in dem ebenfalls abgebildeten Schnittbereich, welcher sich durch die zwei elliptischen Kegel, aufgespannt von den Kreisen und den optischen Zentren der Kamera, ergibt. Ebenfalls deutlich wird, dass sich das Aussehen des Schnittbereiches in Abhängigkeit der wahren 3D-Position ändert, es handelt sich demnach um einen heteroskedastischen Fehler.



(a) Im Bild bestimmte Objektcluster mit farblicher Kodierung der durch die Semantik ermittelten Objektklassen. Dabei sind Autos in rot, Radfahrer in blau und Fußgänger in grün dargestellt.



(b) Aus Projektion der Clusterpunkte entstehende Rasterkarte. Der Grauwert beschreibt die Belegheit jeder Zelle und verläuft von weiß für freie Zellen über grau für unbeobachtete bis hin zu Zellen mit einer Belegtheit von $p_o(x, z) = 1$ in schwarz.

Abbildung 5.2: Objektcluster und kumulierte Rasterkarte aller in der Szene gefundenen Objekte.

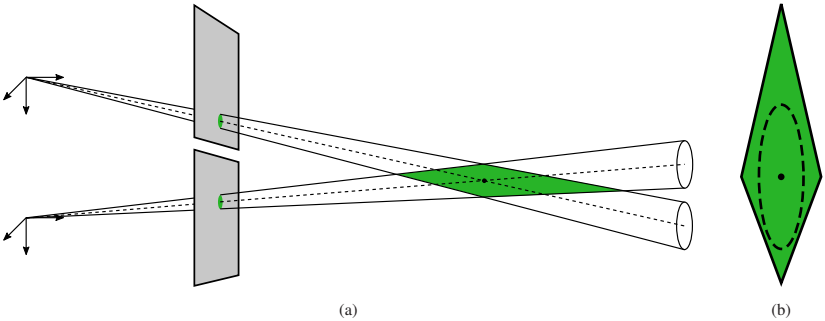


Abbildung 5.3: Veranschaulichung der Unsicherheit der Stereo-Tiefenschätzung (a) in Draufsicht und (b) verwendete Approximation dieser Unsicherheit.

Deshalb werden mithilfe der Jakobi-Matrix der Rekonstruktionsfunktion (Gleichung 3.11)

$$\mathbf{J} = \begin{bmatrix} \frac{-ub}{d^2} + \frac{b}{d} & 0 & \frac{ub}{d^2} & 0 \\ \frac{-vb}{d^2} & \frac{b}{d} & \frac{vb}{d^2} & 0 \\ \frac{-fb}{d^2} & 0 & \frac{fb}{d^2} & 0 \end{bmatrix} \tag{5.7}$$

die Unsicherheiten der beiden Bilder in den 3D-Raum [MS87] propagiert:

$$\begin{bmatrix} \sigma_x^2 \\ \sigma_y^2 \\ \sigma_z^2 \end{bmatrix} = \operatorname{diag}\left(\mathbf{J} \begin{bmatrix} \mathbf{\Sigma}_L & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_R \end{bmatrix} \mathbf{J}^T \right). \tag{5.8}$$

Dadurch ergibt sich eine dreidimensionale Gauß-Verteilung, welche den Fehler in den rekonstruierten 3D-Koordinaten abschätzt. Da die Triangulation eine nichtlineare Operation ist, ist die wahre Verteilung jedoch nicht gaußförmig. Dies ist auch in Abbildung 5.3b dargelegt. Die Ellipse repräsentiert das beschriebene Fehlermodell, das Viereck die wahre Verteilung der Unsicherheit aus der Draufsicht aus Abbildung 5.3a. Der Schwachpunkt der Gaußschen Näherung liegt darin, dass die längeren Schweife der wahren Fehlerverteilung nicht dargestellt werden können. Die wahre Verteilung ist schief, während

Normalverteilungen symmetrisch sind. Die Schiefe ist nicht signifikant, wenn die Punkte nahe liegen, wird aber ausgeprägter, je weiter die Punkte entfernt sind. Aus Gründen der Rechenzeit und da keine extrem weit entfernten Punkte betrachtet werden, genügt die beschriebene Approximation.

Zur Schätzung der Boxparameter in der Rasterkarte kann so ein Least Squares Problem aufgestellt werden, welches mit den in Abschnitt 3.5 beschriebenen Techniken effizient gelöst werden kann:

$$\begin{aligned}\hat{\boldsymbol{\mathcal{B}}}(x, z, w, l, \theta) = \underset{x,z,w,l,\theta}{\arg\min} \frac{1}{N} \sum_{i}^{N} p_{\mathrm{o}}(x_i, z_i)\, d^2(\boldsymbol{\mathcal{B}}, x_i, z_i, \sigma_{\mathrm{x}}, \sigma_{\mathrm{z}}) \\ + \lambda_w \left(\frac{w - \mu_{\mathrm{w}}}{\sigma_{\mathrm{w}}}\right)^2 + \lambda_l \left(\frac{l - \mu_{\mathrm{l}}}{\sigma_{\mathrm{l}}}\right)^2 .\end{aligned} \tag{5.9}$$

Dabei bezeichnet d den mit den Standardabweichungen der Position σ_{x} und σ_{z} gewichteten euklidischen Abstand der jeweiligen Zellmittelpunkte $[x_i, z_i]$ zur Box $\boldsymbol{\mathcal{B}}(x, z, w, l, \theta)$. Für alle Zellmittelpunkte, welche innerhalb der aktuellen Boxschätzung liegen, gilt $d = 0$. Jedes dieser Zell-Residuen wird außerdem durch die Belegtheit der Zelle gewichtet. Die Parameter λ_w und λ_l sorgen für eine passende Balancierung der quadratischen Abstände zu den typischen Objektdimensionen, welche ebenfalls mit in die Optimierungsfunktion eingehen. Als Initialisierung des Box-Mittelpunktes $[x, z]$ wird der Schwerpunkt aller zu diesem Objekt gehörenden Punkte verwendet, w und l werden mit deren Mittelwert und θ mit $\frac{\pi}{2}$ initialisiert.

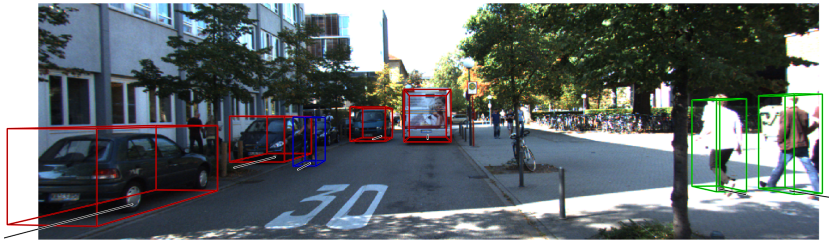
Anschließend wird die optimale Position in y und die Höhe der 3D-Box h mithilfe der minimalen und maximalen Höhen aller zum Objekt gehörenden und beobachteten Zellen und der typischen Objekthöhe bestimmt:

$$\begin{bmatrix} \hat{y} \\ h \end{bmatrix} = \underset{y,h}{\arg\min} \; \frac{1}{N_{\mathrm{O}}} \sum_{i}^{N_{\mathrm{O}}} p_{\mathrm{o}}(x_i, z_i) \left(\frac{y_{i,\mathrm{min/max}} - y}{\sigma_{\mathrm{y}}}\right)^2 + \lambda_h \left(\frac{h - \mu_{\mathrm{h}}}{\sigma_{\mathrm{h}}}\right)^2 . \tag{5.10}$$

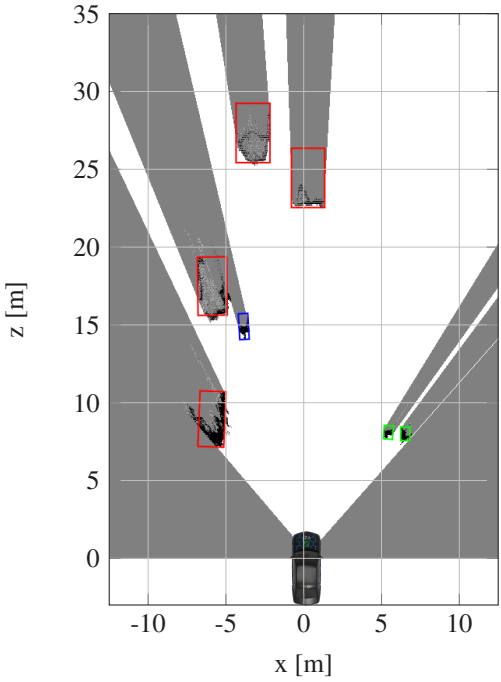
Der Residuenvektor besteht aus der mit der Standardabweichung normierten Differenz der minimalen bzw. maximalen Höhe der jeweiligen Zelle y_i und dem geschätzten Box-Mittelpunkt y. Analog zu Gleichung 5.9 sorgt λ_h für eine passende Gewichtung der quadratischen Abstände im Vergleich zur typischen Objekthöhe.

Die auf diese Weise geschätzten Boxen in der Rasterkarte, sowie die ins linke Kamerabild projizierten 3D Bounding Boxen sind in Abbildung 5.4 dargestellt.

Aus allen Konfidenzwerten pro Pixel und der Größe des Objektclusters im Bildbereich wird ein Konfidenzwert pro Objekt berechnet. Da die Fläche eines Objekts im Bild quadratisch mit seiner Entfernung in Weltkoordinaten abnimmt, wird der quadratisch wachsende Tiefenfehler auf diese Weise auch hier implizit berücksichtigt.



(a) 3D Bounding Boxen



(b) Rasterkarte mit in Graustufen dargestellter Belegheit der einzelnen Zellen und den geschätzten Objektboxen.

Abbildung 5.4: In das linke Kamerabild und in die Rasterkarte projizierte Ergebnisse der 3D Bounding Box Schätzung. Dabei sind Autos in rot, Radfahrer in blau und Fußgänger in grün dargestellt.

6 Lernbasierte Boxschätzung mithilfe von Rasterkarten-Ausschnitten

In diesem Kapitel wird der Ansatz aus Kapitel 5 erweitert und ein Ausschnitt der erstellten Rasterkarten pro Objekt verwendet um mit einem CNN direkt dessen 3D Bounding Box Parameter zu schätzen. Es basiert auf unserer Veröffentlichung [KS20] und erweitert diese um eine genauere Modellierung der Belegtheit der Rasterkartenzellen.

Abschnitt 6.1 gibt einen Überblick über den verwendeten Ansatz und beschreibt wie mit diesem einige Fehlerquellen der Methode aus Kapitel 5 beseitigt werden können. Die Netzarchitektur inklusive der verwendeten Verlustfunktion und des Optimierers werden in Abschnitt 6.2 vorgestellt. Anschließend wird in Abschnitt 6.3 darauf eingegangen, welche verschiedenen Möglichkeiten von Parameterzusammensetzungen existieren um ein rotiertes Rechteck eindeutig zu beschreiben. Die Erzeugung der für das Training des CNNs benötigten Daten wird in Abschnitt 6.4 beschrieben. Abschnitt 6.5 erläutert welche Metriken während des Trainings verwendet werden um insbesondere die verschiedenen Box-Kodierungen miteinander vergleichen zu können.

6.1 Übersicht

Trotz der, wie später in der Evaluation zu sehen, durchaus guten Ergebnisse mit dem Ansatz aus Kapitel 5, gibt es einige Fehlerquellen, welche in diesem Kapitel beseitigt werden sollen. Neben der bereits angesprochenen Approximation des nicht gaußförmig verteilten Tiefenfehlers, welcher für weit entfernte Objekte zu einer Boxschätzung führt, die tendenziell etwas hinter der wahren Objektposition liegt, gibt es noch zwei weitere Einschränkungen. Wie in Abbildung 5.4a zu sehen, ist die geschätzte Orientierung durch die gewählte

Optimierungsfunktion nur innerhalb eines π eindeutig, was bedeutet, dass keinerlei Information über die Bewegungsrichtung des Objekts vorliegen. Dieses Problem lässt sich zumindest für dynamische Objekte in der späteren Anwendung leicht beheben, indem die Informationen eines weiteren Zeitpunktes genutzt werden, dennoch kann es von Vorteil sein die genaue Orientierung aus einem einzigen Frame heraus zu schätzen.

Außerdem gibt es eine gewisse Tendenz dazu, dass die geschätzten Boxen eine Orientierung in Sichtstrahlrichtung besitzen, da die gewählte Optimierungsfunktion dafür sorgt, dass möglichst viele belegte und unbeobachtete Zellen innerhalb der Boxschätzung liegen und insbesondere an vertikalen Objektkanten besonders viele Punkte detektiert werden und unbeobachtete Zellen folglich in Sichtstrahlrichtung hinter diesen liegen.

Deshalb wird für den Schritt der Bestimmung der Boxparameter aus den erstellten Rasterkarten in diesem Kapitel ein CNN verwendet. Um eine geringe Rechenzeit zu ermöglichen, wird dabei nur ein kleiner Ausschnitt der Rasterkarte um das jeweilige Objekt verwendet. Das Netz hat dann die Aufgabe in diesem Ausschnitt ein rotiertes Rechteck zu schätzen, welches die x-z-Position, Breite und Tiefe und Orientierung der 3D-Box repräsentiert. Die y-Position und Höhe der Box wird anschließend analog zu Kapitel 5 bestimmt.

6.2 Netzarchitektur

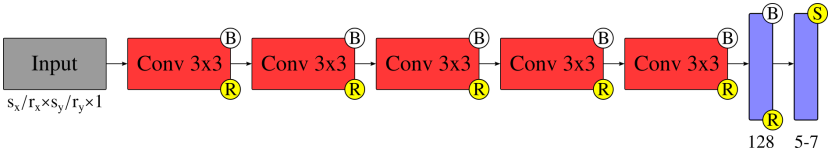


Abbildung 6.1: Architektur des verwendeten CNNs. Dabei sind Faltungsschichten in rot und vollständig verbundene Schichten in blau dargestellt. Die gelben Kreise beschreiben die jeweilige Aktivierung der Schicht, wobei das R für eine ReLU-Aktivierung und das S für eine Sigmoid-Aktivierung steht. Das B kennzeichnet eine auf diese Schicht folgende Batch-Normalisierung. Als Input dient ausschließlich der Rasterkartenausschnitt des jeweiligen Objektes.

Neben den bereits angesprochenen kleinen Rasterkartenausschnitten wird ein sehr flaches CNN genutzt um möglichst geringe Inferenzzeiten zu erreichen.

Das Netz besteht aus fünf Faltungsschichten und zwei vollständig verbundenen Schichten und ist in Abbildung 6.1 dargestellt. Die Eingangsdimension hängt von der Größe der Rasterkarten-Patches in Pixeln ab. Nach den fünf Faltungsschichten mit ReLU-Aktivierung und anschließender Batch-Normalisierung wird die Ausgabe ausgerollt (engl. Flattening) und es folgen zwei vollständig verbundene Schichten mit ReLU beziehungsweise Sigmoid-Aktivierung, wobei der Ausgang der zweiten Schicht die Dimension der verwendeten Box-Kodierung besitzt. Für diese Box-Kodierung stehen verschiedene Möglichkeiten zur Verfügung, welche in Abschnitt 6.3 genauer beschrieben werden.

Für die Regression der N Parameter der jeweiligen Box-Kodierung wird die L2-Verlustfunktion

$$L = \frac{1}{N} \sum_{i=1}^{N} (\hat{y}_i - y_i)^2 \tag{6.1}$$

appliziert. Als Optimierer kommt Adadelta [Zei12] zum Einsatz. Dies ist ein stochastisches Gradientenverfahren, welches die Lernrate je Parameter anpasst. Für Parameter, welche häufig auftretende Merkmale repräsentieren, werden kleinere Anpassungen durchgeführt, wohingegen Parametern, die mit seltenen Merkmalen verbunden sind, höhere Lernraten zugeteilt werden. Die hier seltener auftretenden weit entfernten Boxen, die einen dementsprechenden Tiefenfehler aufweisen, können dadurch besser repräsentiert werden. Anstatt alle vergangenen Gradienten zu akkumulieren nutzt es außerdem nur ein bestimmtes Fenster um eine streng monoton sinkende Lernrate zu umgehen, welche dazu führen könnte, dass der Algorithmus von einem bestimmten Punkt an kein zusätzliches Wissen mehr erlangen könnte. Durch die Verwendung von Adadelta ist auch keine vorgegebene Lernrate notwendig, da diese sowieso mit der Aktualisierungsregel eliminiert wird. Dies ist insbesondere von Vorteil für die, wie später in der Evaluation zu sehen, im Trainingsdatensatz unterschiedlich häufig vorkommenden Objekttypen, welche sonst eine Abstimmung der Lernrate benötigen würden.

6.3 Box-Kodierungen

Am einfachsten lässt sich ein rotiertes Rechteck durch seinen Mittelpunkt $[x_c, y_c]$, seine Breite w, seine Höhe h und seine Ausrichtung θ beschreiben. Wenn die Orientierung jedoch direkt verwendet wird, kann ihre Schätzung aufgrund der Winkelperiodizität Probleme verursachen. Beispielsweise ist ein Rechteck mit einem Drehwinkel von π identisch mit dem gleichen Rechteck mit dem Drehwinkel $-\pi$, ihre Winkel sind bei dieser Kodierung aber maximal unterschiedlich.

Um dieses Problem zu lösen, schlagen Beyer et al. [BHL15] Biternionen vor, welche die Orientierung durch zwei Parameter $\cos(\theta)$ und $\sin(\theta)$ repräsentieren. Diese Orientierungsvektordarstellung behandelt implizit die Winkelperiodizität, da jedes $\theta \in [-\pi, \pi]$ durch einen eindeutigen Vektor dargestellt werden kann.

In [WFSF18] wird dieser Ansatz leicht angepasst, indem $\cos(2\theta)$ und $\sin(2\theta)$ verwendet werden, die eine eindeutige Abbildung innerhalb von $[-\frac{\pi}{2}, \frac{\pi}{2}]$ darstellen. Dies ist vorteilhaft für die Regressionsgenauigkeit der Box, da Orientierungen in einem Abstand von π zu derselben Bounding Box führen. Der entstehende Nachteil der Unwissenheit über die Bewegungsrichtung des Objekts, kann durch einen zusätzlichen Parameter d ausgeglichen werden. Dieser Parameter gibt an, ob es sich um ein entgegenkommendes oder in dieselbe Richtung bewegendes Objekt handelt.

Außer bei der ersten Kodierung werden bei den vorgeschlagenen Kodierungen zwar mehr Parameter als die Mindestanzahl von fünf verwendet, jedoch kann dadurch eine genauere Orientierungsschätzung erwartet werden. Eine weitere Kodierung, die lediglich die Mindestanzahl an 2D-Box-Parametern verwendet und Winkeldarstellungen vollständig auslässt, ist in [JZW+17] zu finden. Anstatt einen Winkel zur Darstellung der Orientierungsinformationen zu verwenden, werden die Koordinaten von zwei Punkten und die Höhe der Bounding Box genutzt. Der erste Punkt $[x_1, y_1]$ stellt immer den Punkt an der linken oberen Ecke dar. Der zweite Punkt $[x_2, y_2]$ ist der nachfolgende Punkt im Uhrzeigersinn und h_3 ist der Abstand zwischen dem zweiten und dritten Punkt der Bounding Box. Durch den Abgleich dieses Parameters mit dem Abstand des ersten zum zweiten Punkt kann zurück auf eine Orientierung geschlossen werden, welche innerhalb eines π eindeutig ist. Einzige Ausnahme bildet der

Fall einer Box mit gleicher Breite und Höhe. Eine Zusammenfassung aller beschriebenen Kodierungen ist in Tabelle 6.1 zu finden.

