HAW HAMBURG

BACHELOR THESIS
Timon Rupelt

Untersuchung von Lösungen mit maschinellem Lernen für die Tiefenbestimmung mit Stereobilddaten unter umweltbedingten Störungen.

FAKULTÄT TECHNIK UND INFORMATIK
Department Informatik

Faculty of Engineering and Computer Science
Department Computer Science

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Timon Rupelt

Untersuchung von Lösungen mit maschinellem Lernen für die Tiefenbestimmung mit Stereobilddaten unter umweltbedingten Störungen.

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Informatik Technischer Systeme*
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Tim Tiedemann
Zweitgutachter: Prof. Dr. Peer Stelldinger

Eingereicht am: 02.10.2023

Timon Rupelt

Thema der Arbeit

Untersuchung von Lösungen mit maschinellem Lernen für die Tiefenbestimmung mit Stereobilddaten unter umweltbedingten Störungen.

Stichworte

Stereo Vision, Tiefenbestimmung, Stereokorrespondenz, Maschinelles Lernen, Deep Learning, Überwachtes Lernen, Regentropfen, Nebel, Verdeckung

Kurzzusammenfassung

Das Ziel dieser Arbeit ist, die Eignung von neuronalen Netzen für die Tiefenbestimmung mit Stereobildern unter dem Einfluss von Störungen zu untersuchen. Der Fokus wird dabei auf die Störung durch Regentropfen, Nebel und die Verdeckung durch Objekte wie Blätter oder Eis gelegt. Zu diesem Zweck wird eine ausgewählte allgemeine Netzwerkarchitektur für Stereokorrespondenz mit verschiedenen Datensätzen echter und synthetischer Daten trainiert. Es werden handelsübliche Stereokameras für die Aufnahme von Trainingsdaten und passenden Testszenarien für die Evaluierung genutzt. Durch die Entwicklung verschiedener Methoden zur Augmentation von existieren Daten mit diesen Störeffekten wird die Menge der für das Training geeigneten Datensätze vergrößert. Die Evaluierung zeigt, dass die Disparitäten für Bildbereiche, die durch Regentropfen und Verdeckungen kleiner bis mittlerer Größe beeinflusst sind, zuverlässig bestimmt werden können. Im Vergleich mit den verwendeten Stereokameras ist ein deutlicher Vorteil der Netzwerke festzustellen. Für Bedingungen mit dichtem Nebel führt der große Anteil gleichfarbiger Bildinformationen zu hohem Detailverlust in den Schätzungen. Das Training mit einem synthetischen Datensatz resultiert dabei in vielversprechenden, aber noch unzureichenden Ergebnissen.

Timon Rupelt

Title of Thesis

Study of machine learning solutions for depth estimation with stereo image data under environmental disturbances.

Keywords

stereo vision, depth estimation, stereo matching, machine learning, deep learning, supervised learning, Regentropfen, Nebel, Verdeckung

Abstract

This work aims to investigate the suitability of neural networks for depth estimation using stereo images in the presence of various disturbances. The primary focus will be on perturbations caused by raindrops, fog, and occlusion by objects such as leaves or ice. A selected neural network architecture designed for stereo correspondence will serve as the foundation for training using various datasets, comprising both real-world and synthetic data. Commercially available stereo cameras will be utilized to capture training data, along with suitable test scenarios for evaluation. By developing various methods to augment existing data with disturbances, the set of datasets suitable for training is expanded. The evaluation reveals that the disparities in image areas affected by raindrops and small to medium-sized occlusions can be reliably determined. In comparison to the stereo cameras used, the networks exhibit a clear advantage. However, in conditions of dense fog, the extensive amount of similarly colored image information leads to a significant loss of detail in the estimates. Although training with a synthetic dataset yields promising results, they still remain insufficient.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungen

CNN Convolutional Neural Network (deut.: Faltungsnetzwerk).

EPE End-Point-Error.

FM Feature Map (deut.: Merkmalskarte).

LiDAR Light Detection and and Ranging.

1 Einleitung

In diesem Kapitel wird die Motivation und die Zielsetzung dieser Bachelorarbeit erläutert. Im Weiteren werden die Inhalte der Arbeit anhand des Aufbaus der Arbeit beschrieben und ein kurzer Überblick des aktuellen Forschungsstands zu dem Themenbereich der Arbeit geben.

1.1 Motivation

Die Bestimmung von Tiefe ist ein essenzieller Bestandteil der Wahrnehmungsfähigkeit eines Systems in vielen Bereich wie der Robotik und dem autonomen Fahren. Diese Tiefeninformationen werden meistens durch die Nutzung von LiDAR oder Tiefen-/Stereokameras erlangt. Beide Methoden bergen jedoch in manchen Anwendungsfällen Probleme, welche zu unzuverlässigen Ergebnissen führen können. LiDAR und andere Time-of-Flight Sensoren haben Schwierigkeiten in Situationen mit spiegelnden, absorbierenden und transparenten Oberflächen oder lichtbrechenden Partikel wie bei Schnee und Nebel korrekte Messungen zu geben [Bijelic et al., 2020a]. Sie bieten zudem nur spärliche (engl. sparse) Messungen der Umgebung. Die Verbesserung der Messungen ist mit hohen Kosten verbunden.

Eine günstigere Alternative stellt die Verwendung von Stereokameras dar, mit denen es möglich ist, Tiefeninformationen aus Bildern zu bestimmen, die eine Szene aus unterschiedlichen Perspektiven zeigen. Dies ist der zentrale Inhalt des Themenbereichs der *Stereo Vision*, die ein Teilgebiet von *Computer Vision* ist. Dabei werden die korrespondierenden Pixel in beiden Bildern gesucht und deren relative Verschiebung als Disparität bestimmt. Das Ergebnis ist eine Disparitätskarte, die für jedes Pixel der abgebildeten Szene die Verschiebung zwischen den Perspektiven angibt. Mit diesen Verschiebungen und den Parametern der verwendeten Kameras kann dann die Tiefe der Szene bestimmt werden. Im Vergleich zu den spärlichen Messpunkten eines LiDAR ist die Dichte der vorhandenen Tiefeninformationen sehr hoch. Das Finden der korrespondierenden Pixel wird

als das Stereokorrespondenzproblem bezeichnet [Beyerer et al., 2016, S. 354]. Zur Lösung des Problems existieren unterschiedliche traditionelle und modernere Methoden.

Doch auch die Qualität von Bildern kann durch umweltbedingten Störeffekten wie Nebel oder Regentropfen beeinträchtigt werden, was zu veränderten oder verdeckten Bildinformationen führt. Eine mögliche Lösung, um auch für diese Art von Daten gute Ergebnisse zu erhalten, könnten Ansätze des maschinellen Lernens sein. Der erfolgreiche Einsatz von *Deep Learning* in der Computer Vision trieb auch die Forschung für den Einsatz neuronaler Netzwerke zur Lösung des Stereokorrespondenzproblems voran [Poggi et al., 2021]. Diese Netzwerke sollen das Finden der Korrespondenzen erlernen, um bessere und effizientere Schätzungen der Disparitäten zu erbringen. Dies funktioniert gut mit Bildern in optimalen Bedingungen. Besonders in den Bedingungen mit Störeffekten könnten solche lern-basierte Ansätze einen großen Unterschied machen. Ein Netzwerk könnte erlernen, die Informationen trotz Störung zu extrahieren und fehlende Informationen sinnvoll zu ersetzten. Sollte dies Erfolg zeigen, könnten Stereokameras anstelle von einem oder mehreren teureren Sensoren für diese Situationen eingesetzt werden.

1.2 Zielsetzung

Im Rahmen dieser Arbeit soll der Einsatz von neuronalen Netzwerken zur Lösung des Stereokorrespondenzproblems unter dem Einfluss von umweltbedingten Störungen untersucht werden. Ziel ist es, ein Überblick über die existierenden Ansätze zu geben und die Leistung einer gewählten Netzwerkarchitektur nach dem Training für eine Auswahl von Störeffekten auszuwerten. Aus der Menge der Störeffekte, die auftreten könnten, wird der Fokus auf Regentropfen, Nebel und allgemeine Verdeckung durch Objekte gelegt.