Box-Kodierung	Parameter
B1	$[x_c, y_c, w, h, \theta]$
B2	$[x_c, y_c, w, h, \cos(\theta), \sin(\theta)]$
B3	$[x_c, y_c, w, h, \cos(2\theta), \sin(2\theta)]$
B4	$[x_c, y_c, w, h, \cos(2\theta), \sin(2\theta), d]$
B5	$[x_1, y_1, x_2, y_2, h_3]$

Tabelle 6.1: Übersicht über alle in diesem Abschnitt beschriebenen Box-Kodierungen.

6.4 Trainingsdatengenerierung

Um das Netz trainieren zu können, werden Ground Truth Daten aus den Trainingsdaten des KITTI Objekterkennungsdatensatzes [GLU12] erzeugt. Dafür werden Semantik und Disparität der Stereobilder bestimmt und die Objektcluster innerhalb der 2D-Detektionen aus dem Datensatz, wie in Kapitel 5 beschrieben, geschätzt. Unter Verwendung aller Punkte eines jeden Clusters wird der Schwerpunkt in der x-z-Ebene $[\mu_x, \mu_z]$ berechnet. Dieser Punkt ist das Zentrum des jeweiligen Rasterkartenausschnitts. Dadurch wird sichergestellt, dass bei der späteren Verwendung des CNNs nicht nur die korrekte Position des Objekts in diesem Ausschnitt sondern auch die globale Position ermittelt werden kann. Darüber hinaus kann so der bereits angesprochene Tiefenfehler implizit mitgeschätzt werden, da die Ground Truth Box nicht notwendigerweise in der Mitte der Patches liegt. Alle Punkte jedes Clusters werden in Rasterkartenausschnitte mit der Größe in Metern $[s_x, s_y]$ und der Auflösung $[r_x, r_y]$ projiziert:

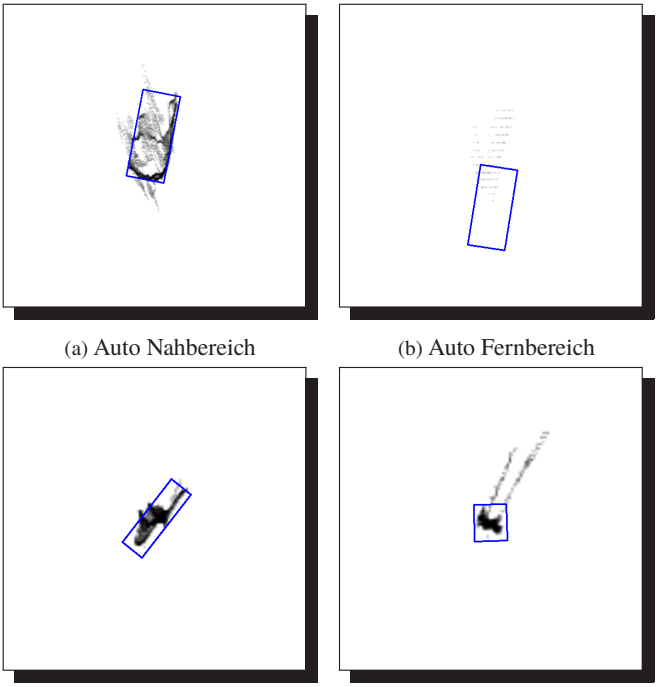
$$x = \frac{1}{r_x}\left(\frac{(u - c_u)\cdot b}{d} + \frac{s_x}{2} - \mu_x\right), \tag{6.2}$$

$$y = \frac{1}{r_y}\left(\frac{s_y}{2} - \frac{f\cdot b}{d} + \mu_z\right). \tag{6.3}$$

Die Belegtheit jeder Rasterzelle wird durch Gleichung 5.5 berechnet und definiert den Grauwert der jeweiligen Zelle in der Rasterkarte. Auf die gleiche Weise werden aus der x- und z-Koordinate und der Breite, Tiefe und Orientierung der 3D-Boxen der Trainingsdaten rotierte Rechtecke in den Koordinaten der Rasterkarten-Patches berechnet. Hierfür können die in Abschnitt 6.3 beschriebenen verschiedenen Box-Kodierungen verwendet werden. Auf eine explizite Berücksichtigung unbeobachteter Zellen wird bei diesem Ansatz verzichtet, da die Dichte der einzelnen Punkte wertvolle Informationen über die Entfernung des Objekts und somit der Unsicherheit der Tiefenschätzung enthält.

Abbildung 6.2 zeigt vier Beispiele für diese Patches mit den dazugehörigen Ground Truth Boxen. Die Dichte der Punkte zeigt, dass Patch (a) eher von einem Objekt stammt, welches aus dem Nahbereich ist und bei dem die Box sehr gut mit den einzelnen Rasterkartenpunkten übereinstimmt. Patch (b) hingegen gehört zu einem Objekt, das sich in einem größeren Abstand zur Kamera befindet, sodass die Punkte einen gewissen Versatz gegenüber der tatsächlichen Box aufweisen. An dieser Stelle sei aber auch erwähnt, dass die Ground Truth des verwendeten Datensatzes basierend auf LiDAR-Punktwolken annotiert wurde und dadurch ein potenzieller Kalibrierfehler zwischen Kamera und LiDAR ebenfalls zu einem solchen Versatz beitragen kann. Nichtsdestotrotz sorgt der in diesem Kapitel vorgestellte lernbasierte Ansatz für eine Kompensation dieser beiden Fehlerquellen. Des Weiteren sind noch die Patches eines Fahrrads und eines Fußgängers dargelegt.

Um die Menge der verfügbaren Trainingsdaten insbesondere im Fall der sehr unterrepräsentierten Klassen Radfahrer und Fußgänger zu erhöhen, werden zusätzliche Daten augmentiert. Die erstellten Rasterkartenausschnitte enthalten implizit Stereoinformationen, was bedeutet, dass die Dichte der Punkte und auch die Form entlang der y-Achse nicht verändert werden sollte. Aus diesem Grund besteht die einzige Möglichkeit zusätzliche Patches zu erstellen darin, diese entlang der y-Achse zu spiegeln.



(a) Auto Nahbereich (b) Auto Fernbereich

(c) Fahrrad (d) Fußgänger

Abbildung 6.2: Vier Beispiele von Rasterkarten-Patches mit den dazugehörigen Ground Truth Boxen in blau. Die Auflösung beträgt für alle Patches $r_x = r_y = 0{,}05\,\text{m}$, die Größe $s_x = s_y = 12{,}8\,\text{m}$ für (a) und (b) und $s_x = s_y = 6{,}4\,\text{m}$ für (c) und (d).

6.5 Metriken

Um die verschiedenen Box-Kodierungen mit unterschiedlicher Anzahl Parametern vergleichen zu können, wird als Metrik während des Trainings die Intersection over Union (auch Jaccard-Koeffizient)

$$\text{IoU}(\hat{\mathcal{B}}, \mathcal{B}_{\mathbf{GT}}) = \frac{|\hat{\mathcal{B}} \cap \mathcal{B}_{\mathbf{GT}}|}{|\hat{\mathcal{B}} \cup \mathcal{B}_{\mathbf{GT}}|} = \frac{|\hat{\mathcal{B}} \cap \mathcal{B}_{\mathbf{GT}}|}{|\hat{\mathcal{B}}| + |\mathcal{B}_{\mathbf{GT}}| - |\hat{\mathcal{B}} \cap \mathcal{B}_{\mathbf{GT}}|} \tag{6.4}$$

der rotierten Rechtecke verwendet. Diese dient als Maß für die Ähnlichkeit zwischen Box-Schätzung $\hat{\mathcal{B}}$ und Ground Truth $\mathcal{B}_{\mathbf{GT}}$ und kann durch die Bestimmung der Schnittfläche der beiden Boxen $|\hat{\mathcal{B}} \cap \mathcal{B}_{\mathbf{GT}}|$, welche im Falle einer nicht-leeren Schnittmenge in einem konvexen Polygon resultiert, und den einzelnen Boxflächen $|\hat{\mathcal{B}}| = \hat{w}\hat{h}$ und $|\mathcal{B}_{\mathbf{GT}}|$ berechnet werden.

Wie bereits in Abschnitt 6.3 angesprochen, führen Orientierungen in einem Abstand von π zu der selben Box. Im Falle einer Abweichung zwischen Schätzung und Ground Truth von π würde sich dennoch eine IoU von 1 ergeben, obwohl die Bewegungsrichtung maximal unterschiedlich ist. Liegen Objekte mit gleicher Breite und Höhe vor, passiert dies sogar für alle Orientierungsabweichungen, welche ein Vielfaches von $\frac{\pi}{2}$ sind. Speziell um auch den Unterschied zwischen den Box-Kodierungen deutlich machen zu können, welche nur innerhalb π eindeutig sind und welche innerhalb 2π, ist es sinnvoll eine weitere Metrik für die Genauigkeit der Orientierungsschätzung einzuführen. Dafür wird die auch im KITTI Objekterkennungs-Benchmark [GLU12] eingesetzte Average Orientation Similarity

$$\mathrm{AOS}(\hat{\theta}, \theta_{\mathrm{GT}}) = \frac{1 + \cos(\theta_{\mathrm{GT}} - \hat{\theta})}{2} \tag{6.5}$$

genutzt, welche für Abweichungen von π den minimalen Wert 0 liefert und so in Verbindung mit der IoU eine optimale Kombination darstellt um die verschiedenen Box-Kodierungen evaluieren zu können.

7 Monokulare 3D-Objektdetektion

In diesem Kapitel wird erläutert, wie die 3D-Informationen von Objekten direkt aus einem einzigen Kamerabild gewonnen werden können. Es basiert auf unserer Veröffentlichung [KLS20], unterscheidet sich von dieser aber durch eine andere monokulare Tiefenschätzung, der Bestimmung von 2D-Detektionen und Semantik in einem einzigen Schritt und kleineren Änderungen in den verwendeten Verlustfunktionen. Die einzelnen Verarbeitungsschritte dieses Ansatzes werden in Abschnitt 7.1 beschrieben. Abschnitt 7.2 geht auf die verwendete Netzarchitektur und die genaue Zusammensetzung der Verlustfunktion ein. In Abschnitt 7.3 wird erläutert, wie eine Überanpassung des Netzes durch die geeignete Augmentation von zusätzlichen Trainingsdaten verhindert werden kann.

7.1 Übersicht

Die einzelnen Schritte vom Kamerabild bis hin zur 3D Bounding Box sind in Abbildung 7.1 zusammengefasst. Für die 2D-Objektdetektion und semantische Segmentierung kommt das bereits in Abschnitt 5.1.1 beschriebene CNN zum Einsatz. Die Tiefe wird mithilfe des BTS-Netzes [LHKS19] geschätzt. Dieses überwachte monokulare Netz zur Tiefenschätzung bestimmt Merkmale auf verschiedenen Auflösungsstufen und stellt durch sogenannte lokale Führungsschichten eine explizite Beziehung dieser internen Merkmalskarten zur gewünschten Prädiktion her. Beide Aufgaben werden für eine schnellere Laufzeit parallel ausgeführt.

Im Gegensatz zu Weng et al. [WK19], welche die (Instanz-)Segmentierung direkt als Maske für die Tiefeninformation nutzen, wird die gesamte semantische Information in jeder 2D-Detektion verwendet. Dies führt zu einer hö-

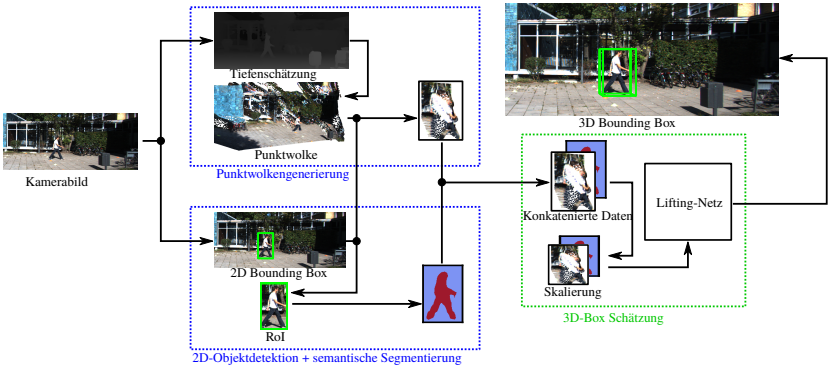


Abbildung 7.1: Übersicht über die einzelnen Schritte der monokularen Objektdetektion. Die blauen Umrandungen kennzeichnen die verwendeten Eingangsdaten, der grüne Rahmen zeigt den Kernbaustein des in diesem Kapitel vorgestellten Ansatzes.

heren Robustheit sowohl gegen Fehler in der Tiefenschätzung als auch in der Segmentierung. Außerdem konnte festgestellt werden, dass sowohl der Hintergrund als auch die Wahrscheinlichkeitsverteilung über alle Klassen wertvolle Informationen für die 3D Bounding Box Schätzung liefern. Statt also nur die wahrscheinlichste Klasse pro Bildpunkt zu verwenden, werden die Pseudo-Wahrscheinlichkeiten aller semantischen Klassen genutzt.

Für alle Punkte innerhalb der jeweiligen 2D Bounding Box wird die 3D-Position berechnet und mit der Semantik innerhalb dieser RoI konkateniert, was in einer Bildgröße von $W \times H \times (C+3)$ resultiert. W und H sind die Breite und Höhe jeder RoI und C ist die Gesamtklassenanzahl aus der semantischen Segmentierung. Die zusätzlichen drei Kanäle stellen die Koordinaten x, y und z jedes Pixels dar. Da die konkatenierten Ausschnitte zwar die Daten einer Punktwolke beinhalten aber in Bildformat bleiben, können sie auf eine vorgegebene Größe skaliert und mit dem im folgenden Abschnitt genauer beschriebenen Lifting-Netz weiterverarbeitet werden.

7.2 Architektur des Lifting-Netzes

Die Architektur des Lifting-Netzes ist in Abbildung 7.2 schematisch dargestellt. Es besteht aus einem modifizierten ResNet50 [HZRS16] und einem MLP, welches die Parameter der 3D Bounding Box schätzt. Das modifizierte ResNet50 erzeugt für die auf eine einheitliche Größe skalierten Ausschnitte mit semantischen und räumlichen Informationen eine Feature Map, welche in einen Merkmalsvektor ausgerollt wird. Dieser wird mit den projizierten 3D-Koordinaten des 2D-Box-Mittelpunktes $\mathbf{p_m}$, den mittleren Ausmaßen dieses Objekttyps $\mathbf{d_{prior,k}}$ und zusätzlich noch einem sogenannten One-Hot Vektor, welcher die Klasse des Objekts aus der 2D-Detektion angibt, konkateniert. Dieser enthält eine Eins an der Stelle des vorliegenden Objekttyps und ansonsten Nullen. Durch die Kenntnis der Objektklasse können neben der Größenschätzung, welche schon durch die mittlere Größe von dieser profitiert auch die Schätzung der Position und Orientierung besser für jede einzelne Klasse trainiert werden. Der konkatenierte Merkmalsvektor geht dann durch drei Zweige mit vollständig verbundenen Schichten um die Position $\Delta\mathbf{p}$, die Größe $\Delta\mathbf{d}$ und die durch zwei Parameter beschriebene Orientierung $\mathbf{b}_\theta$ und $\Delta\hat{\theta}$ für die 3D Bounding Box zu schätzen.

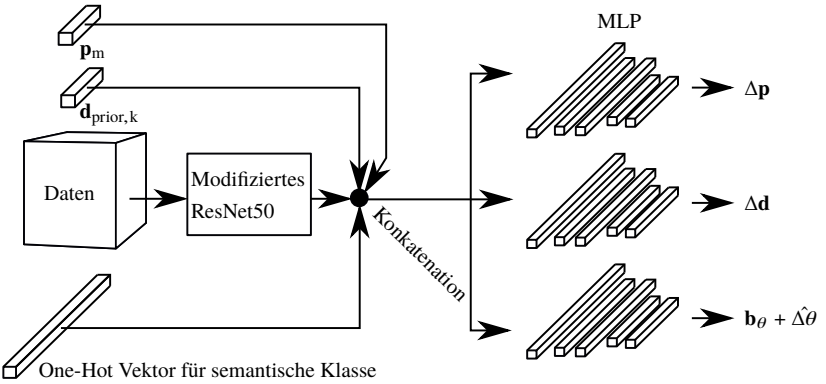


Abbildung 7.2: Schematische Darstellung der Architektur des Lifting-Netzes. Der durch das ResNet50 bestimmte Merkmalsvektor wird gemeinsam mit den 3D-Koordinaten des 2D-Box-Mittelpunktes $\mathbf{p_m}$, den mittleren Objektdimensionen $\mathbf{d_{prior,k}}$ und der semantischen Klasse konkateniert und einem dreizweigigen MLP zur Schätzung der 3D Bounding Box Parameter übergeben.

Die Schätzung dieser Parameter inklusive der verwendeten Verlustfunktionen wird in den nächsten drei Abschnitte genauer beschrieben. Die Gesamtverlustfunktion des Lifting-Netzes ergibt sich dann aus der mit den Hyperparametern λ_1 und λ_2 gewichteten Summe dieser drei Verlustfunktionen für die Positionsschätzung $L_{\Delta\mathbf{p}}$, die Größenschätzung $L_{\Delta\mathbf{d}}$ und die Orientierungsschätzung L_θ zu:

$$L_{\text{total}} = L_{\Delta\mathbf{p}} + \lambda_1 L_{\Delta\mathbf{d}} + \lambda_2 L_\theta. \tag{7.1}$$

Aufgrund unterschiedlicher Maßstäbe der semantischen Information ($[0, 1]$) und der räumlichen Koordinaten ($[-50\,\text{m}, 50\,\text{m}]$) wird das ResNet50 modifiziert, indem alle Batch-Normalisierungsschichten entfernt werden. Eine andere Möglichkeit wäre, auch die räumlichen Koordinaten auf $[0, 1]$ zu normalisieren. Da auf diese Weise alle Punkte auf einen kleinen Bereich schrumpfen würden, wären viele geometrische Merkmale numerisch schwieriger zu unterscheiden. Zusätzlich haben die Experimente gezeigt, dass das Training mit normalisierten Koordinaten für dieses Netz zu einer schlechteren Konvergenz führt.

7.2.1 Positionsschätzung

Um die genaue Position des Objektes zu bestimmen, schätzt das Netz die Abweichung $\hat{\Delta\mathbf{p}}$ zu den projizierten 3D-Koordinaten des Mittelpunktes der 2D Bounding Box $\mathbf{p_m} = (x_\text{m}, y_\text{m}, z_\text{m})^\text{T}$. Dadurch ergibt sich die Verlustfunktion

$$L_{\Delta\mathbf{p}} = \mathcal{H}_1\left(\|\hat{\Delta\mathbf{p}} - \Delta\mathbf{p_{GT}}\|\right). \tag{7.2}$$

$\Delta\mathbf{p_{GT}}$ ist dabei die Differenz zwischen der Ground Truth Position und $\mathbf{p_m}$ und $\mathcal{H}_1()$ die Huber-Verlustfunktion (engl. auch smooth L1 loss). Dies ist eine Kombination aus L1- und L2-Verlustfunktion welche weniger empfindlich gegenüber Ausreißern ist als die L2-Verlustfunktion und ist definiert durch

$$\mathcal{H}_1(x) = \begin{cases} \frac{1}{2}x^2 & |x| \leq 1 \\ |x| - \frac{1}{2} & \text{sonst.} \end{cases} \tag{7.3}$$

7.2.2 Größenschätzung

Da die Objektklassifizierung durch die 2D-Detektion bereits bekannt ist, können die mittleren Größen $\mathbf{d_{prior,k}} = (h_{\text{prior,k}}, w_{\text{prior,k}}, l_{\text{prior,k}})^{\text{T}}$ jedes Objekttyps k, wie bereits in Kapitel 5, als zusätzliche Information verwendet werden. Diese können zum Beispiel aus einem Datensatz stammen.

Das Modell schätzt dann die normierte Abweichung $\frac{\hat{\Delta\mathbf{d}}}{\mathbf{d_{prior,k}}}$ mit $\hat{\Delta\mathbf{d}} = \hat{\mathbf{d}} - \mathbf{d_{prior,k}}$. Dies erleichtert das Lernen, da die Werte auf einen kleineren Bereich beschränkt werden. Die Verlustfunktion ergibt sich dementsprechend durch

$$L_{\Delta\mathbf{d}} = \mathcal{H}_1\left(\left\|\begin{bmatrix} \frac{\hat{\Delta h} - \Delta h_{\text{GT}}}{h_{\text{prior,k}}} \\ \frac{\hat{\Delta w} - \Delta w_{\text{GT}}}{w_{\text{prior,k}}} \\ \frac{\hat{\Delta l} - \Delta l_{\text{GT}}}{l_{\text{prior,k}}} \end{bmatrix}\right\|\right) \tag{7.4}$$

mit $\Delta_{\cdot\text{GT}}$ als Abweichung der jeweiligen Ground Truth von den typischen Objektdimensionen.

7.2.3 Orientierungsschätzung

Eine Schätzung der globalen Objektorientierung θ nur aus dem Inhalt des RoI-Ausschnitts ist nicht möglich, da auch die Lage des Ausschnitts innerhalb der Bildebene benötigt wird. Abbildung 7.3a zeigt ein Beispiel für ein Auto, welches sich geradlinig bewegt. Obwohl sich die globale Orientierung des Fahrzeugs nicht ändert, ändert sich seine lokale Ausrichtung θ_l in Bezug auf den Strahl durch das RoI-Zentrum und erzeugt Änderungen im Aussehen des ausgeschnittenen Bildes. In dem in diesem Kapitel vorgestellten Netz wird deshalb die lokale Orientierung geschätzt. Bei gegebenen intrinsischen

(a) RoI-Ausschnitte und komplette Szene

(b) Skizze der verschiedenen Orientierungen

Abbildung 7.3: Veranschaulichung der lokalen und globalen Orientierung anhand des Beispiels eines entgegenkommenden Fahrzeugs.

Kameraparametern ist die Berechnung der in Abbildung 7.3b dargestellten Sichtstrahlrichtung θ_{ray} zu einem bestimmten Pixel trivial. Zur Inferenzzeit wird die Sichtstrahlrichtung zum Zentrum des jeweiligen Ausschnitts mit der geschätzten lokalen Orientierung kombiniert um die globale Orientierung $\theta = \theta_l + \frac{\pi}{2} - \theta_{\text{ray}}$ zu berechnen.

Wie in [MAFK17] beschrieben, kann die direkte Schätzung der Orientierung in einem multimodalen Regressionsproblem zu Problemen führen. Deshalb wird auch hier die in dieser Arbeit vorgeschlagene *MultiBin* Architektur verwendet. Dabei wird der Winkel diskretisiert und in N_θ Bereiche geteilt. Für jeden Abschnitt schätzt das CNN sowohl eine Konfidenz c_i, dass der Ausgabewinkel innerhalb des i-ten Abschnitts liegt, als auch einen Korrekturwinkel $\hat{\Delta\theta}$, der auf die Orientierung des mittleren Winkels dieses Abschnitts angewendet werden muss um den Ausgabewinkel zu erhalten. Der Korrekturwinkel wird mit der bereits in Abschnitt 6.3 beschriebenen Biternionendarstellung durch zwei Parameter $\cos(\Delta\theta)$ und $\sin(\Delta\theta)$ repräsentiert. Die Bestimmung der Konfidenzen der einzelnen Winkelabschnitte stellt ein Klassifikationsproblem dar, für welches die Kreuzentropie

$$L_{\mathbf{b}_\theta} = -\sum_{i=1}^{N_\theta} \mathbf{b}_\theta(i) \log(\hat{p}(c_i)) \tag{7.5}$$

als Verlustfunktion und die Softmax-Funktion

$$\hat{p}(c_i) = \frac{e^{c_i}}{\sum_{n=1}^{N_\theta} e^{c_n}} \tag{7.6}$$

als Wahrscheinlichkeitsmaß verwendet werden. $\mathbf{b}_\theta$ ist ein Vektor mit N_θ Elementen in One-Hot Kodierung, der eine Eins an der Stelle des Ground Truth Winkelabschnitts enthält.

Die Verlustfunktion $L_{\Delta\theta}$ versucht die Differenz zwischen $\hat{\Delta\theta}$ und dem Ground Truth Korrekturwinkel $\Delta\theta_{\mathrm{GT}}$ zu minimieren und resultiert in

$$L_{\Delta\theta} = \mathcal{H}_1 \left(\left\| \begin{bmatrix} \cos(\hat{\Delta\theta}) \\ \sin(\hat{\Delta\theta}) \end{bmatrix} - \begin{bmatrix} \cos(\Delta\theta_{\mathrm{GT}}) \\ \sin(\Delta\theta_{\mathrm{GT}}) \end{bmatrix} \right\| \right). \tag{7.7}$$

Die gesamte Verlustfunktion für die Orientierungsschätzung ergibt sich dementsprechend zu:

$$L_\theta = L_{\mathbf{b}_\theta} + \lambda L_{\Delta\theta}. \tag{7.8}$$

Der Hyperparameter λ balanciert die relative Bedeutung der beiden Anteile.

7.3 Datenaugmentation

Um das Modell zu verallgemeinern und eine Überanpassung zu vermeiden, wird für das Training eine Datenaugmentation genutzt, bei welcher anstatt der Ground Truth Boxen, die Form der 2D Bounding Boxen variiert wird.

Wie in Abbildung 7.4 dargestellt, wird mithilfe des linken oberen Punktes $\mathbf{q_1} = [u_1, v_1]^{\mathrm{T}}$ und des rechten unteren Punktes $\mathbf{q_2} = [u_2, v_2]^{\mathrm{T}}$ der 2D Bounding Box, die Breite w und Höhe h berechnet. Anschließend werden zufällige Werte für die Verschiebung $\Delta h_1, \Delta h_2 \in [-0{,}25h \;\; 0{,}25h]$ und $\Delta w_1, \Delta w_2 \in [-0{,}25w \;\; 0{,}25w]$ gewählt und die neue Bounding Box mit $\mathbf{q_1^*} = [u_1 + \Delta w_1, v_1 + \Delta h_1]$ und $\mathbf{q_2^*} = [u_2 + \Delta w_2, v_2 + \Delta h_2]$ bestimmt.

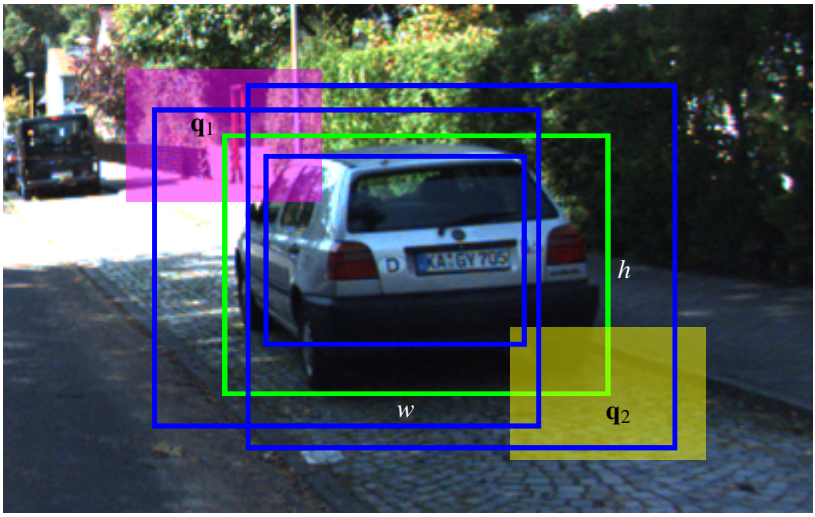


Abbildung 7.4: Datenaugmentation für das Lifting-Netz. Die grüne Box zeigt die Ground Truth 2D Bounding Box des Autos. In Abhängigkeit der Breite w und Höhe h ergibt sich für die möglichen Positionen des linken oberen Punktes $\mathbf{q}_1^*$ der lila Bereich und des rechten unteren Punktes $\mathbf{q}_2^*$ der gelbe Bereich. Die drei blauen Boxen sind Beispiele für zufällig variierte Bounding Boxen.

8 Experimentelle Auswertung

In diesem Kapitel werden die in den Kapiteln 5, 6 und 7 beschriebenen Methoden evaluiert und untereinander sowie zu anderen Arbeiten verglichen. Die Evaluationen sind hierbei in der gleichen Reihenfolge zu finden, in welcher die Ansätze beschrieben wurden und verwenden hauptsächlich den KITTI 3D-Objekterkennungsdatensatz [GLU12], welcher in Abschnitt 8.1 kurz vorgestellt wird. Abschnitt 8.2 geht dann auf die verwendeten Parameter für die Rasterkartenbasierte Objektdetektion aus Kapitel 5 ein und bewertet die Genauigkeit der 2D- und 3D-Detektionen. In Abschnitt 8.3 werden verschiedene Auflösungen und Box-Kodierungen für die Rasterkarten-Patches der lernbasierten Boxschätzung aus Kapitel 6 evaluiert. Die Ergebnisse der monokularen 3D-Objektdetektion aus Kapitel 7 und insbesondere die Verschiebungsinvarianz des Lifting-Netzes werden in Abschnitt 8.4 vorgestellt. Abschnitt 8.5 zeigt die Ergebnisse dieser drei Ansätze auf dem Testdatensatz des KITTI Objekterkennungsbenchmarks und vergleicht diese mit anderen Arbeiten bezüglich Genauigkeit und Laufzeit. Eine qualitative Evaluation auf dem Versuchsträger BerthaOne in Abschnitt 8.6 bildet den Abschluss dieses Kapitels.

8.1 KITTI Objekterkennungsdatensatz

Der KITTI 3D-Objekterkennungsdatensatz [GLU12] besteht aus einem Trainingsdatensatz mit 7481 Bildern und einem Testdatensatz mit 7518 Bildern mit insgesamt ca. 80 000 in den zugehörigen LiDAR-Punktwolken annotierten Objekten mit einer maximalen Entfernung von ca. 80 m. Dazu gehören verschiedene Objektklassen, welche gemeinsam mit ihren Häufigkeiten in Tabelle 8.1 aufgelistet sind.

Die Evaluation auf dem Testdatensatz wird allerdings nur für die Klassen Auto, Fußgänger und Radfahrer durchgeführt, weshalb der Fokus in diesem und folgenden Abschnitten auf diesen drei Klassen liegt. Zusätzlich sei erwähnt,

Objektklasse	Häufigkeit	$\mu_{\mathbf{dim}}$ $[m]$			$\sigma_{\mathbf{dim}}$ $[m]$		
	[%]	h	w	l	h	w	l
Auto	70,8	1,53	1,63	3,88	0,14	0,10	0,43
Fußgänger	11,1	1,76	0,66	0,84	0,11	0,14	0,23
Transporter	7,2	2,21	1,90	5,08	0,32	0,17	0,83
Radfahrer	4,0	1,74	0,60	1,76	0,09	0,12	0,18
LKW	2,7	3,25	2,59	10,11	0,45	0,22	2,86
Sonstiges	2,4	1,91	1,51	3,57	0,81	0,67	2,86
Tram	1,3	3,53	2,54	16,09	0,18	0,22	7,86
Sitzende Person	0,6	1,27	0,59	0,80	0,11	0,08	0,22

Tabelle 8.1: Semantische Klassen des KITTI Objekterkennungsdatensatzes 2017 mit Häufigkeiten, Mittelwerten und Standardabweichungen für den Trainingsdatensatz.

dass bei der Auswertung auf dem Testdatensatz Transporter nicht als falsch positiv für Autos und sitzende Personen nicht als falsch positiv für Fußgänger betrachtet werden, da sie sich im Aussehen ähneln. Außerdem ist die Evaluation nochmal in die drei Schwierigkeitsstufen Einfach, Moderat und Schwierig unterteilt, welche sich durch die maximale Sichtbarkeit und Trunkierung an Bildrändern und die minimale Größe der Bounding Boxen unterscheiden.

Die einfachen Testdaten enthalten ausschließlich Objekte, welche vollständig sichtbar, an Bildrändern maximal zu 15 % abgeschnitten sind und eine minimale 2D Bounding Box Höhe von 40 px haben. Durch diese Beschränkung der minimalen Bounding Box Höhe, findet die Evaluation auf dem Testdatensatz dementsprechend nur für Objekte bis zu einer gewissen Entfernung abhängig von ihrer tatsächlichen Größe statt. So führt eine Höhe des Objektes von beispielsweise 1,7 m, für die Evaluation auf den einfachen Daten zu einer maximalen Objektentfernung von etwa 30 m. Beim Datensatz mit der Schwierigkeit Moderat sind alle vollständig sichtbaren und teilweise verdeckten Objekte mit einer maximalen Trunkierung von 30 % und einer minimalen 2D Bounding Box Höhe von 25 px enthalten. Dies führt für das eben genannte Beispiel zu einer maximalen Entfernung von etwa 50 m. Die schwierigen Daten umfassen alle Objekte mit den bereits genannten Verdeckungsgraden und zusätzlich Objekte, welche „schwer zu sehen“ sind. Außerdem wird eine Trunkierung von

maximal 50 % und ebenfalls eine minimale Bounding Box Höhe von 25 px gefordert.

Ebenfalls in Tabelle 8.1 zu finden, sind die mittleren Dimensionen und deren Standardabweichungen für jeden Objekttyp, welche für die Ansätze aus Kapitel 5 und 7 benötigt werden.

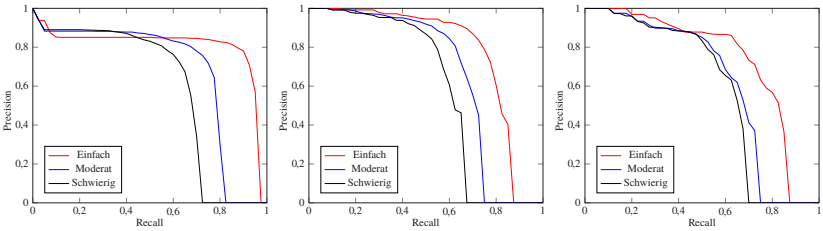
8.2 Evaluation der rasterkartenbasierten Objektdetektion

In diesem Abschnitt wird die Genauigkeit des Clusterings und der 3D-Detektionen des in Kapitel 5 vorgestellten Ansatzes evaluiert. Der in Abschnitt 5.2 beschriebene Assoziationsschwellwert für das Clustering wird zu $t_{\text{ass}} = 0{,}5\,\text{m}$ gewählt. Die Auflösung der Rasterkarten wird zu $r_{\text{x}} = r_{\text{z}} = 0{,}1\,\text{m}$ gewählt und die Anzahl Punkte, welche in eine Rasterzelle fallen müssen, damit diese als belegt gilt, zu $n_{\text{o}} = 16$. Eine Evaluation über verschiedene Auflösungen der Rasterkarte ist in Abschnitt 8.3.1 zu finden.

8.2.1 2D Bounding Box Evaluation

Da der KITTI Objekterkennungsdatensatz keine pixelweisen semantischen Annotationen enthält, wird das in Abschnitt 5.1.1 vorgestellte CNN zur semantischen Segmentierung und 2D-Objektdetektion zunächst auf dem Cityscapes Datensatz [COR+16] trainiert. Dafür werden sowohl die 20 000 Bilder mit grober als auch die 3475 Bilder mit feiner Annotation verwendet. Anschließend wird das Netz mit den 2D-Boxen des KITTI Datensatzes nachtrainiert. Dafür werden vom Trainingsdatensatz die Boxen von 5930 zufällig ausgewählten Bildern verwendet. Die restlichen Bilder und deren Boxen dienen als Validierungsdaten.

Für diese Validierungsdaten werden alle Objektcluster, wie in Abschnitt 5.2 beschrieben, bestimmt und in eine Bounding Box umgerechnet. Die Ergebnisse dieser Validierung stellen also direkt die Genauigkeit des verwendeten Clusterings dar, welches sowohl fehlerhafte Boxen des CNNs herausfiltern als auch zusätzliche Objekte in Bereichen ohne Bounding Boxen detektieren



(a) Autos (b) Fußgänger (c) Radfahrer

Abbildung 8.1: Precision-Recall Kurven der 2D-Detektionen auf dem Validierungsdatensatz für einen IoU-Schwellwert von 70 % für Autos und 50 % für Fußgänger und Radfahrer.

kann. Es sei jedoch erwähnt, dass sich die Ergebnisse insgesamt in einem ähnlichen Bereich befinden, wie die Resultate der 2D-Boxen des CNNs. Die Ergebnisse sind durch Precision-Recall-Kurven für alle drei Objektklassen in Abbildung 8.1 dargestellt. Der IoU-Schwellwert für eine korrekte Detektion entspricht hierbei dem gleichen wie im Test-Benchmark von KITTI und beträgt 70 % für Autos und 50 % für Fußgänger und Radfahrer. Die durchschnittliche Genauigkeit AP_{2D} für jede Klasse wird aus den Mittelwerten dieser Kurven berechnet und ergibt sich für die einfache Schwierigkeit zu 80,0 % für Autos, 77,5 % für Fußgänger und 73,2 % für Radfahrer.

Durch das gemeinsame Training auf Cityscapes und KITTI, können diese Ergebnisse nicht ganz mit den besten 2D-Detektoren auf KITTI mithalten, da letztere ausschließlich darauf trainiert wurden. Jedoch ergibt sich eine bessere Generalisierbarkeit auf andere Kameramodelle, wie zum Beispiel das in unserem Forschungsfahrzeug BerthaOne. Auch die mit etwa 40 ms deutlich geringere Laufzeit bei gleichzeitiger Schätzung der pixelweisen Semantik stellt einen weiteren Vorteil des verwendeten CNNs dar.