1.3 Aufbau der Arbeit

Zu Beginn werden in Kapitel 2 die nötigen Grundlagen der Stereo Vision und des maschinellen Lernens mit neuronalen Netzwerken erläutert. Es wird ein Überblick über die Forschung für Stereokorrespondenz mit Deep Learning gegeben und über die Eigenschaften der Netzwerkarchitekturen gesprochen. Darauffolgend werden in Kapitel 3 die betrachteten Störeffekte und deren Einfluss auf LiDAR und Kameras besprochen. Es werden existierenden *Deep Learning* Ansätze für Stereokorrespondenz vorgestellt, die speziell für

diese Bedingungen entwickelt wurden.
In Kapitel 4 wird die Auswahl des verwendeten Netzwerkes erläutert und dessen Architektur und Besonderheiten beschrieben. Im Weiteren werden in Kapitel 5 die im Rahmen dieser Arbeit betrachteten Datensätze vorgestellt. Dabei werden verfügbare Datensätze anderer Projekte beschrieben und im Hinblick auf die Verwendbarkeit für diese Arbeit bewertet. Es wird die selbst durchgeführte Datensammlung spezieller Aufnahmen für die Auswertung der trainierten Netzwerke beschrieben. Des Weiteren werden die entwickelten Augmentationsmethoden für die Erstellung künstlicher Störeffekte und die damit erstellten Datensätze vorgestellt.
Kapitel 6 beinhaltet die Beschreibung des Trainings der neuronalen Netzwerke für die verschiedenen Störeffekte. Die Auswertung der trainierten Netze wird in Kapitel 7 dargestellt. Es werden die verwendeten Metriken erläutert, mit denen die Qualität der Schätzungen der Netzwerke bewertet werden. Anhand dieser Metriken werden die Netzwerke verglichen und die Ergebnisse diskutiert.
Zum Schluss wird in Kapitel 8 auf Basis der Ergebnisse für die Eignung von neuronalen Netzwerken für die Tiefenbestimmung unter Einfluss der betrachteten Störeffekte ein Fazit gezogen. Auch wird die Qualität der Datensätze und die Eignung der Augmentationsmethoden und die damit erstellen künstlichen Datensätze besprochen. Im darauffolgenden Ausblick wird dargestellt, welche Verbesserungen und mögliche alternative Ansätze verfolgt werden könnten.

1.4 Forschungsstand

Der Einfluss von unterschiedlichen Wettereffekten auf verschiedene Sensoren ist besonders im Kontext des autonomen Fahrens ein wichtiges Thema. Arbeiten wie die von Zhang et al. [2023] und Yoneda et al. [2019] geben einen Überblick über den Einfluss, den Bedingungen wie Regen, Nebel auf LiDAR, Radar, Kameras und andere Sensoren haben. Die Autoren Kutila et al. [2018] nutzen eine Nebelkammer, um in ihrer Arbeit konkrete Tests für LiDAR und Radar in Regen und Nebel durchzuführen. Jedoch werden keine Untersuchungen für Stereokameras bzw. Stereokorrespondenz-Algorithmen durchgeführt.
Im Kontext von Deep Learning für Stereokorrespondenz existieren vereinzelte Ansätze speziell für nebelige Bedingungen wie die von Song et al. [2020] und Yao und Yu [2022]. Für den Einfluss von Regentropfen oder anderer Verdeckung durch Objekte existieren

keine bekannten Arbeiten. Die existierenden Ansätze für neblige Bedingungen und Ansätze, die mit Verdeckung durch Regentropfen und anderen Objekten in Verbindung gebracht werden können, werden in Kapitel 3 näher beschrieben.
Die Forschung für Deep Learning für Stereokorrespondenz bezieht sich grundsätzlich auf optimale Bedingungen und ist seit 2015 durch die Arbeit [Žbontar und LeCun, 2015] angestoßen worden. Sie verwendeten ein Faltungsnetzwerk als Unterstützung für das Finden von Korrespondenzen. Ein weiterer Meilenstein war die Arbeit von Mayer et al. [2016b], die das erste *End-to-End*-Netzwerk entwickelten, das eine vollständige Deep-Learning-Lösung für das Korrespondenzproblem darstellte. Auf diesen Arbeiten aufbauend wurde die Forschung mit vielen weiteren Arbeiten vorangetrieben. Die Entwicklung und Eigenschaften der entwickelten Netzwerkarchitekturen werden in Abschnitt 2.3 näher erläutert.

2 Grundlagen

Bevor Lösungen für die Problemstellung besprochen werden können, ist es nötig, einige grundlegende Begriffe und Konzepte zu erläutern. Deshalb findet in diesem Kapitel eine Einführung in die Bereiche der *Stereo Vision* und dem *Deep Learning* statt. Des Weiteren wird die Entwicklung von Architekturen neuronaler Netzwerke für die Tiefenbestimmung und deren Merkmale beschrieben.