8.2.2 3D Bounding Box Evaluation

Auch für die Berechnung der durchschnittlichen Genauigkeit der 3D-Boxen wird ein Schwellwert der IoU von 70 % für Autos und 50 % für Fußgänger und Radfahrer verwendet. Diese Werte sind vor allem auch wegen der geringen Auflösung der KITTI-Bilder von nur 0,4 MP eine größere Herausforderung für

kamerabasierte Ansätze im Vergleich zu LiDAR-basierten. Dies wirkt sich insbesondere auf weit entfernte und kleine Objekte, wie Fußgänger und Radfahrer aus. Zwar würden in diesen Fällen auch weniger LiDAR-Punkte zur Schätzung der 3D Bounding Box zur Verfügung stehen, dennoch bietet LiDAR eine höhere Genauigkeit, welche zudem annähernd konstant über alle Entfernungen ist. Auch die höhere Position des LiDAR-Sensors, welche Vorteile bei Verdeckungen liefert, und etwaige Kalibrierfehler zwischen LiDAR und Kamera führen dazu, dass die speziell für LiDAR-basierte Methoden gewählten IoU-Schwellwerte nur eine gewisse Aussagekraft über die Anwendbarkeit des kamerabasierten Algorithmus liefern. Deshalb zeigt Abbildung 8.2 die durchschnittliche Genauigkeit der 3D-Boxen AP_{3D} aufgetragen über verschiedenen IoU-Schwellwerten.

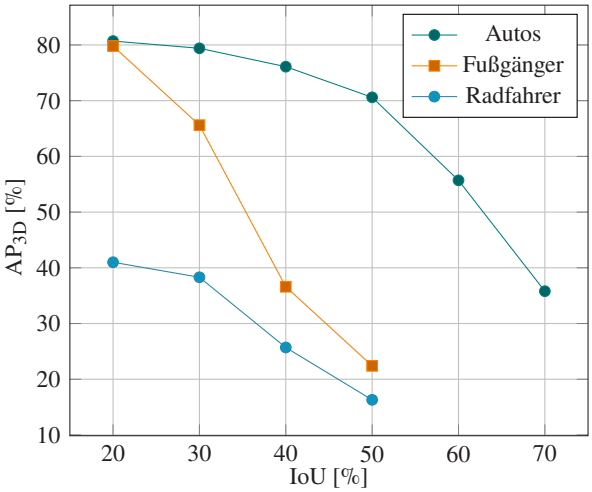


Abbildung 8.2: Durchschnittliche Genauigkeit auf den Validierungsdaten in Abhängigkeit des IoU-Schwellwertes.

Dabei werden für Autos relativ hohe Genauigkeiten bis zu einer IoU von 50 % erzielt. Bei Fußgängern und Radfahrern, die deutlich kleinere Abmessungen haben, werden bis zu einer IoU von 30 % solide Ergebnisse erzielt. Um eine bessere Vorstellung von der Bedeutung der jeweiligen IoUs für die drei Objektklassen zu bekommen, hier ein kurzes Beispiel: Bei den durchschnittlichen Abmessungen der drei Objektklassen ergibt sich bei einem Fehler von nur 0,5 m

in jede Richtung eine IoU von 50 % für Autos, 30 % für Radfahrer und 25 % für Fußgänger. Eine Positionsabweichung in diesem Bereich ist, zumindest in größerer Entfernung, ausreichend für die in automatisierten Fahrzeugen auf die Umfelderfassung folgenden Module wie die Prädiktion, Verhaltensgenerierung und Trajektorienplanung.

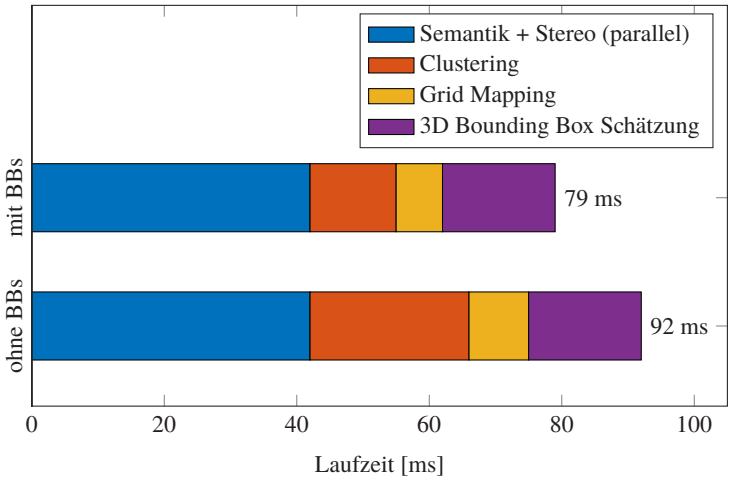


Abbildung 8.3: Aufschlüsselung der Laufzeiten der einzelnen Verarbeitungsschritte. Die semantische Segmentierung und die Disparitätsschätzung werden parallel ausgeführt und sind deshalb nicht einzeln aufgelistet.

In Abbildung 8.3 sind die Laufzeiten für die einzelnen Verarbeitungsschritte dargestellt, welche sich mit oder ohne Verwendung der 2D Bounding Boxen als zusätzliche Eingabe für das Clustering ergeben. Wie bereits in Abschnitt 5.2 beschrieben, resultieren die Unterschiede in den Rechenzeiten aus der besseren Parallelisierbarkeit bei Verwendung der Bounding Boxen. Die Schätzung der Bodenoberfläche ist hier nicht explizit aufgeführt, da deren Verwendung einerseits optional ist und andererseits mit einer Laufzeit von etwa 15 ms parallel zum Clustering ausgeführt werden kann. Die Laufzeiten wurden auf einer NVIDIA Titan X GPU ausgewertet. Sie sind die mittleren Zeiten über den gesamten Datensatz und können je nach Anzahl und Größe der zu erkennenden Objekte pro Bild entsprechend variieren. Ein Vergleich dieser Laufzeiten mit denen verwandter Arbeiten ist in Abschnitt 8.5.1 zu finden.

8.3 Evaluation der lernbasierten Boxschätzung mithilfe von Rasterkarten-Ausschnitten

In diesem Abschnitt werden verschiedene Parameter der in Kapitel 6 vorgestellten lernbasierten Boxschätzung in Rasterkarten-Ausschnitten experimentell verglichen. Dazu gehören die Auflösung und Größe der Patches, die verwendete Box-Kodierung und ob eine zusätzliche Augmentation von Daten bessere Ergebnisse liefert. Da bereits beim Training, aber auch bei der Inferenz durch die gegebenen oder geschätzten 2D Bounding Boxen die Information über den Typ des Objektes bekannt ist, werden die Trainingsdaten nach den Objektklassen aufgeteilt und dementsprechend drei verschiedene Netze trainiert.

8.3.1 Auflösung und Größe der Patches

Sowohl für die Auflösung als auch für die Größe der Rasterkarten wurden für die Evaluation jeweils die gleichen Werte in x- und y-Richtung gewählt. Eine unterschiedliche Auflösung r_x und r_y würde zum Beispiel Sinn ergeben, wenn einzig Daten weit entfernter Objekte vorliegen, da in diesem Falle die quadratisch wachsende Unsicherheit in y-Richtung eine größere Verteilung der Punkte als die linear wachsende Unsicherheit in x-Richtung hervorrufen würde. Nicht-quadratische Rasterkarten mit $s_\mathrm{x} \neq s_\mathrm{y}$ würden wiederum sinnvoll sein, wenn die Objekte hauptsächlich in eine Richtung ausgerichtet wären. Da im verwendeten Datensatz sowohl die Entfernung als auch die Orientierung der Objekte verschiedenste Werte annehmen, wurden dementsprechend $r_\mathrm{x} = r_\mathrm{y} = r$ und $s_\mathrm{x} = s_\mathrm{y} = s$ gewählt.

In Abbildung 8.4 sind die Ergebnisse verschiedener Auflösungen zwischen 2,5 cm und 40 cm anhand der in Abschnitt 6.5 vorgestellten Metriken dargestellt. Die Werte resultieren aus den Mittelwerten über jeweils alle im nächsten Abschnitt vorgestellten Box-Kodierungen, wobei für die AOS nur die Kodierungen verwendet werden, welche eindeutig innerhalb von 2π sind.

Betrachtet man beide Metriken ergeben sich für Autos in etwa gleich gute Ergebnisse für Auflösungen von 5 cm und 10 cm. Bei gleicher Größe der Rasterkarten, führt eine Halbierung der Auflösung allerdings zu einer Vervierfachung der Anzahl Zellen und somit auch etwa zu einer Vervierfachung der

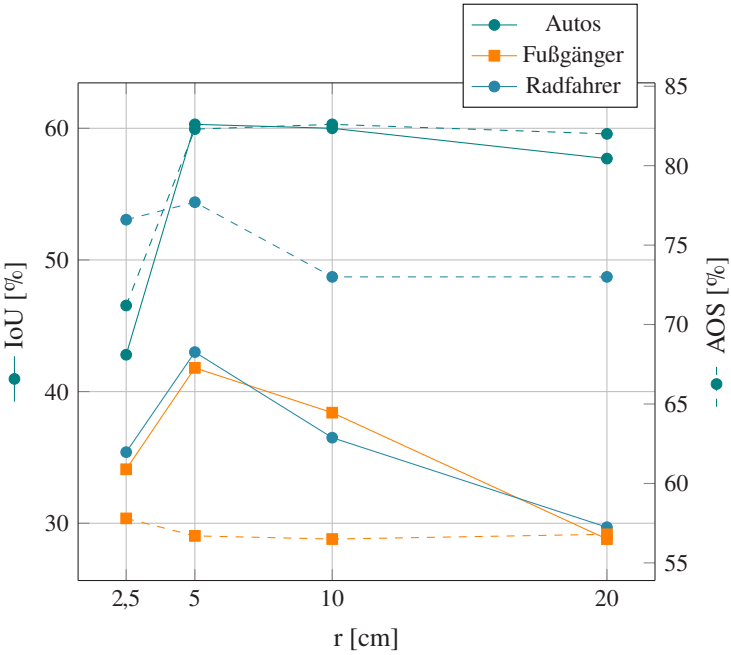


Abbildung 8.4: Evaluation verschiedener Rasterkarten-Auflösungen für die Objektklassen Auto, Fußgänger und Radfahrer.

Laufzeit. Deshalb wird die Auflösung für die Rasterkarten-Patches für Autos zu $r = 10\,\text{cm}$ gewählt.

Für Fußgänger ergibt sich für die IoU ein eindeutiges Maximum bei 5 cm, wohingegen die Orientierung über alle Auflösungen nahezu konstant bleibt. Dies liegt daran, dass die zu Fußgängern gehörenden Boxen in der Rasterkarte annähernd gleiche Breite und Höhe besitzen und so eine Orientierungsbestimmung unabhängig von der Auflösung erheblich erschwert wird. Da auch für Radfahrer ein eindeutiges Maximum sowohl für die IoU als auch für die AOS bei einer Auflösung von fünf Zentimetern liegt, wird eine Auflösung der Rasterkarten für Fußgänger und Radfahrer von $r = 5\,\text{cm}$ gewählt.

Die optimale Größe der Rasterkarten ergibt sich durch einen Kompromiss zwischen der Menge vorhandener Trainingsdaten und der Laufzeit. Da der

jeweilige Patch nur sinnvoll für das Training genutzt werden kann, wenn sich sowohl die Ground Truth Box als auch der Großteil der zum Objekt gehörenden Punkte innerhalb der vorhandenen Rasterkartenzellen befindet, sollte dieser nicht zu klein gewählt werden. Eine zu große Rasterkarte hingegen resultiert in vielen unbelegten Zellen, welche keinen Informationsgewinn darstellen, aber die Laufzeit erhöhen.

Wählt man für Autos eine Größe von $s = 12{,}8\,\text{m}$ existieren bei nur etwa 3 % der Trainingsdaten einzelne belegte Zellen, welche außerhalb dieses Ausschnitts liegen, halbiert man diese Größe sind es hingegen schon 34 %. Auf die gleiche Weise ergibt sich die optimale Patch Größe für Fußgänger und Radfahrer zu $s = 6{,}4\,\text{m}$. Daraus resultieren Rasterkartenausschnitte mit einer Bildgröße von $128\,\text{px} \times 128\,\text{px}$ für alle drei Objektklassen.

8.3.2 Validierung der Box-Kodierungen

Wie in Abschnitt 6.4 beschrieben werden die Ground Truth Daten aus den Trainingsdaten des KITTI Objekterkennungsdatensatzes erstellt. Dabei findet die bereits in Abschnitt 8.2.1 erwähnte Aufteilung in Trainings- und Validierungsdaten statt. Außerdem sollen verschiedene Aufsplittungen von Datensätzen evaluiert werden, welche sich an den bereits in Abschnitt 8.1 beschriebenen Schwierigkeitsstufen orientieren. Die einfachen Daten enthalten einzig Objekte, die vollständig sichtbar sind, und an Bildrändern maximal zu 15 % abgeschnitten sind. Beim Datensatz mit der Schwierigkeit Moderat sind alle vollständig sichtbaren und teilweise verdeckten Objekte mit einer maximalen Trunkierung von 30 % enthalten. Auch die Validierungsdaten werden für diese Schwierigkeit erstellt. Die schwierigen Daten umfassen alle Objekte bis hin zu einem Verdeckungsgrad von „schwer zu sehen“ und einer Trunkierung von 50 %. Datensätze mit höherer Schwierigkeit enthalten dementsprechend mehr Daten, sorgen aber auch für eine schlechtere Konvergenz des CNNs aufgrund von unvollständigen Rasterkarten verdeckter Fahrzeuge.

Auf die im KITTI-Benchmark zusätzliche Filterung anhand der minimalen 2D-Box Höhe wurde hier verzichtet um eine größere Menge Trainingsdaten

zur Verfügung zu haben und das Netz auch für weiter als 50 m[1] entfernte Objekte zu trainieren. Diese werden zwar bei der Evaluation in KITTI nicht beachtet, sind aber für die spätere Anwendung im automatisierten Fahrzeug ebenso wichtig.

Mit der in Abschnitt 6.4 beschriebenen Datenaugmentation kann die Menge der verfügbaren Trainingsdaten noch weiter erhöht werden. Deshalb wird für jede Schwierigkeitsaufteilung ebenfalls überprüft, ob diese Augmentation zu besseren Resultaten führt („+Aug."). Anhand der IoU- und AOS-Metrik wird für alle drei Objektklassen untersucht, welche Kombination aus Box-Kodierung und Datensatz die besten Ergebnisse liefert. Dabei ist zu beachten, dass die Box-Kodierungen B3 und B5 die Orientierung nur bis zu einer additiven Konstante von $\pm\pi$ kennen und dadurch bei einem über alle Orientierungswinkel gleich verteilten Datensatz nur einen maximalen Wert der AOS von 50 % erreichen können.

		E	**E+Aug.**	**M**	**M+Aug.**	**S**	**S+Aug.**
B1	IoU	52,9	56,4	55,6	57,4	53,7	52,9
	AOS	79,2	78,0	82,1	81,5	82,3	80,3
B2	IoU	58,8	58,7	58,8	**62,6**	59,7	59,3
	AOS	82,1	79,6	**84,3**	**84,0**	**85,2**	83,4
B3	IoU	61,4	60,3	59,2	58,2	59,5	**61,5**
	AOS	47,1	52,5	48,3	45,9	45,1	45,4
B4	IoU	59,5	59,4	57,9	**63,9**	59,8	58,0
	AOS	82,1	77,5	82,6	82,4	82,1	81,3
B5	IoU	43,2	45,3	46,7	41,6	43,8	45,0
	AOS	45,5	51,0	45,0	46,0	47,3	48,3

Tabelle 8.2: Mittlere IoUs für die Klasse Auto auf dem Validierungsdatensatz für alle fünf Box-Kodierungen und das Training auf den **E**infachen, **M**oderaten oder **S**chwierigen Daten ohne oder mit zusätzlicher **Aug**mentation. Die besten drei Ergebnisse für IoU und AOS sind jeweils fett dargestellt.

[1] Die angegebene maximale Entfernung ergibt sich aus der Rekonstruktionsfunktion aus Gleichung 3.11, der minimal geforderten Bounding Box Höhe von 25 px und der durchschnittlichen Objekthöhe und kann dementsprechend je nach Objekt etwas von diesem Wert abweichen

Für das Training des CNNs zur Autodetektion wird für alle Kombinationen aus Box-Kodierung und Datensatz eine Batch-Größe von 128 und eine Anzahl Epochen von 160 gewählt. Tabelle 8.2 zeigt die nach der letzten Epoche resultierenden mittleren IoU- und AOS-Werte. Für Autos ergeben sich mit den Box-Kodierungen B2, B3 und B4 über alle Datensätze hinweg, hohe IoU-Werte. Die Orientierung wird mit Kodierung B2 am genauesten geschätzt. Trotz der mit über 14 000 Rasterkarten hohen Menge an verfügbaren Trainingsdaten ergeben sich teilweise weitere Verbesserungen durch die Augmentation. Insgesamt die besten Resultate liefert ein Training mit Kodierung B2, also eine Repräsentation der Orientierung als $\cos(\theta)$ und $\sin(\theta)$, der moderate Datensatz und eine zusätzliche Augmentation von Daten.

		E	**E+Aug.**	**M**	**M+Aug.**	**S**	**S+Aug.**
B1	IoU	41,2	**42,4**	36,8	38,7	34,8	37,8
	AOS	56,7	**61,0**	56,1	57,2	54,2	57,7
B2	IoU	40,6	41,2	34,1	38,2	34,2	35,1
	AOS	59,7	**62,8**	58,0	58,1	59,1	**61,0**
B3	IoU	39,4	**45,2**	38,6	37,7	35,1	39,0
	AOS	52,4	54,2	52,3	53,9	51,9	56,1
B4	IoU	**42,4**	**43,0**	38,7	30,3	34,6	39,4
	AOS	54,6	59,2	55,1	52,2	51,1	58,0
B5	IoU	42,3	40,8	37,8	40,0	36,9	37,4
	AOS	52,6	54,7	50,9	50,6	51,4	54,8

Tabelle 8.3: Mittlere IoUs für die Klasse Fußgänger auf dem Validierungsdatensatz für alle fünf Box-Kodierungen und das Training auf den **E**infachen, **M**oderaten oder **S**chwierigen Daten ohne oder mit zusätzlicher **Aug**mentation. Die besten drei Ergebnisse für IoU und AOS sind jeweils fett dargestellt.

Für das Training der Objektklasse Fußgänger wird eine Batch-Größe von 64 und eine Anzahl Epochen von 400 gewählt. Die in Tabelle 8.3 dargestellten Werte zeigen, dass sich für Fußgänger über alle Kodierungen IoU-Ergebnisse in ähnlichen Bereichen herausbilden. Das liegt vor allem daran, dass die Boxen annähernd quadratisch sind und eine fehlerhafte Orientierungsschätzung so einen geringen Einfluss auf die IoU hat. Auch werden im Durchschnitt die besten Ergebnisse auf dem einfachen Datensatz erzielt. Dies ist auf die sowie-

so schon wenigen zum Fußgänger gehörenden Rasterzellen zurückzuführen, welche bei einer teilweisen Verdeckung nochmals reduziert werden und so die Schätzung der Dimensionen und Orientierung zusätzlich erschweren. Da die Validierung allerdings auf Daten mit teilweise verdeckten Fußgänger stattfindet, ist dennoch die Generalisierbarkeit gewährleistet. Eine Augmentation liefert auch hier vor allem bei dem einfachen Datensatz mit ansonsten nur knapp über 2000 Trainingsrasterkarten bessere Ergebnisse. Insgesamt schneidet Kodierung B1, welche die Orientierung θ direkt schätzt, auf dem einfachen Datensatz mit augmentierten Daten am besten ab.

		E	**E+Aug.**	**M**	**M+Aug.**	**S**	**S+Aug.**
B1	IoU	37,7	33,4	33,9	31,9	33,7	33,7
	AOS	**82,4**	74,7	80,3	69,8	81,0	72,3
B2	IoU	41,8	**45,4**	40,4	39,8	37,5	32,6
	AOS	78,3	70,5	82,7	71,6	82,0	76,3
B3	IoU	43,5	44,8	41,3	41,1	40,2	44,5
	AOS	49,3	54,4	49,8	52,3	53,2	50,9
B4	IoU	42,8	39,9	**46,7**	**45,1**	40,3	40,3
	AOS	**83,6**	76,5	**83,3**	73,7	**82,4**	75,9
B5	IoU	32,5	31,8	23,8	33,1	33,6	31,7
	AOS	49,9	49,1	47,5	44,7	47,6	47,4

Tabelle 8.4: Mittlere IoUs für die Klasse Radfahrer auf dem Validierungsdatensatz für alle fünf Box-Kodierungen und das Training auf den **E**infachen, **M**oderaten oder **S**chwierigen Daten ohne oder mit zusätzlicher **Aug**mentation. Die besten drei Ergebnisse für IoU und AOS sind jeweils fett dargestellt.

Für das Training des CNNs zur Radfahrerdetektion wird eine Batch-Größe von 32 und eine Anzahl Epochen von 800 gewählt. Die besten der in Tabelle 8.4 dargelegten Ergebnisse verteilen sich über fast alle Box-Kodierungen und Datensätze. Dies ist auf die geringe Menge an Trainingsdaten zurück zu führen, welche selbst mit Augmentation nur knapp 1000 beträgt. Dennoch ist zu erkennen, dass insgesamt Kodierung B4, welche die Orientierung durch $\cos(2\theta)$, $\sin(2\theta)$ und einen Richtungsparameter d repräsentiert, auf dem moderaten Datensatz die besten Ergebnisse liefert.

8.3.3 3D Bounding Box Evaluation

Für die in Abschnitt 8.3.2 ermittelten vielversprechendsten Netze pro Objektklasse soll in diesem Abschnitt die Genauigkeit der resultierenden 3D Bounding Boxen auf den Validierungsdaten überprüft werden. Dafür wird das schon aus Abschnitt 8.2.2 bekannte Diagramm verwendet, welches die durchschnittliche Genauigkeit der 3D-Boxen AP_{3D} über verschiedenen IoU-Schwellwerten aufträgt. Dieses ist in Abbildung 8.5 dargestellt.

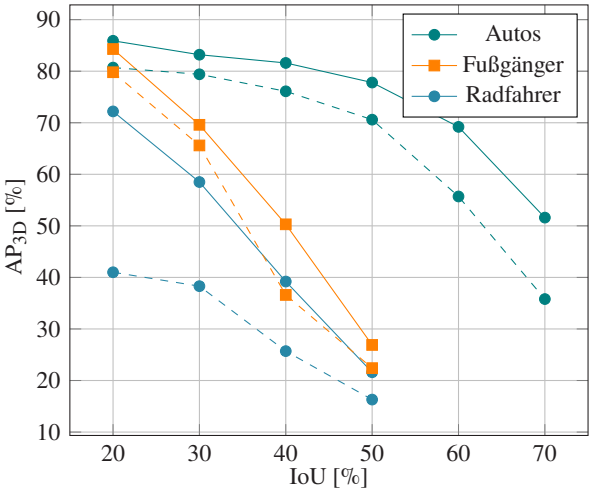


Abbildung 8.5: Durchschnittliche Genauigkeit auf den Validierungsdaten in Abhängigkeit des IoU-Schwellwertes. Zum Vergleich sind die Ergebnisse aus Abschnitt 8.2.2 gestrichelt dargestellt.

Für alle drei Objektklassen, vor allem für Radfahrer, ergeben sich deutlich höhere Genauigkeiten über alle IoU-Werte. Dies lässt sich einerseits durch die bessere Schätzung des nicht gaußförmig verteilten Tiefenfehlers erklären. Andererseits konnte auch eine weitere Einschränkung des Ansatzes aus Kapitel 5 behoben werden, welche dazu geführt hat, dass die geschätzten Boxen stärker zu einer Orientierung in Sichtstrahlrichtung tendieren. Dies wirkt sich besonders deutlich auf die im Vergleich zur Länge sehr schmalen Radfahrer aus, da in diesem Falle ein Fehler in der Orientierung zu einer sehr viel geringeren IoU führt.

Bis zu einer IoU von 50 % werden für Autos Genauigkeiten von über 80 % erzielt. Auch für Fußgänger und Radfahrer ergeben sich für geringere IoU-Werte Genauigkeiten von über 70 %. Da auch die 2D Bounding Boxen auf den Validierungsdaten Ergebnisse in diesem Bereich erzielt haben, kann davon ausgegangen werden, dass für jede 2D-Detektion eine 3D Bounding Box gefunden wurde, deren Genauigkeit für die Anwendung im automatisierten Fahrzeug ausreichend ist.

Durch die geringe Größe der Rasterkartenausschnitte von nur 128 px × 128 px und das sehr flache CNN, beträgt die Inferenzzeit des Netzes weniger als 1 ms. Die Erstellung dieser Rasterkarten aus den Objektclustern benötigt ebenfalls etwa 1 ms. Da diese beiden Schritte parallel für alle Objekte ausgeführt werden können, beträgt die gesamte Laufzeit mit den bereits in Abbildung 8.3 dargestellten Rechenzeiten für semantische Segmentierung, Disparitätsschätzung und Clustering nur 60 ms. Wie auch später in Abschnitt 8.5.1 zu sehen, ergibt sich mit dieser Laufzeit der schnellste stereobasierte 3D-Objektdetektor auf dem KITTI-Datensatz.

8.4 Evaluation der monokularen 3D-Objektdetektion

Für die in Kapitel 7 vorgestellte monokulare 3D-Objekterkennung kommt ebenfalls die in Abschnitt 8.2.1 beschriebene Aufteilung in Trainings- und Validierungsdaten zum Einsatz. Für den Input des Lifting-Netzes wird die Semantik und Tiefenschätzung innerhalb der jeweiligen RoI konkateniert und auf eine einheitliche Größe skaliert. Für die Skalierung wurden dabei verschiedene Werte zwischen 48 px × 48 px und 128 px × 128 px analysiert. Es hat sich gezeigt, dass eine Eingangsgröße von 64 px × 64 px die besten Ergebnisse liefert, da diese die Merkmale selbst für naheliegende Objekte schon ausreichend gut darstellt, wohingegen dies für kleinere Größen nicht der Fall ist. Höhere Werte führen zu einer etwas erhöhten Laufzeit und verbessern die Leistung des CNNs ebenfalls nicht, da vor allem weiter entfernte Objekte durch die Hochskalierung der Eingangsdaten, weniger präzise Merkmale besitzen.

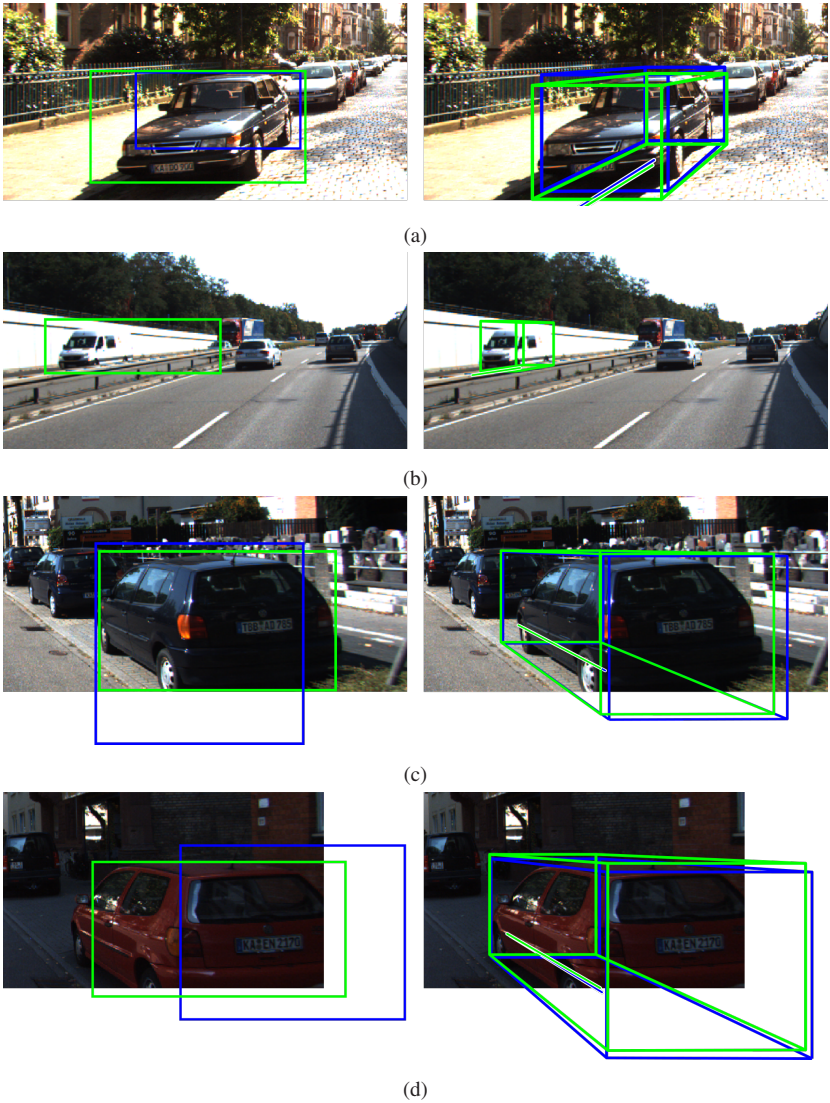
8.4.1 Untersuchung der Verschiebungsinvarianz

Mit der in Abschnitt 7.3 vorgestellten Datenaugmentation soll eine Überanpassung an die 2D-Detektionen vermieden werden und die Verschiebungsinvarianz des Lifting-Netzes gestärkt werden um es weniger anfällig gegen Fehler in den 2D-Detektionen zu machen. Je nach Schwellwert der *Non-maximum Suppression* können während der Inferenz mehrere 2D Bounding Boxen zu einem Objekt gehören. In der qualitativen Auswertung werden diese genutzt um die Verschiebungsinvarianz des Lifting-Netzes zu analysieren.

In Abbildung 8.6a sehen wir, dass die aus unterschiedlichen 2D-Boxen bestimmten 3D Bounding Boxen eine hohe IoU aufweisen. Ihre geschätzten Orientierungen, Positionen und Abmessungen weichen nur minimal voneinander ab. Abbildung 8.6b zeigt, dass das Netz sogar für sehr ungenaue 2D Bounding Boxen vielversprechende 3D Bounding Boxen liefert. Aus Abbildung 8.6c und d kann abgeleitet werden, dass die Verschiebungsinvarianz auch für am Bildrand abgeschnittene Objekte gilt. Für Objekte, die teilweise außerhalb der Begrenzung liegen, ist es unmöglich, eine perfekte 2D Bounding Box oder Instanzmaske zu erhalten. Daher kann das in [WK19] vorgeschlagene „2D-3D Bounding-Box Consistency Constraint“ in diesen Situationen nicht genutzt werden. Das in dieser Arbeit vorgestellte Netz hingegen kann gute 3D Bounding Boxen selbst für diese abgeschnittenen Objekte schätzen. Daraus kann geschlossen werden, dass die verwendete Methode der Datenaugmentation die Verschiebungsinvarianz und die Generalisierungsfähigkeit des Lifting-Netzes verbessert.

8.4.2 3D Bounding Box Evaluation

Die Genauigkeit der 3D-Box Schätzung in Abhängigkeit verschiedener IoU-Schwellwerte ist in Abbildung 8.7 dargestellt. Für die Objektklasse Auto können mit einer durchschnittlichen Genauigkeit von über 60 % bei einer IoU von 50 % für einen monokularen Ansatz sehr gute Ergebnisse erzielt werden. Aufgrund der geringen Menge an Trainingsdaten liefert die 3D-Box Schätzung für Fußgänger und Radfahrer keine Resultate in diesen Bereichen. Dennoch können über 40 % der Fußgänger und 20 % der Radfahrer mit einem Fehler von maximal 0,5 m in jede Richtung detektiert werden.



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

Abbildung 8.6: Qualitative Evaluation der Verschiebungsinvarianz. Die Bilder links zeigen außer für (b) verschiedene 2D Bounding Boxen für dasselbe Objekt (grün und blau), in den rechten Bildern sind die dazugehörigen 3D Bounding Boxen mit ihren Orientierungen dargestellt.

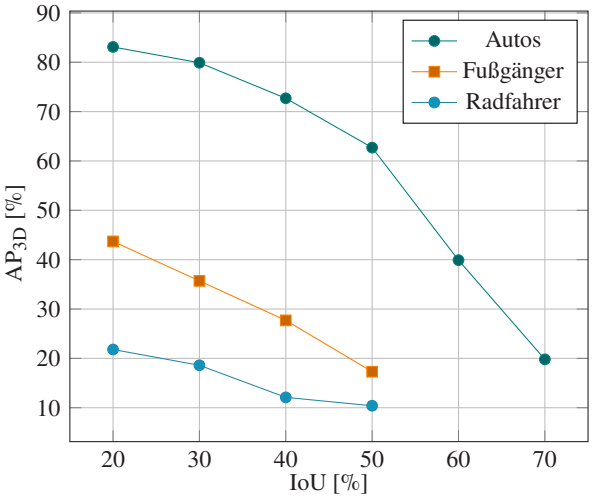


Abbildung 8.7: Durchschnittliche Genauigkeit auf den Validierungsdaten in Abhängigkeit des IoU-Schwellwertes.

8.5 Evaluation und Vergleich mit verwandten Arbeiten auf dem KITTI Testdatensatz

In diesem Abschnitt werden die Ergebnisse aller drei Ansätze auf den Testdaten des KITTI Objekterkennungsbenchmarks gezeigt und mit den in Kapitel 2 vorgestellten Arbeiten verglichen. Da die Ground Truth dieses Testdatensatzes nicht öffentlich zur Verfügung steht, findet die Evaluation durch das Hochladen aller Detektionen auf der KITTI-Webseite statt. Dadurch ist zwar keine so ausführliche Evaluation wie auf den Validierungsdaten möglich, die Ergebnisse zwischen verschiedenen Arbeiten sind aber umso besser vergleichbar. Neben der Evaluation der 2D- und 3D-Detektionen wird außerdem auch die Genauigkeit der 3D-Ergebnisse in Vogelperspektive (engl. Bird's Eye View, BEV) angegeben um Ansätze vergleichen zu können, welche ausschließlich auf Rasterkarten basieren. Für alle drei Benchmarks wird für eine korrekte Detektion eine IoU von mindestens 70 % für Autos und 50 % für Fußgänger und Radfahrer gefordert. Ein vierter Benchmark evaluiert die Orientierungsgenauigkeit anhand der bereits vorgestellten AOS-Metrik. Während im Abschnitt 8.3

allerdings die Genauigkeit der globalen Orientierung untersucht wurde, findet auf dem KITTI-Benchmark eine Evaluation der lokalen Orientierung statt. Die Ergebnisse aller vier Benchmarks sind außerdem in die bereits erläuterten Schwierigkeitsstufen Einfach, Moderat und Schwierig aufgeteilt.

Objektklasse	Einfach	Moderat	Schwierig
Auto	78,85 %	54,19 %	43,23 %
Fußgänger	55,56 %	39,83 %	35,18 %
Radfahrer	33,64 %	22,90 %	19,87 %

Tabelle 8.5: Durchschnittliche Genauigkeiten der 2D-Detektionen auf dem Testdatensatz.

Für die 2D-Objektdetektionen aller in dieser Arbeit vorgestellten Ansätze wurde das in Abschnitt 5.1.1 vorgestellte CNN verwendet, dessen Ergebnisse auf dem Testdatensatz in Tabelle 8.5 dargestellt sind. Durch die geringe Anzahl verfügbarer Trainingsdaten für die Objektklassen Fußgänger und Radfahrer ergibt sich speziell für diese eine größere Differenz in der durchschnittlichen Genauigkeit im Vergleich zum Validierungsdatensatz.

8.5.1 Binokulare Objektdetektion

Die Resultate des in Kapitel 5 vorgestellten Ansatzes auf dem Testdatensatz sind in Tabelle 8.6 dargestellt.

Trotz der zusätzlichen Optimierung der y-Position und Höhe der Objekte ergibt sich durch die Rasterkartenbasierte Detektion ein gewisser Unterschied zwischen den 3D- und BEV-Ergebnissen. Nichtsdestotrotz ergibt sich mit ca. 30 % durchschnittlicher Genauigkeit für Autos auf dem einfachen Datensatz ein gutes Ergebnis, welches im Bereich der Resultate auf dem Validierungsdatensatz liegt. Die Anzahl korrekt lokalisierter Fußgänger und Radfahrer kann auch auf dem Testdatensatz nicht ganz mit diesen Ergebnissen mithalten. Die Approximation des nicht gaußförmig verteilten Tiefenfehlers, welcher zu Boxschätzungen führt, die tendenziell etwas hinter der wahren Objektposition liegen, wirkt sich speziell auf diese kleinen Objekte aus. Da die Orientierung mit diesem Ansatz nur innerhalb von π eindeutig geschätzt werden kann, ergibt sich bei einem über alle Orientierungswinkel gleich verteilten Datensatz ein

Objektklasse	Benchmark	Einfach	Moderat	Schwierig
Auto	3D-Detektion	29,90 %	23,28 %	18,96 %
	BEV	58,81 %	46,82 %	38,38 %
	Orientierung	25,58 %	21,41 %	17,52 %
Fußgänger	3D-Detektion	3,28 %	2,45 %	2,35 %
	BEV	4,72 %	3,65 %	3,00 %
	Orientierung	21,41 %	15,34 %	13,23 %
Radfahrer	3D-Detektion	5,29 %	3,37 %	2,57 %
	BEV	7,03 %	4,10 %	3,88 %
	Orientierung	5,46 %	3,88 %	3,54 %

Tabelle 8.6: Ergebnisse der Rasterkartenbasierten Objektdetektion auf dem Testdatensatz.

maximaler Wert der AOS von 50 %. Da unbeobachtete Zellen weniger stark in die Optimierungsfunktion einfließen als freie Zellen und insbesondere an vertikalen Objektkanten besonders viele Punkte detektiert werden, ergibt sich aber eine gewisse Tendenz dazu, dass die geschätzten Boxen eine Orientierung in Sichtstrahlrichtung besitzen. Deshalb können für alle drei Objektklassen auch die 50 % Orientierungsgenauigkeit nicht erreicht werden.