2.1 Stereo Vision

Computer Vision umfasst viele Methoden, wie ein System seine Umgebung wahrnehmen kann. Eine Möglichkeit, Tiefeninformationen über die Umgebung zu erhalten, ist die Tiefenbestimmung auf Basis von *Stereoskopie*. Durch die Aufnahme einer Szene aus mindestens zwei oder mehr Perspektiven kann bei Kenntnis der intrinsischen (interen) und extrinsischen (externen) Kameraparameter die räumliche Position eines Punktes in der Szene bestimmt werden [Franke und Gehrig, 2015, S. 378].

Das Prinzip beruht auf Triangulation, einem Verfahren zur Entfernungsmessung, anhand der Geometrie von ebenen Dreiecken. Mit den bekannten Werten für zwei Winkel α und β und einer Seitenlänge, lässt sich das Dreieck eindeutig bestimmen (s. Abbildung 2.1). Um das Verfahren anwenden zu können, müssen zuerst die Werte für die Winkel und die Seitenlänge bestimmt werden. Dafür ist der Aufbau des Stereokamerasystems ausschlaggebend.

Ein Stereokamerasystem besteht aus zwei meistens identischen Kameras mit den optischen Zentren C_l und C_r, welche auf denselben Bereich eines dreidimensionalen Raumes ausgerichtet sind. Die Position der Kamera und ihre Orientierung werden als *extrinsische* Kameraparameter bezeichnet.

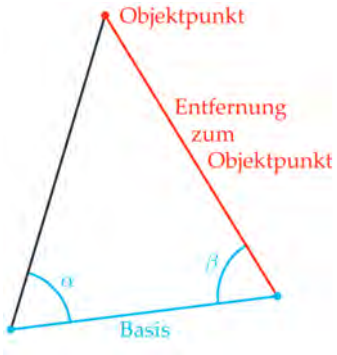


Abbildung 2.1: Entfernungsbestimmung mit dem Triangulationsverfahren bei bekannten Winkeln α und β und der Länge der *Basis*. Aus [Beyerer et al., 2016, Abb. 7.32].

Diese werden als 3×4 Matrix (Gl. 2.1) beschrieben, welche aus der 3×3 Rotationsmatrix und dem 3×1 Translationsvektor zusammengesetzt ist.

$$[R|T] = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{11} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \tag{2.1}$$

Die internen Kameraparameter wie die das optische Zentrum (x_0, y_0) und die Brennweite f (engl.: focal length) werden als *intrinische* Kameraparameter bezeichnet und werden mit der 3×3 Matrix K (Gl. 2.2)

$$K = \begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

beschrieben. Dabei ist $f_x = -\frac{b}{s_x}$ und $f_y = -\frac{b}{s_y}$, mit der Bildbreite b und der Höhe s_y und Breite s_x des Bildsensors der Kamera.

Die optischen Zentren der Kameras sind um eine gewisse Distanz verschoben, damit der dreidimensionale Verschiebungseffekt auftritt [Franke und Gehrig, 2015, 379]. Diese Distanz wird als die *Basis* (engl.: baseline) bezeichnet. Die zwei Kameras bilden einen Punkt X_W des dreidimensionalen Raumes als je einen zweidimensionalen Punkt x_1 und x_2 auf die linke und rechte Bildebene ab (Abbildung 2.2).

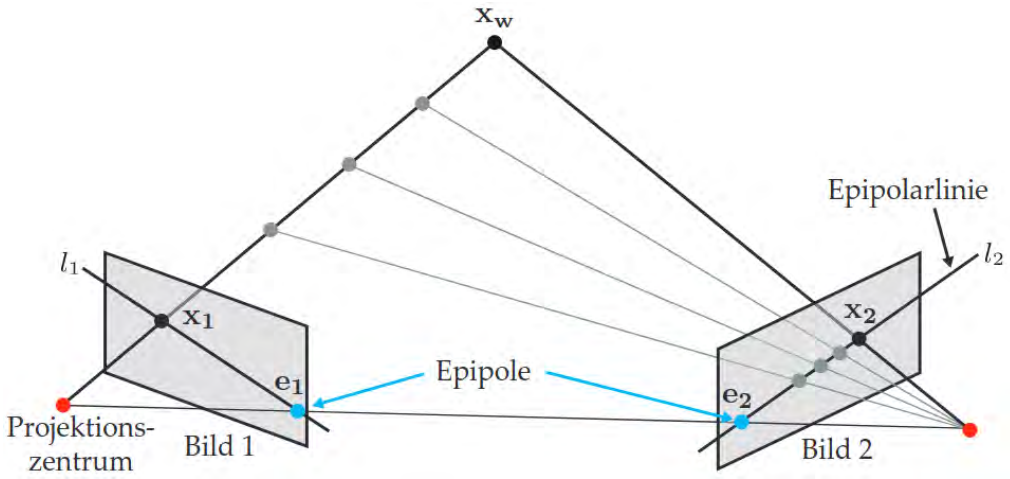


Abbildung 2.2: Epipolare Geometrie eines Stereokamerasystems. Aus [Beyerer et al., 2016, Abb. 7.67].

Mit dem Punkt X_W spannen die optischen Zentren eine Ebene, welche als *Epipolarebene* bezeichnet wird [Franke und Gehrig, 2015, 379]. Die Geraden, wo die Epipolarebene die Bildebenen schneidet, sind die *Epipolarlinien* l_1 und l_2. Auf diesen Linien liegen beide 2D Punkte x_l und x_r.
Mit der Basis als Seitenlänge und den Winkeln, welche mit den x_l und x_r bestimmt werden können, kann das Triangulationsverfahren angewendet werden. Die Positionen der Punkte x_l und x_r sind jedoch nicht bekannt und müssen deshalb zuerst bestimmt werden. Durch den Verschiebungseffekt sind diese Punkt aber auch nicht an den gleichen Stellen in den jeweiligen Bildebenen zu finden.
Dies wird als das *Korrespondenzproblem* bezeichnet, da die korrespondierenden Punkte aus der linken und rechten Bildebene gefunden werden müssen [Beyerer et al., 2016, S. 354]. Aufgrund der beschriebene Epipolargeometrie lässt sich dieser Prozess jedoch vereinfachen. Unter der Voraussetzung, dass die Kameras korrekt ausgerichtet sind, sodass x_l und x_r auf einer Linie liegen, kann die Suche nach Korrespondenzen auf eine horizontale Linie beschränkt werden [Franke und Gehrig, 2015, 379].

Die schräg verlaufenden Epipolarlinien stellen dabei allerdings eine ungünstige Struktur für die Suche dar. Durch *Rektifizierung* der Bilder können die Kamerakoordinatensysteme virtuell gleich ausgerichtet werden. Die Epipolarlinien verlaufen damit horizontal auf der gleichen Höhe, wie in Abbildung 2.3 dargestellt ist.

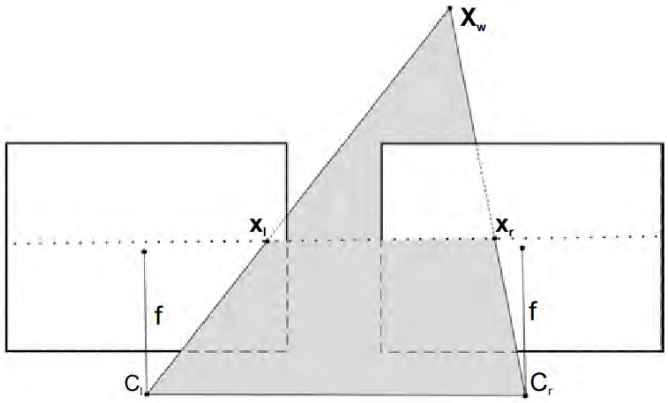


Abbildung 2.3: Ein rektifiziertes Strereokamerasystem. Die gepunktete Epipolarlinie verläuft auf gleicher Höhe durch beide Bildebene. Mit f ist die Brennweite beschrieben. Aus [Franke und Gehrig, 2015, Abb. 21.8].

Damit sind die optimalen Bedingungen für die Lösung des Korrespondenzproblems gegeben. Der Prozess des Findens von Korrespondenzen wird auch *stereo matching* genannt. Algorithmen zum Lösen des Problems werden als Stereokorrespondenz-Algorithmen bzw. Stereo-Matching-Algorithmen bezeichnet.

2.1.1 Stereo-Matching

Der Kern des zugrunde liegende *Korrespondenzproblems* ist das Finden von Pixeln in dem linken und rechten Bild, die denselben Punkt in der aufgenommenen Szene darstellten. Die relative horizontale Verschiebung der korrespondierenden Pixel wird als *Disparität d* bezeichnet und wird in der Einheit Pixel (px) angegeben [Franke und Gehrig, 2015, S. 380].
Auf der Grundlage des linken Bildes wird durch den Vergleich eines Pixels mit einem Pixel des rechten Bildes eine Schätzung der Disparität für das jeweilige Pixel aufgestellt. In Abbildung 2.4 wird eine simple Darstellung dies Prinzips gezeigt. Für diesen Vergleich wird der Ähnlichkeitsgrad der Pixel meist anhand der Pixelintensitäten, der Farbwerte eines Pixels bestimmt. Die resultierenden Werte werden als *Ähnlichkeitskosten* bzw. *Matching-Kosten* bezeichnet.