Die genannten Einschränkungen werden mit der Bestimmung der Boxparameter durch das in Kapitel 6 vorgestellte CNN behoben. Die Ergebnisse dieses Ansatzes auf dem Testdatensatz sind in Tabelle 8.7 wiedergegeben.

Es zeigt sich eine deutliche Verbesserung für alle Objekttypen und über alle Benchmarks. Insbesondere für Fußgänger und Radfahrer ergibt sich eine verfünf- bzw. verdreifachte Genauigkeit der 3D-Detektionen. Auch die AOS ist durch die Wahl geeigneter Box-Kodierungen auf das doppelte bis vierfache gestiegen.

Wie gut diese Ergebnisse im Vergleich zu anderen binokularen Objektdetektoren sind, zeigt sich in Abbildung 8.8. Hier findet sich ein Vergleich aller auf KITTI veröffentlichten Ansätze anhand der Laufzeiten und der durchschnittlichen Genauigkeit der 3D-Detektionen für Autos auf dem Testdatensatz. Die aufgelisteten Arbeiten wurden bereits in Abschnitt 2.3 genauer beschrieben. Da

Objektklasse	Benchmark	Einfach	Moderat	Schwierig
Auto	3D-Detektion	45,79 %	38,76 % (+ 15,48 %)	30,00 %
	BEV	69,14 %	59,00 % (+ 12,18 %)	45,49 %
	Orientierung	44,06 %	36,31 % (+ 14,90 %)	27,32 %
Fußgänger	3D-Detektion	16,23 %	11,41 % (+ 8,96 %)	10,12 %
	BEV	19,92 %	14,22 % (+ 10,57 %)	12,83 %
	Orientierung	40,81 %	28,75 % (+ 13,41 %)	25,13 %
Radfahrer	3D-Detektion	18,31 %	12,99 % (+ 9,62 %)	10,63 %
	BEV	20,59 %	13,92 % (+ 9,82 %)	12,74 %
	Orientierung	23,91 %	16,18 % (+ 12,30 %)	14,23 %

Tabelle 8.7: Ergebnisse der Lernbasierten Boxschätzung mithilfe von Rasterkarten-Ausschnitten auf dem Testdatensatz und Verbesserung in Prozentpunkten gegenüber der Rasterkartenbasierten Objektdetektion für die moderate Schwierigkeit.

auch die auf der KITTI-Webseite verfügbare Rangliste die Ergebnisse anhand des Moderaten Datensatzes sortiert, sind die Resultate für diesen dargestellt.

Legt man eine für das automatisierte Fahren durchaus übliche Bildfrequenz von 10 Hz zu Grunde, ist neben den in dieser Arbeit vorgestellten Ansätzen einzig das TLNet mit einer Inferenzzeit von exakt 100 ms echtzeitfähig. Die Ergebnisse sind jedoch um einiges schlechter. Insgesamt lässt sich sagen, dass mit dem in Kapitel 6 vorgestellten Ansatz Genauigkeiten erzielt werden konnten, die im Bereich anderer Arbeiten mit fünf- bis sechsfacher Laufzeit liegen. Einzig drei Netze mit etwa zehnfacher Inferenzzeit erzielen nochmals bessere Ergebnisse.

Für Fußgänger und Radfahrer erzielt der jeweils beste Ansatz etwa doppelt so hohe AP_{3D}-Werte bei Laufzeiten die ebenfalls um ein Vielfaches höher sind. Für Ersteres ergibt sich die höchste Genauigkeit mit Disp R-CNN zu etwa 25 %, für Radfahrer erreicht CG-Stereo mit etwa 30 % die besten Resultate.

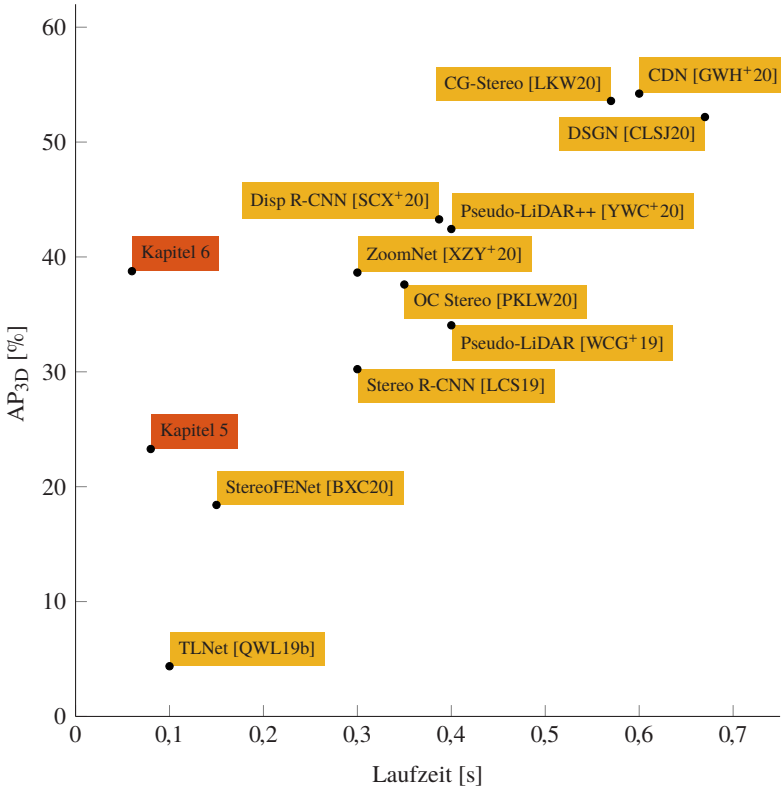


Abbildung 8.8: Vergleich der Genauigkeiten und Laufzeiten aller der KITTI-Webseite veröffentlichten binokularen 3D-Objektdetektoren für die Objektklasse Auto.

8.5.2 Monokulare Objektdetektion

Für die in Kapitel 7 vorgestellte monokulare 3D-Objektdetektion sind die Ergebnisse in Tabelle 8.8 dargestellt.

Verständlicherweise ergeben sich durch die ungenauere Tiefenschätzung aus nur einem Bild geringere Genauigkeiten der 3D- und BEV-Detektionen gegenüber den binokularen Methoden. Durch die Nutzung der Bildinformation kann jedoch speziell für Autos eine deutlich höhere Genauigkeit der Orientierungs-

Objektklasse	Benchmark	Einfach	Moderat	Schwierig
Auto	3D-Detektion	14,72 %	9,28 %	7,28 %
	BEV	23,60 %	14,89 %	11,87 %
	Orientierung	78,33 %	53,61 %	42,64 %
Fußgänger	3D-Detektion	10,08 %	6,33 %	5,01 %
	BEV	11,59 %	8,31 %	7,18 %
	Orientierung	33,58 %	24,30 %	22,00 %
Radfahrer	3D-Detektion	8,66 %	3,54 %	2,95 %
	BEV	9,80 %	4,20 %	3,60 %
	Orientierung	21,69 %	12,70 %	12,33 %

Tabelle 8.8: Ergebnisse der monokularen 3D-Objektdetektion auf dem Testdatensatz.

schätzung erreicht werden. Ein Vergleich mit den in Abschnitt 2.2 beschriebenen Ansätzen zur monokularen 3D-Objektdetektion ist für die Objektklasse Auto in Abbildung 8.9 zu sehen. Da die Laufzeiten sehr stark variieren, wurde hierfür eine logarithmische Skala gewählt.

Das in dieser Arbeit vorgestellte CNN liefert Ergebnisse im Bereich der besten monokularen 3D-Detektoren und bleibt dabei mit etwa 100 ms Inferenzzeit echtzeitfähig. Auf dem einfachen Schwierigkeitsgrad ist es sowohl für Autos als auch für Fußgänger und Radfahrer der beste echtzeitfähige monokulare Ansatz auf dem KITTI Testdatensatz.

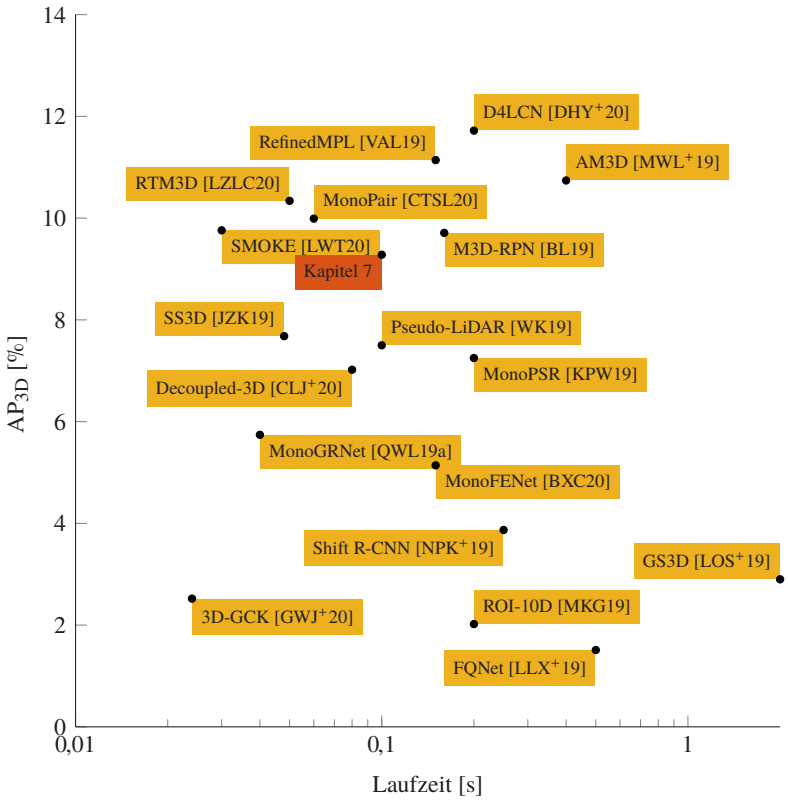


Abbildung 8.9: Vergleich der Genauigkeiten und Laufzeiten aller auf der KITTI-Webseite veröffentlichten monokularen 3D-Objektdetektoren für die Objektklasse Auto.

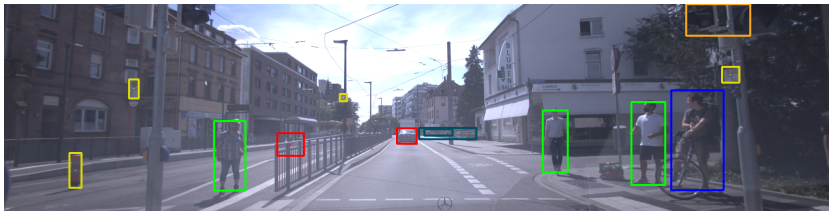
8.6 Qualitative Evaluation auf dem Testfahrzeug BerthaOne

In diesem Abschnitt werden die Ergebnisse der Objektdetektion auf dem bereits in Kapitel 1 vorgestellten Forschungsfahrzeug BerthaOne gezeigt. Da in dem Fahrzeug ein binokulares Kamerasystem vorhanden ist, werden die Re-

sultate des Ansatzes aus Kapitel 6 dargelegt, welcher in der vorangegangenen Evaluation auf dem KITTI Datensatz die besten Ergebnisse geliefert hat.

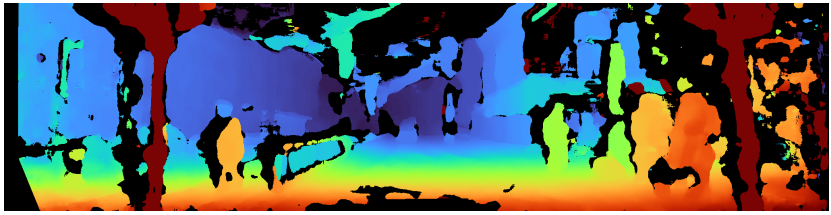
Die im Fahrzeug verwendeten Kameras sind FLIR Blackfly S mit 8,9 MP, wobei für die Linke eine Farbkamera und für die Rechte eine Graustufenkamera zum Einsatz kommt. Mit den verwendeten Objektiven ergeben sich Bildwinkel von über 130°. Dadurch können auch Objekte detektiert werden, welche sich im Nahbereich links und rechts neben der Fahrbahn befinden. Nachteil ist allerdings die damit verbundene niedrige Brennweite, welche für weit entfernte Objekte zu einer schlechten Entfernungsauflösung führt. Die Bilder werden mit einer Rate von 10 Hz aufgezeichnet. Aus Laufzeitgründen wird die Auflösung herunterskaliert, sodass sich nach der Rektifizierung Bildgrößen von 512 px × 2048 px ergeben. Das in Abschnitt 5.1.1 vorgestellte CNN zur semantischen Segmentierung und 2D-Objektdetektion benötigt für diese Auflösung etwa 50 ms auf der im Fahrzeug verfügbaren NVIDIA Titan V GPU. Das Stereo-Matching wird parallel dazu auf einer NVIDIA Titan X GPU in etwa 40 ms ausgeführt. Die auf diese Weise erzeugten Eingangsdaten für die Objektdetektion sind in Abbildung 8.10a-c dargestellt.

Neben der bereits bekannten Darstellung von Autos in rot, Fußgänger in grün und Radfahrern in blau detektiert das Netz unter anderem auch Verkehrszeichen in gelb und Ampeln in orange. Die dunkelgrüne Bounding Box in Abbildung 8.10a wird für Straßenbahnen verwendet, wobei es sich in diesem Bild um einen Fehldetektion handelt. Auch zu sehen ist eine fehlerhafte Bounding Box für ein Auto. Durch das auf der Semantik und Disparität basierende Clustering können solche Fehldetektionen effektiv heraus gefiltert werden, wie in Abbildung 8.10d zu sehen. Da keine Ground Truth zur Verfügung steht, werden die Ergebnisse der 3D-Objektdetektion qualitativ anhand der Projektionen der 3D Bounding Boxen ins linke Kamerabild evaluiert. Diese sind gemeinsam mit der geschätzten Orientierung in Abbildung 8.11a zu finden. Ebenfalls dargestellt ist die kumulierte Rasterkarte aller in der Szene befindlichen Objekte in Abbildung 8.11b. Da der auf Rasterkarten-Ausschnitten basierende Ansatz gewählt wurde, sind unbeobachtete Zellen nicht explizit dargestellt. Durch das gewählte Beispiel mit Objekten im Nahbereich des Fahrzeuges, lässt sich anhand der 3D-Box Projektion sehr gut die Genauigkeit der Objektschätzung beurteilen. Es zeigt sich, dass für alle in der Szene befindlichen Verkehrsteilnehmer sowohl die Positionen, als auch die Objektdimensionen und Orientierungen sehr genau bestimmt werden.

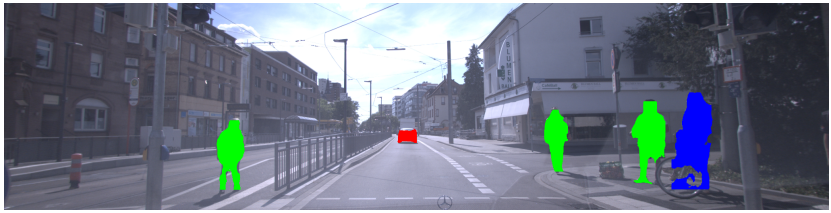


(a) 2D Bounding Boxen

(b) Semantik

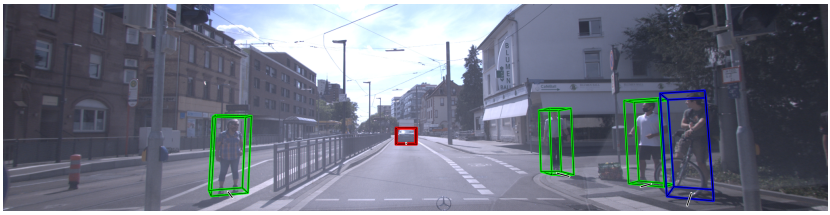


(c) Disparität

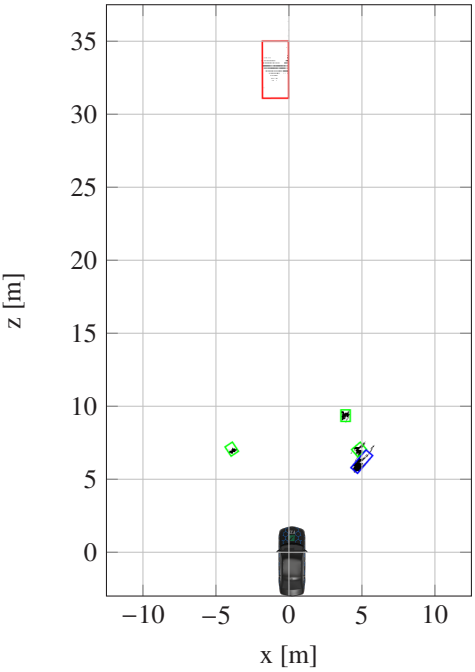


(d) Ergebnisse des Clusterings

Abbildung 8.10: Ergebnisse der 2D Bounding Box Schätzung, der semantischen Segmentierung, des Stereo-Matchings und des Clusterings auf dem Versuchsträger BerthaOne. In Abbildung a, b und d sind Autos in rot, Fußgänger in grün, Radfahrer in blau, Verkehrszeichen in gelb, Ampeln in orange, Fahrräder in dunkelblau und Straßenbahnen in dunkelgrün dargestellt. Die Disparität in Abbildung c verläuft von hohen Werten in dunkelrot hin zu niedrigen in dunkelblau.



(a) 3D Bounding Boxen



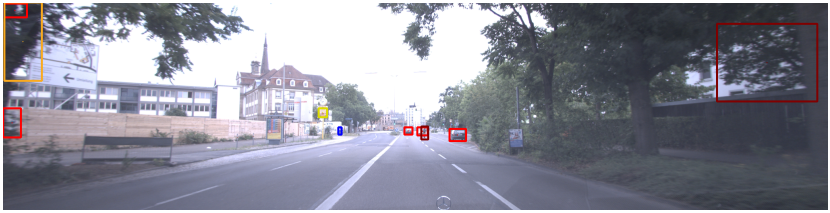
(b) Rasterkarte mit geschätzten Objektboxen

Abbildung 8.11: In das linke Kamerabild und in die Rasterkarte projizierte Ergebnisse der 3D Bounding Box Schätzung auf dem Forschungsfahrzeug BerthaOne für die Szene aus Abbildung 8.10. Dabei sind Autos in rot, Radfahrer in blau und Fußgänger in grün dargestellt.

Mit einem weiteren Beispiel soll gezeigt werden, bis zu welcher Entfernung eine ausreichend gute 3D-Objektschätzung möglich ist. Die 2D Bounding Boxen, die Semantik und die Disparität sind für dieses Beispiel in Abbildung 8.12 dargelegt.

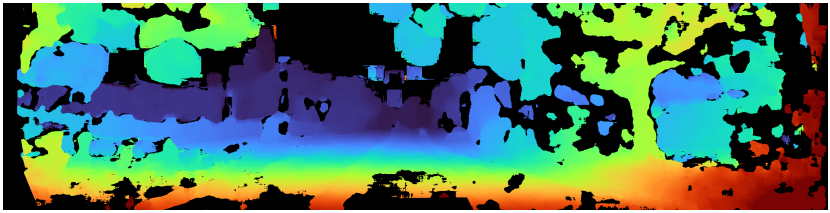
Neben einigen Fehldetektionen am Bildrand, resultieren 2D-Boxen für drei Autos, einen Radfahrer und einen Motorradfahrer. Die dazugehörigen ins Bild projizierten 3D Bounding Boxen und die Rasterkarte für alle in der Szene befindlichen dynamischen Objekte sind in Abbildung 8.13 zu sehen. Da nur wenige Punkte in die weit entfernten Rasterkartenzellen fallen, besitzen diese eine geringe Belegtheit und somit einen niedrigen Grauwert. Obwohl der Stereo-Matcher die Disparitäten auf $1/16\,\mathrm{px}$ genau schätzt, führen die Brennweite von $f = 850\,\mathrm{px}$ und die Basisbreite zwischen den Kameras von $b = 0{,}47\,\mathrm{m}$ dazu, dass bei einer Entfernung von beispielsweise 80 m nur noch eine Entfernungsauflösung von 1 m vorliegt. Dadurch ergibt sich in diesen Bereichen eine sehr dünn besetzte Rasterkarte. Dennoch können durch den gewählten lernbasierten Ansatz 3D Bounding Boxen bis zu einer Entfernung von 100 m geschätzt werden, deren Projektion in das linke Kamerabild auf eine hohe Genauigkeit schließen lässt. Auch die Position und Dimension des hinter dem Motorrad teilweise verdeckten Fahrzeuges, wird trotz der geringen Anzahl belegter Zellen noch ausreichend gut geschätzt.

Die Laufzeit für Clustering, Erzeugung der Rasterkarten-Ausschnitte und die Schätzung der Box-Parameter beträgt für die verwendete Auflösung von 1 MP etwa 40 ms auf der im Fahrzeug verbauten NVIDIA Titan X GPU. Dadurch ergibt sich mit den bereits genannten Rechenzeiten für Semantik und Stereo-Matching eine Gesamtlaufzeit von unter 100 ms und somit Echtzeitfähigkeit.



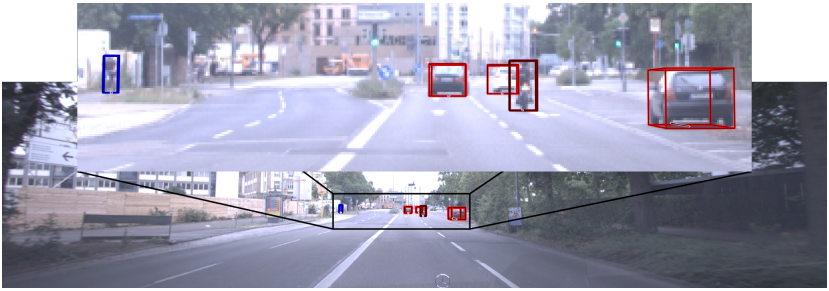
(a) 2D Bounding Boxen

(b) Semantik

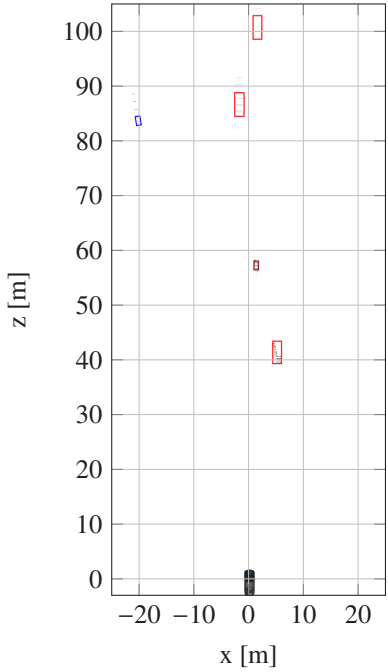


(c) Disparität

Abbildung 8.12: Ergebnisse der 2D Bounding Box Schätzung, der semantischen Segmentierung und des Stereo-Matchings auf dem Versuchsträger BerthaOne. In Abbildung a und b sind Autos in rot, Radfahrer in blau, Verkehrszeichen in gelb, Ampeln in orange und Motorradfahrer in dunkelrot dargestellt. Die Disparität in Abbildung c verläuft von hohen Werten in dunkelrot hin zu niedrigen in dunkelblau.



(a) 3D Bounding Boxen



(b) Rasterkarte mit geschätzten Objektboxen

Abbildung 8.13: In das linke Kamerabild und in die Rasterkarte projizierte Ergebnisse der 3D Bounding Box Schätzung auf dem Forschungsfahrzeug BerthaOne für die Szene aus Abbildung 8.12. Dabei sind Autos in rot, Radfahrer in blau und Motorradfahrer in dunkelrot dargestellt.

9 Zusammenfassung und Ausblick

Die vorliegende Arbeit befasst sich mit der echtzeitfähigen kamerabasierten 3D-Objekterkennung. Im Gegensatz zu existierenden Ansätzen liegt der Fokus dieser Arbeit nicht nur auf der Erkennung von Autos, sondern sowohl auf Kraftfahrzeugen als auch auf Fußgängern und Radfahrern. Auch die Echtzeitfähigkeit der vorgestellten Ansätze stellt ein essentielles Kriterium für die Anwendung in hochautomatisierten Fahrzeugen dar. Untersuchungen auf dem öffentlich verfügbaren KITTI-Datensatz [GLU12] zeigen, dass die in dieser Arbeit vorgestellten binokularen Ansätze und auch der monokulare 3D-Objektdetektor den Stand der Technik für echtzeitfähige kamerabasierte 3D-Objekterkennung übertreffen.

Da sich in einem Verkehrsszenario alle Objekte auf einer gemeinsamen Bodenoberfläche bewegen, wurde für die beiden binokularen Methoden die Rasterkarte als abstrahierte Darstellung gewählt. Durch die damit verbundene Dimensionsreduktion verringert sich die Komplexität der Boxschätzung und damit die Laufzeit. Durch die Verwendung von semantischer Information kann bereits im Bild zwischen Boden und anderen Bereichen unterschieden werden. Die in Kapitel 4 vorgestellte Bodenoberflächenschätzung robustifiziert diese Filterung und hilft außerdem die Höhe und Position der detektierten Objekte korrekt abzubilden. Auch kann sie zur Bestimmung befahrbarer Flächen genutzt werden, welche zu einem vollständigen Umfeldmodell für das automatisierte Fahrzeug beitragen.

In Kapitel 5 wurde beschrieben, wie mithilfe der Tiefenschätzung aus einem Stereobildpaar und semantischer Information Objektcluster im Bild gefunden werden, welche in die angesprochene Rasterkartendarstellung überführt werden. Für die Position, Orientierung und Dimension des jeweiligen Objektes wurde anschließend ein Optimierungsproblem aufgestellt, welche sich durch die verwendete Rasterkarten-Abstraktion sehr schnell lösen lässt. Da Rasterkarten normalerweise hauptsächlich für LiDAR-Sensoren genutzt werden, welche eine sehr genaue Tiefeninformation liefern, ist die Modellierung der Unsicher-

heit der Stereo-Tiefenschätzung ein wichtiger Bestandteil des vorgestellten Ansatzes. Durch die Berechnung von Konfidenzwerten während des Stereo-Matchings kann außerdem ein Konfidenzwert pro Objekt geschätzt werden, der für eine mögliche Fusion mit anderen Sensoren wichtig ist.

Mit den in Kapitel 6 vorgestellten Rasterkartenausschnitten wurde eine nochmals weiter vereinfachte Darstellung der Daten gefunden, die schnell zu verarbeiten und dennoch aussagekräftig ist. Durch die Verwendung eines CNNs zur Schätzung der Boxparameter in diesen Ausschnitten wurden nicht nur einige Einschränkungen des im vorherigen Kapitel beschriebenen Ansatzes behoben, sondern auch die Gesamtlaufzeit nochmals verringert. Mit einer Inferenzzeit des CNNs von weniger als 1 ms und einer Gesamtlaufzeit von etwa 60 ms ergibt sich so der schnellste stereobasierte 3D-Objektdetektor des KITTI-Benchmarks. Auch die Ergebnisse liegen im Bereich von anderen Arbeiten, welche eine bis zu sechsfache Rechenzeit benötigen. Experimente mit unserem Testfahrzeug BerthaOne haben gezeigt, dass mit einer höheren Auflösung als die auf KITTI verwendeten 0,4 MP, gute Objektdetektionen hinsichtlich der Position, Abmessungen und Orientierung auch für Objekte in 100 m Entfernung bestimmt werden können.

Da eine Tiefenschätzung aus nur einem Bild eine nochmals höhere Unsicherheit besitzt, basiert die in Kapitel 7 vorgestellte monokulare Objektdetektion nicht auf Rasterkarten, sondern versucht alle im Bild vorhandenen Informationen zu nutzen. Dafür wurden die semantische und räumliche Information innerhalb jeder 2D-RoI genutzt um mithilfe eines CNNs die 3D Bounding Box zu schätzen. Um ebenfalls eine Dimensionsreduktion zu erreichen, werden die Punktwolken im Bildformat angeordnet und können so mit einem 2D-CNN schneller verarbeitet werden. Durch die ebenfalls vorgestellte Methode der Datenaugmentation wird die Verschiebungsinvarianz und Generalisierungsfähigkeit des Netzes verbessert. Auf dem einfachen Schwierigkeitsgrad des KITTI-Benchmarks liefert das CNN sowohl für Autos als auch für Fußgänger und Radfahrer die besten Ergebnisse aller echtzeitfähigen monokularen Ansätze.

Da bei den beiden binokularen Ansätzen noch eine gewisse Differenz zwischen den BEV- und 3D-Ergebnissen besteht, wäre eine Erweiterungsmöglichkeit die Höheninformation pro Zelle in die Rasterkarte mit aufzunehmen und so die y-Information als zusätzlichen Parameter im Netz zu schätzen. Dies würde allerdings die Komplexität etwas erhöhen und könnte so zu höheren Rechen-

zeiten führen. Für die Anwendung in automatisierten Fahrzeugen ist außerdem die Genauigkeit entlang der Bodenoberfläche von größerer Bedeutung als eine genaue Schätzung der Objekthöhe.

Zukünftige Arbeiten im Bereich der monokularen 3D-Objektdetektion könnten alternative Ansätze zur Schätzung der Tiefe aus monokularen Bildern untersuchen. Obwohl sich diese in den letzten Jahren erheblich verbessert hat, ist die genaue Tiefenschätzung über große Entfernungen immer noch ein herausforderndes Problem. Auch könnten die Ergebnisse durch die gemeinsame Schätzung von Tiefe und Semantik in einem Multi-Task-Netz weiter verbessert werden. Durch die direkte Verwendung der in diesem Netz berechneten Feature Map für die 3D-Box-Schätzung könnte außerdem eine noch geringere Laufzeit erzielt werden.

Literaturverzeichnis

[Agg18] Aggarwal, Charu C.: *Neural Networks and Deep Learning: A Textbook*. Cham, Schweiz : Springer International Publishing, 2018. http://dx.doi.org/10.1007/978-3-319-94463-0. – ISBN 978–3–319–94463–0

[BBC+14] Bernini, Nicola; Bertozzi, Massimo; Castangia, Luca; Patander, Marco; Sabbatelli, Mario: Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey. In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. Qingdao, China, 2014, S. 873–878

[BBH03] Brown, Myron Z.; Burschka, Darius; Hager, Gregory D.: Advances in computational stereo. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25 (2003), Nr. 8, S. 993–1008

[BHL15] Beyer, Lucas; Hermans, Alexander; Leibe, Bastian: Biternion Nets: Continuous Head Pose Regression from Discrete Training Labels. In: Gall, Juergen (Hrsg.); Gehler, Peter (Hrsg.) ; Leibe, Bastian (Hrsg.): *German Conference on Pattern Recognition*. Aachen, Deutschland : Springer International Publishing, 2015, S. 157–168

[Bis06] Bishop, Christopher M.: *Pattern recognition and machine learning*. New York, USA : Springer, 2006. – ISBN 978–0–387–31073–2

[Bjö96] Björck, Åke: *Numerical methods for least squares problems*. Philadelphia, PA, USA : Society for Industrial and Applied Mathematics, 1996

[BKM+20] Beker, Deniz; Kato, Hiroharu; Morariu, Mihai A.; Ando, Takahiro; Kehl, Wadim; Gaidon, Adrien: Monocular Differentiable Rendering for Self-supervised 3D Object Detection.

In: *Computer Vision – ECCV 2020, Part XXI*. Glasgow, UK : Springer, 2020. – ISBN 978–3–030–58589–1, S. 514–529

[BL19] Brazil, Garrick; Liu, Xiaoming: M3D-RPN: Monocular 3D Region Proposal Network for Object Detection. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Südkorea, 2019, S. 9286–9295

[BXC20] Bao, Wentao; Xu, Bin; Chen, Zhenzhong: MonoFENet: Monocular 3D Object Detection With Feature Enhancement Networks. In: *IEEE Transactions on Image Processing* 29 (2020), S. 2753–2765

[CC18] Chang, Jia-Ren; Chen, Yong-Sheng: Pyramid Stereo Matching Network. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018, S. 5410–5418

[CCR+17] Chabot, Florian; Chaouch, Mohamed; Rabarisoa, Jaonary; Teuliere, Celine; Chateau, Thierry: Deep MANTA: A Coarse-to-Fine Many-Task Network for Joint 2D and 3D Vehicle Analysis from Monocular Image. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI, USA, 2017, S. 1827–1836

[CKZ+15] Chen, Xiaozhi; Kundu, Kaustav; Zhu, Yukun; Berneshawi, Andrew G.; Ma, Huimin; Fidler, Sanja; Urtasun, Raquel: 3D Object Proposals for Accurate Object Class Detection. In: *Advances in Neural Information Processing Systems*. Montréal, Kanada, 2015, S. 424–432

[CKZ+16] Chen, Xiaozhi; Kundu, Kaustav; Zhang, Ziyu; Ma, Huimin; Fidler, Sanja; Urtasun, Raquel: Monocular 3D Object Detection for Autonomous Driving. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA, 2016, S. 2147–2156

[CLJ+20] Cai, Yingjie; Li, Buyu; Jiao, Zeyu; Li, Hongsheng; Zeng, Xingyu; Wang, Xiaogang: Monocular 3D Object Detection with Decoupled Structured Polygon Estimation and Height-Guided Depth Estimation. (2020). `http://arxiv.org/abs/2002.01619`. – zuletzt abgerufen: 06.03.2021

[CLSJ20] Chen, Yilun; Liu, Shu; Shen, Xiaoyong; Jia, Jiaya: DSGN: Deep Stereo Geometry Network for 3D Object Detection. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA, 2020, S. 12533–12542

[COR+16] Cordts, Marius; Omran, Mohamed; Ramos, Sebastian; Rehfeld, Timo; Enzweiler, Markus; Benenson, Rodrigo; Franke, Uwe; Roth, Stefan; Schiele, Bernt: The cityscapes dataset for semantic urban scene understanding. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. Las Vegas, NV, USA, 2016, S. 3213–3223

[CTSL20] Chen, Yongjian; Tai, Lei; Sun, Kai; Li, Mingyang: MonoPair: Monocular 3D Object Detection Using Pairwise Spatial Relationships. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA, 2020, S. 12090–12099

[DB78] De Boor, Carl: *A practical guide to splines*. New York, USA : Springer, 1978

[DHY+20] Ding, Mingyu; Huo, Yuqi; Yi, Hongwei; Wang, Zhe; Shi, Jianping; Lu, Zhiwu; Luo, Ping: Learning Depth-Guided Convolutions for Monocular 3D Object Detection. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Seattle, WA, USA, 2020, S. 4306–4315

[Fau93] Faugeras, Olivier: *Three-dimensional computer vision: a geometric viewpoint*. Cambridge, MA, USA : MIT press, 1993

[FGW+18] Fu, Huan; Gong, Mingming; Wang, Chaohui; Batmanghelich, Kayhan; Tao, Dacheng: Deep Ordinal Regression Network for Monocular Depth Estimation. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018, S. 2002–2011

[FTV00] Fusiello, Andrea; Trucco, Emanuele; Verri, Alessandro: A compact algorithm for rectification of stereo pairs. In: *Machine Vision and Applications* 12 (2000), Nr. 1, S. 16–22

[GBC16] Goodfellow, Ian; Bengio, Yoshua; Courville, Aaron: *Deep learning*. Cambridge, MA, USA : The MIT Press, 2016. – ISBN 978–0–262–03561–3

[GLSU13] Geiger, Andreas; Lenz, Philip; Stiller, Christoph; Urtasun, Raquel: Vision meets robotics: The kitti dataset. In: *The International Journal of Robotics Research* 32 (2013), Nr. 11, S. 1231–1237

[GLU12] Geiger, Andreas; Lenz, Philip; Urtasun, Raquel: Are we ready for autonomous driving? the kitti vision benchmark suite. In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. Providence, RI, USA, 2012, S. 3354–3361

[GWH+20] Garg, Divyansh; Wang, Yan; Hariharan, Bharath; Campbell, Mark; Weinberger, Kilian Q.; Chao, Wei-Lun: Wasserstein Distances for Stereo Disparity Estimation. (2020). `http://arxiv.org/abs/2007.03085`. – zuletzt abgerufen: 06.03.2021

[GWJ+20] Gählert, Nils; Wan, Jun-Jun; Jourdan, Nicolas; Finkbeiner, Jan; Franke, Uwe; Denzler, Joachim: Single-Shot 3D Detection of Vehicles from Monocular RGB Images via Geometry Constrained Keypoints in Real-Time. (2020). `http://arxiv.org/abs/2006.13084`. – zuletzt abgerufen: 06.03.2021

[GWW+19] Gählert, Nils; Wan, Jun-Jun; Weber, Michael; Zöllner, J. M.; Franke, Uwe; Denzler, Joachim: Beyond Bounding Boxes: Using Bounding Shapes for Real-Time 3D Vehicle Detection from Monocular RGB Images. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. Paris, Frankreich, 2019, S. 675–682

[Hir07] Hirschmuller, Heiko: Stereo processing by semiglobal matching and mutual information. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30 (2007), Nr. 2, S. 328–341

[HM12] Hu, Xiaoyan; Mordohai, P.: A Quantitative Evaluation of Confidence Measures for Stereo Vision. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34 (2012), Nr. 11, S. 2121–2133

[HS97] Hartley, Richard I.; Sturm, Peter: Triangulation. In: *Computer Vision and Image Understanding* 68 (1997), Nr. 2, S. 146–157

[HS19] He, Tong; Soatto, Stefano: Mono3D++: Monocular 3D Vehicle Detection with Two-Scale 3D Hypotheses and Task Priors. In: *Proceedings of the AAAI Conference on Artificial Intelligence* Bd. 33. Honolulu, HI, USA, 2019, S. 8409–8416

[HZ04] Hartley, Richard; Zisserman, Andrew: *Multiple view geometry in computer vision*. Cambridge, UK : Cambridge University Press, 2004. – ISBN 978–0–511–18618–9

[HZRS16] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian: Deep Residual Learning for Image Recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA, 2016, S. 770–778

[IS15] Ioffe, Sergey; Szegedy, Christian: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In: Bach, Francis (Hrsg.); Blei, David (Hrsg.): *Proceedings of the 32nd International Conference on Machine Learning*. Lille, Frankreich, 2015, S. 448–456

[Joo11] Joos, Malte: *Modellierung und echtzeitfähiges Tracking befahrbarer Flächen unter Verwendung von Stereo-Kameras*. Karlsruhe, Deutschland, Institut für Mess- und Regelungstechnik, Karlsruher Institut für Technologie, Diplomarbeit, 2011

[JZK19] Jörgensen, Eskil; Zach, Christopher; Kahl, Fredrik: Monocular 3D Object Detection and Box Fitting Trained End-to-End Using Intersection-over-Union Loss. (2019). `http://arxiv.org/abs/1906.08070`. – zuletzt abgerufen: 06.03.2021

[JZW+17] Jiang, Yingying; Zhu, Xiangyu; Wang, Xiaobing; Yang, Shuli; Li, Wei; Wang, Hua; Fu, Pei ; Luo, Zhenbo: R2CNN: Rotational Region CNN for Orientation Robust Scene Text Detection. (2017). `https://arxiv.org/abs/1706.09579`. – zuletzt abgerufen: 06.03.2021

[KLR18] Kundu, Abhijit; Li, Yin; Rehg, James M.: 3D-RCNN: Instance-Level 3D Object Reconstruction via Render-and-Compare. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018, S. 3559–3568

[KLS20] Königshof, Hendrik; Li, Kun; Stiller, Christoph: Vision-based Lifting of 2D Object Detections for Automated Driving. In: *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. Sun City, Südafrika, 2020, S. k.A.