Die Disparität ist ein Maß für die Raumtiefe eines Punktes der Szene und verhält sich umgekehrt proportional zur Tiefe [Franke und Gehrig, 2015, S. 381]. Eine hohe Disparität entspricht einer geringen Tiefe und eine geringe Disparität einer hohen Tiefe. Für den von

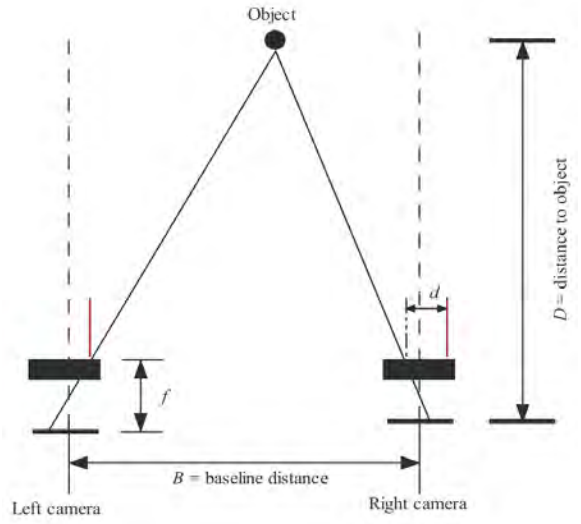


Abbildung 2.4: Simple Darstellung des Prinzips von Stereo-Vision. Die rote Linie gibt die Position des Punktes in der linken Bildebene dar. Aus [Hong und Kim, 2017, Abb. 1].

den korrespondierenden Pixeln beschriebenen Punkt der Szene kann mit der Disparität d und den kameraspezifischen Parametern der Brennweite f und der Basis b der Tiefenwert Z mit der Gleichung 2.3 (aus [Franke und Gehrig, 2015, Gl. 21.22].) berechnet werden.

$$Z = \frac{bf}{d} \tag{2.3}$$

Die Tiefe kann dabei aber nur begrenzt weit bestimmt werden. Abhängig der kameraspezifischen Parameter sind Disparitäten für Objekte ab einer gewissen Distanz null, wodurch sich keine Tiefe bestimmen lässt.

Das Ergebnis der Disparitätsschätzung für jedes Pixel aus Sicht des linken Bildes wird grundsätzlich als 16-Bit Schwarz-Weiß-Bild abgespeichert, einer sogenannte Disparitätskarte (engl.: Disparity map). Zumeist wird hier das *Portable Network Image*[1] Bildformat verwendet, das verlustfreie Kompression und verschiedene Farbtiefen bietet. Oftmals werden die Disparitäten auch mit dem Wert 256 skaliert, um nach Einlesen und dem Dividieren durch 256 Disparitäten mit Nachkommastellen zu erhalten. Eine alternative Form stellt das *Portable Float Map (PFM)*[2] Format dar, mit welchem sich die Disparitätskarte in 32-Bit Gleitkommazahlen sichern lässt. Dies bietet den Vorteil einer höheren Präzision

[1] http://www.libpng.org/pub/png/spec/
[2] https://pauldebevec.com/Research/HDR/PFM/

auf Kosten der Dateigröße. So benötigt eine Disparitätskarte, die im PFM-Format eine Dateigröße von 2,1 MB besitzt, als 16-Bit PNG-Datei nur 73,4 kB Speicherplatz.

Für das Erstellen von Disparitätskarten durch die Suche von Korrespondenzen in Stereobildern existieren zahlreiche Methoden, welche sich in der Qualität der Ergebnisse und der Rechenkomplexität unterscheiden [Beyerer et al., 2016, S. 358]. Diese Methoden lassen sich jedoch anhand einiger Gemeinsamkeiten bei ihrem Vorgehen mit einer allgemeinen Taxonomie beschreiben.

Taxonomie von Stereokorrespondenz-Algorithmen

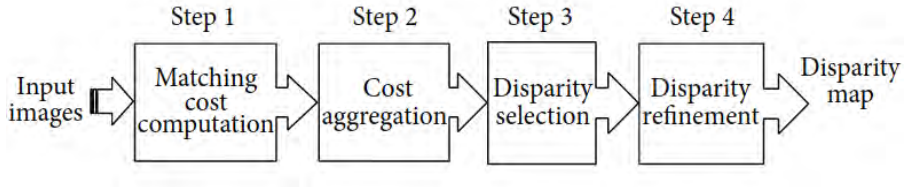


Abbildung 2.5: Die 4 Schritte eines Stereo-Matching Algorithmus. Aus [Hamzah und Ibrahim, 2015, Abb. 2]

.

Als Grundlage für die Erklärung und den Vergleich von Algorithmen zum Finden von Korrespondenz wird sich meistens auf die Taxonomie von Scharstein et al. [2001] bezogen. Dieser Taxonomie nach bestehen solche Algorithmen in der Regel aus vier Schritten (Abb. 2.5), um von einem Paar von Eingabebildern eine endgültige Disparitätskarte zu erstellen. Man unterscheidet dabei zwischen *lokalen* und *globalen* Ansätzen. Lokale Methoden oder auch *Fenster-Methoden* (engl.: window methods) durchlaufen alle 4 Schritte eines Stereo Vision Algorithmus. Für die Berechnung der Disparitäten wird nur die Intensität der einzelnen Werte innerhalb des betrachteten Fensters (*engl.: sliding window*) verwendet. Es werden nur *lokale* Werte für die Berechnung verwendet, was zu einem geringen Rechenaufwand und schneller Ausführungszeit führt, aber auch schlechtere Ergebnisse produziert als mit globalen Methoden.

Globale Methoden formulieren das Erstellen einer Disparitätskarte als ein *Optimierungsproblem* zur Minimierung einer globalen Zielfunktion, bei welchem durch einen iterativen Prozess eine glatte und zutreffende Disparitätskarte ermittelt werden kann [Scharstein et al., 2001]. Die dabei verwendete Energiefunktion hat einen Datenterm, welcher Lösungen bestraft, die nicht zu den betrachteten Daten passen und einen Glättungsterm,

welcher räumlichen Zusammenhalt und damit gleichmäßige Disparitäten erzwingt. Hierbei oft eingesetzte Methoden sind *Graph-Cut* oder *Belief-Propagation*. Globale Methoden überspringen meistens den 2. Schritt „Kostenaggregation". Der Einsatz von globalen Methoden produzierte Disparitätskarten mit hoher Qualität, birgt aber im Vergleich zu lokalen Methoden einen höheren Rechenaufwand.
Um zu verstehen, wie ein Stereo-Matching-Algorithmus funktioniert, werden die vier Schritte erläutert.

1. Schritt - Matching-Kosten Berechnung

Matching-Kosten (*engl.: matching cost*) stellen ein Kriterium da, welches das Ausmaß von Übereinstimmungen zwischen zwei Pixel bzw. Bildausschnitten zu beschreibt [Hamzah und Ibrahim, 2015]. Je geringer die Kosten, desto höher die Wahrscheinlichkeit, dass die verglichenen Pixel bzw. Ausschnitte zueinander passen und möglicherweise dieselbe Stelle in der abgebildeten Szene darstellen. Dafür werden grundsätzlich die Pixelintensitäten verglichen.

Für die Berechnung der Kosten können verschiedene Funktionen wie der *mittlere quadratischer Fehler (engl.: Mean squared error)*, die *Summe der absoluten Differenzen (engl.: Sum of squared differnces)* oder die *normalisierte Kreuzkorrelation (engl.: Normalized cross-correlation)* verwendet werden.

2. Schritt - Kostenaggregation

Bei der Kostenaggregation geht es darum, welche Pixel für die Berechnung der Kosten verwendet werden.
Lokale Methoden verwenden grundsätzlich ein *Fenster* (engl.: window), das den Kontext (auch Support region) der für die Berechnung der Kosten wichtigen Pixel vorgibt [Scharstein et al., 2001]. Dies ist nötig, da die Matching-Kosten eines einzelnen Pixels nicht ausreichen, um eine genaue Zuordnung zu erreichen. Je größer das gewählte Fenster ist, desto größer ist der Kontext der für die Berechnung einbezogen werden kann. Ein größerer Kontext gibt mehr Anhaltspunkte und verbessert die Genauigkeit, erhöht aber auch die benötigte Rechenzeit.
Ein betrachtetes Pixel $p_l = (x, y)$ des linken Bildes wird mit allen Pixel $p_r = (x-d, y)$ für jedes $d \in [Disparitaet_{min}, Disparitaet_{max}]$ verglichen. Das bedeutet, dass das Fenster von $p_r = (x - Disparitaet_{min}, y)$ bis $p_r = (x - Disparitaet_{max}, y)$ bewegt wird und für

jede Stelle die Matching-Kosten berechnet und gespeichert werden. Zur Berechnung der Kosten wird die in Schritt 1 gewählte Funktion verwendet.