[KML+18] Ku, Jason; Mozifian, Melissa; Lee, Jungwook; Harakeh, Ali; Waslander, Steven L.: Joint 3d proposal generation and object

detection from view aggregation. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid, Spanien, 2018, S. k.A.

[KPW19] Ku, Jason; Pon, Alex D.; Waslander, Steven L.: Monocular 3D Object Detection Leveraging Accurate Proposals and Shape Reconstruction. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 11859–11868

[KS20] Königshof, Hendrik; Stiller, Christoph: Learning-Based Shape Estimation with Grid Map Patches for Realtime 3D Object Detection for Automated Driving. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. Rhodos, Griechenland, 2020, S. k.A.

[KSS19] Königshof, Hendrik; Salscheider, Niels O.; Stiller, Christoph: Realtime 3D Object Detection for Automated Driving Using Stereo Vision and Semantic Information. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. Auckland, Neuseeland, 2019, S. 1405–1410

[LAE+16] Liu, Wei; Anguelov, Dragomir; Erhan, Dumitru; Szegedy, Christian; Reed, Scott; Fu, Cheng-Yang; Berg, Alexander C.: SSD: Single shot multibox detector. In: *Computer Vision – ECCV 2016, Part I*. Amsterdam, Niederlande : Springer, 2016. – ISBN 978–3–319–46448–0, S. 21–37

[LAT02] Labayrade, R.; Aubert, D.; Tarel, J.-P.: Real Time Obstacle Detection on Non Flat Road Geometry through 'V-Disparity' Representation. In: *Proceedings of IEEE Intelligent Vehicle Symposium* Bd. 2. Versailles, Frankreich, 2002, S. 646–651

[LCS19] Li, Peiliang; Chen, Xiaozhi; Shen, Shaojie: Stereo R-CNN Based 3D Object Detection for Autonomous Driving. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 7636–7644

[LGG+17] Lin, Tsung-Yi; Goyal, Priya; Girshick, Ross; He, Kaiming; Dollár, Piotr: Focal loss for dense object detection. In: *Proceedings of the IEEE international conference on computer vision*. Venedig, Italien, 2017, S. 2980–2988

[LHKS19] Lee, Jin H.; Han, Myung-Kyu; Ko, Dong W.; Suh, Il H.: From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation. (2019). https://arxiv.org/abs/1907.10326. – zuletzt abgerufen: 06.03.2021

[LKW20] Li, Chengyao; Ku, Jason; Waslander, Steven L.: Confidence Guided Stereo 3D Object Detection with Split Depth Estimation. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Las Vegas, NV, USA, 2020, S. 5776–5783

[LLX+19] Liu, Lijie; Lu, Jiwen; Xu, Chunjing; Tian, Qi; Zhou, Jie: Deep Fitting Degree Scoring Network for Monocular 3D Object Detection. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 1057–1066

[LOS+19] Li, Buyu; Ouyang, Wanli; Sheng, Lu; Zeng, Xingyu; Wang, Xiaogang: GS3D: An Efficient 3D Object Detection Framework for Autonomous Driving. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 1019–1028

[LWT20] Liu, Zechen; Wu, Zizhang; Toth, Roland: SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Seattle, WA, USA, 2020, S. 4289–4298

[LZLC20] Li, Peixuan; Zhao, Huaici; Liu, Pengfei; Cao, Feidao: RTM3D: Real-Time Monocular 3D Detection from Object Keypoints for Autonomous Driving. In: *Computer Vision – ECCV 2020, Part III*. Glasgow, UK : Springer, 2020. – ISBN 978–3–030–58580–8, S. 644–660

[MAFK17] Mousavian, Arsalan; Anguelov, Dragomir; Flynn, John ; Kosecka, Jana: 3D Bounding Box Estimation Using Deep Learning and Geometry. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Honolulu, HI, USA, 2017, S. 5632–5640

[MKCK17] Murthy, J. K.; Krishna, G.V. S.; Chhaya, Falak; Krishna, K. M.: Reconstructing vehicles from a single image: Shape

priors for road scene understanding. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. Singapur, Singapur, 2017, S. 724–731

[MKG19] Manhardt, Fabian; Kehl, Wadim; Gaidon, Adrien: ROI-10D: Monocular Lifting of 2D Detection to 6D Pose and Metric Shape. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 2064–2073

[MS87] Matthies, Larry; Shafer, Stevena: Error modeling in stereo navigation. In: *IEEE Journal on Robotics and Automation* 3 (1987), Nr. 3, S. 239–248

[MWL+19] Ma, Xinzhu; Wang, Zhihui; Li, Haojie; Zhang, Pengbo; Ouyang, Wanli; Fan, Xin: Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Südkorea, 2019, S. 6850–6859

[NDF+04] Nedevschi, Sergiu; Danescu, Radu; Frentiu, Dan; Marita, Tiberiu; Oniga, Florin; Pocol, Ciprian; Graf, Thorsten; Schmidt, Rolf: High accuracy stereovision approach for obstacle detection on non-planar roads. In: *8th IEEE International Conference on Intelligent Engineering Systems (INES)*. Cluj-Napoca, Rumänien, 2004, S. 211–216

[NPK+19] Naiden, Andretti; Paunescu, Vlad; Kim, Gyeongmo; Jeon, ByeongMoon; Leordeanu, Marius: Shift R-CNN: Deep Monocular 3D Object Detection With Closed-Form Geometric Constraints. In: *2019 IEEE International Conference on Image Processing (ICIP)*. Taipeh, Taiwan, 2019, S. 61–65

[NW06] Nocedal, Jorge; Wright, Stephen J.: *Numerical optimization*. New York, USA : Springer, 2006. – ISBN 978–0–387–30303–1

[ONMT07] Oniga, Florin; Nedevschi, Sergiu; Meinecke, Marc M. ; To, Thanh B.: Road surface and obstacle detection based on elevation maps from dense stereo. In: *2007 IEEE Intelligent Transportation Systems Conference*. Seattle, WA, USA, 2007, S. 859–865

[PKLW20] Pon, Alex D.; Ku, Jason; Li, Chengyao; Waslander, Steven L.: Object-Centric Stereo Matching for 3D Object Detection. In:

2020 IEEE International Conference on Robotics and Automation (ICRA). Paris, Frankreich, 2020, S. 8383–8389

[PKVG99] Pollefeys, Marc; Koch, Reinhard; Van Gool, Luc: A simple and efficient rectification method for general motion. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision* Bd. 1. Korfu, Griechenland, 1999, S. 496–501

[PLB15] Papageorgiou, Markos; Leibold, Marion; Buss, Martin: *Optimierung*. Berlin, Deutschland : Springer, 2015. http://dx.doi.org/10.1007/978-3-662-46936-1. – ISBN 978–3–662–46936–1

[QLW+18] Qi, Charles R.; Liu, Wei; Wu, Chenxia; Su, Hao; Guibas, Leonidas J.: Frustum PointNets for 3D Object Detection From RGB-D Data. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018, S. 918–927

[QWL19a] Qin, Zengyi; Wang, Jinglu; Lu, Yan: MonoGRNet: A Geometric Reasoning Network for Monocular 3D Object Localization. In: *Proceedings of the AAAI Conference on Artificial Intelligence* Bd. 33. Honolulu, HI, USA, 2019, S. 8851–8858

[QWL19b] Qin, Zengyi; Wang, Jinglu; Lu, Yan: Triangulation learning network: from monocular to stereo 3d object detection. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Long Beach, CA, USA, 2019, S. 7615–7623

[RDGF16] Redmon, Joseph; Divvala, Santosh; Girshick, Ross; Farhadi, Ali: You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. Las Vegas, NV, USA, 2016, S. 779–788

[RHGS17] Ren, Shaoqing; He, Kaiming; Girshick, Ross; Sun, Jian: Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (2017), Nr. 6, S. 1137–1149

[RKC19] Roddick, Thomas; Kendall, Alex; Cipolla, Roberto: Orthographic feature transform for monocular 3D object detection. In: *30th British Machine Vision Conference 2019, BMVC 2019*. Cardiff, UK, 2019, S. k.A.

[Ros58] Rosenblatt, Frank: The perceptron: A probabilistic model for information storage and organization in the brain. In: *Psychological Review* 65 (1958), Nr. 6, S. 386–408

[RS14] Ranft, Benjamin; Strauss, Tobias: Modeling arbitrarily oriented slanted planes for efficient stereo vision based on block matching. In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. Qingdao, China, 2014, S. 1941–1947

[Sal20] Salscheider, Niels O.: Simultaneous Object Detection and Semantic Segmentation. In: *Proceedings of the 9th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM,*. Valletta, Malta : SciTePress, 2020, S. 555–561

[SBP+19] Simonelli, Andrea; Bulo, Samuel R.; Porzi, Lorenzo; Lopez-Antequera, Manuel; Kontschieder, Peter: Disentangling Monocular 3D Object Detection. In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Seoul, Südkorea, 2019, S. 1991–1999

[SBP+20] Simonelli, Andrea; Bulò, Samuel R.; Porzi, Lorenzo; Ricci, Elisa; Kontschieder, Peter: Towards Generalization Across Depth for Monocular 3D Object Detection. In: *Computer Vision – ECCV 2020, Part XXII*. Glasgow, UK : Springer, 2020, S. 767–782

[SCX+20] Sun, Jiaming; Chen, Linghao; Xie, Yiming; Zhang, Siyu; Jiang, Qinhong; Zhou, Xiaowei; Bao, Hujun: Disp R-CNN: Stereo 3D Object Detection via Shape Prior Guided Instance Disparity Estimation. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Seattle, WA, USA, 2020, S. 10545–10554

[Ska18] Skansi, Sandro: *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Cham, Schweiz : Springer International Publishing, 2018. http://dx.doi.org/10.1007/978-3-319-73004-2. – ISBN 978–3–319–73004–2

[SS02] Scharstein, Daniel; Szeliski, Richard: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms.

In: *International Journal of Computer Vision* 47 (2002), Nr. 1-3, S. 7–42

[Sta17] Statistisches Bundesamt: *Unfallentwicklung auf deutschen Straßen 2017*. https://www.destatis.de/DE/Presse/Pressekonferenzen/2018/Verkehrsunfaelle-2017/pressebroschuere-unfallentwicklung.pdf?__blob=publicationFile. Version: 2017. – zuletzt abgerufen: 06.03.2021

[SWL19] Shi, Shaoshuai; Wang, Xiaogang; Li, Hongsheng: PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 770–779

[Sze11] Szeliski, Richard: *Computer Vision*. London, UK : Springer London, 2011 (Texts in Computer Science). http://dx.doi.org/10.1007/978-1-84882-935-0. – ISBN 978–1–84882–935–0

[TV98] Trucco, Emanuele; Verri, Alessandro: *Introductory techniques for 3-D computer vision*. Bd. 201. Englewood Cliffs, NJ, USA : Prentice Hall, 1998

[VAL19] Vianney, Jean Marie U.; Aich, Shubhra; Liu, Bingbing: RefinedMPL: Refined Monocular PseudoLiDAR for 3D Object Detection in Autonomous Driving. (2019). http://arxiv.org/abs/1911.09712. – zuletzt abgerufen: 06.03.2021

[WBR+09] Wedel, Andreas; Badino, Hernán; Rabe, Clemens; Loose, Heidi; Franke, Uwe; Cremers, Daniel: B-spline modeling of road surfaces with an application to free-space estimation. In: *IEEE Transactions on Intelligent Transportation Systems* 10 (2009), Nr. 4, S. 572–583

[WCG+19] Wang, Yan; Chao, Wei-Lun; Garg, Divyansh; Hariharan, Bharath; Campbell, Mark; Weinberger, Kilian Q.: Pseudo-LiDAR From Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 8437–8445

[WFSF18] Wirges, Sascha; Fischer, Tom; Stiller, Christoph; Frias, Jesus B.: Object detection and classification in occupancy grid maps using deep convolutional networks. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. Maui, HI, USA, 2018, S. 3530–3535

[WK19] Weng, Xinshuo; Kitani, Kris: Monocular 3D Object Detection with Pseudo-LiDAR Point Cloud. In: *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*. Seoul, Südkorea, 2019, S. 857–866

[WSVDH19] Wu, Zifeng; Shen, Chunhua; Van Den Hengel, Anton: Wider or deeper: Revisiting the resnet model for visual recognition. In: *Pattern Recognition* 90 (2019), S. 119–133

[XC18] Xu, Bin; Chen, Zhenzhong: Multi-level Fusion Based 3D Object Detection from Monocular Images. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. Salt Lake City, UT, USA, 2018, S. 2345–2353

[XWLS15] Xiang, Yu; Wongun Choi; Lin, Yuanqing; Savarese, Silvio: Data-driven 3D Voxel Patterns for object category recognition. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Boston, MA, USA, 2015, S. 1903–1911

[XZY+20] Xu, Zhenbo; Zhang, Wei; Ye, Xiaoqing; Tan, Xiao; Yang, Wei; Wen, Shilei; Ding, Errui; Meng, Ajin; Huang, Liusheng: Zoomnet: Part-aware adaptive zooming neural network for 3d object detection. In: *Proceedings of the AAAI Conference on Artificial Intelligence* Bd. 34. New York, USA, 2020, S. 12557–12564

[YWC+20] You, Yurong; Wang, Yan; Chao, Wei-Lun; Garg, Divyansh; Pleiss, Geoff; Hariharan, Bharath; Campbell, Mark; Weinberger, Kilian Q.: Pseudo-LiDAR++: Accurate Depth for 3D Object Detection in Autonomous Driving. In: *International Conference on Learning Representations*. Addis Abeba, Äthiopien, 2020, k.A.

[Zei12] Zeiler, Matthew D.: ADADELTA: An Adaptive Learning Rate Method. (2012). `https://arxiv.org/abs/1212.5701`. – zuletzt abgerufen: 06.03.2021

[ZPYT19] Zhang, Feihu; Prisacariu, Victor; Yang, Ruigang; Torr, Philip H.: GA-Net: Guided Aggregation Net for End-To-End Stereo Matching. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. Long Beach, CA, USA, 2019, S. 185–194

[ZSS14] Zia, Muhammad Z.; Stark, Michael; Schindler, Konrad: Are Cars Just 3D Boxes? Jointly Estimating the 3D Shape of Multiple Objects. In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA, 2014, S. 3678–3685

[ZSSS13] Zia, M Z.; Stark, Michael; Schiele, Bernt; Schindler, Konrad: Detailed 3d representations for object recognition and modeling. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35 (2013), Nr. 11, S. 2608–2623

[ZW94] Zabih, Ramin; Woodfill, John: Non-parametric local transforms for computing visual correspondence. In: *Computer Vision – ECCV '94, Part II*. Stockholm, Schweden, 1994. – ISBN 978–3–540–48400–4, S. 151–158

[ZWK19] Zhou, Xingyi; Wang, Dequan; Krähenbühl, Philipp: Objects as Points. (2019). `https://arxiv.org/abs/1904.07850`. – zuletzt abgerufen: 06.03.2021

Eigene Veröffentlichungen

[1] Königshof, Hendrik; Li, Kun; Stiller, Christoph: Vision-based Lifting of 2D Object Detections for Automated Driving. In: *2020 IEEE 23rd International Conference on Information Fusion (FUSION)*. Sun City, Südafrika, 2020, S. k.A.

[2] Königshof, Hendrik; Salscheider, Niels O.; Stiller, Christoph: Realtime 3D Object Detection for Automated Driving Using Stereo Vision and Semantic Information. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. Auckland, Neuseeland, 2019, S. 1405–1410

[3] Königshof, Hendrik; Stiller, Christoph: Learning-Based Shape Estimation with Grid Map Patches for Realtime 3D Object Detection for Automated Driving. In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. Rhodos, Griechenland, 2020, S. k.A.

[4] Naumann, Maximilian; Königshof, Hendrik; Lauer, Martin; Stiller, Christoph: Safe but not Overcautious Motion Planning under Occlusions and Limited Sensor Range. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. Paris, Frankreich, 2019, S. 140–145

[5] Naumann, Maximilian; Königshof, Hendrik; Stiller, Christoph: Provably Safe and Smooth Lane Changes in Mixed Traffic. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. Auckland, Neuseeland, 2019, S. 1832–1837

[6] Richter, Sven; Wirges, Sascha; Königshof, Hendrik; Stiller, Christoph: Fusion of range measurements and semantic estimates in an evidential framework / Fusion von Distanzmessungen und semantischen Größen im Rahmen der Evidenztheorie. In: *tm - Technisches Messen* 86 (2019), S. 102–106

[7] Schreiber, Markus; Königshof, Hendrik; Hellmund, André-Marcel; Stiller, Christoph: Vehicle localization with tightly coupled GNSS and visual odometry. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. Göteborg, Schweden, 2016, S. 858–863

[8] Zhan, Wei; Sun, Liting; Wang, Di; Shi, Haojie; Clausse, Aubrey; Naumann, Maximilian; Kümmerle, Julius; Königshof, Hendrik; Stiller, Christoph; La Fortelle, Arnaud de; Tomizuka, Masayoshi: INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. (2019). https://arxiv.org/abs/1910.03088. – zuletzt abgerufen: 06.03.2021