Die meisten globalen Methoden *überspringen* diesen Schritt, da sie keine Kosten aggregieren, sondern eine Zuweisung suchen [Scharstein et al., 2001]. Dies wird im folgenden Schritt betrachtet.

3. Schritt - Auswahl der Disparitäten

In diesem Schritt wird anhand der bestimmten Kosten eine Disparitätskarte erstellt. Generell wird bei lokalen Methoden die *Winner-Take-All*-Strategie verwendet, bei der für jedes Pixel die Disparität mit den geringsten Kosten gewählt wird. Dazu wird aus der Menge der im 2. Schritt bestimmten Kosten für ein Pixel p_l nun die Disparität d ausgewählt, deren Matching-Kosten mit $p_r = (x - d, y)$ am niedrigsten sind. Die aggregierten Kosten C' können dabei auch für das ganze Bild als *Kostenvolumen* (engl.: *cost volume*) mit den Dimensionen $Hoehe \times Breite \times Disparitaten$ zusammengefasst werden. Dabei wird nach der Gleichung 2.4 für jedes Pixel das Minimum entlang der Disparitäts-Dimension ausgewählt.

$$d_{p_l} = \underset{d \in D}{arg\,min}\, C'(p_l, d) \tag{2.4}$$

Für globale Methoden ist dies der wichtigste Schritt, da hier der Großteil der Arbeit verrichtet wird. Die Bestimmung der Disparitätswerte wird hier als Optimierungsproblem angesehen. Dafür werden im Allgemeinen zwei Annahmen getroffen:

- Das betrachtet Pixel des linken Bildes und das gesuchte Pixel im rechten Bild sollten eine ähnliche Intensität, das heißt einen ähnlichen Wert besitzen.
- Das benachbarte Pixel der Disparitätskarte sollten ähnliche Disparitätswerte besitzen.

Diese Annahme lassen sich in einer Energiefunktion (Gleichung 2.5, aus [Scharstein et al., 2001, Gl. 3]) darstellen.

$$E(d) = E_{data}(d) + \lambda E_{smooth}(d) \tag{2.5}$$

Der Datenterm E_{data} stellt die Matching-Kosten dar und misst, wie gut die ausgewählten Pixel zusammenpassen.

Der Glättungsterm (engl.: smoothness) $E_{smooth}(d)$ fördert die Zuweisung ähnlicher Werte für benachbarte Pixel, um glatte Verläufe der Disparitäten zu erhalten.

Bei der Umsetzung können verschiedene Verfahren zum Finden des Optimums verwendet werden, wie beispielsweise *belief propagation (BP)* und *graph cut (GC)*. Beide Verfahren arbeiten mit graphischen Modellen, bei welchen die Graphen verwendet werden, um die Beziehung zwischen Pixeln und deren Einfluss aufeinander darzustellen. Hauptsächlich verwendete graphische Modelle sind *Bayes'sche Netze* und *Markov-Netzwerke* (oder auch *Markov-Random-Fields (MRF)*).

4. Schritt - Verbesserung der Disparitäten

Die aus Schritt 3 resultierende Disparitätskarte kann Rauschen oder Fehler wie zum Beispiel ungültige Übereinstimmungen oder Verdeckungen enthalten und wird deshalb noch einmal verbessert [Hamzah und Ibrahim, 2015]. Zur Entfernung von Rauschen wird Regularisierung angewendet. In Regionen, für welche keine oder nur spärlich Disparitäten bestimmt werden konnten, werden die Werte interpoliert. Aus Verdeckung resultierende Lücken werden im Allgemeinen mit Werten gefüllt, welche vergleichbar zu Hintergrund- oder texturlosen Regionen sind oder per Interpolation bestimmt werden.
Typisch für diesen Schritt ist die Nutzung eines *Gauß-* oder *Median-Filters*. Durch das Gauß-Filter wird Rauschen in der Disparitätskarte verringert, jedoch reduziert es auch Details, da diese an die Umgebung angeglichen werden. Der Median-Filter entfernt kleine, isolierte Werte und kann in erweiterten Varianten auch Rauschen entfernen, ohne dass dabei Kanten verloren gehen.

2.1.2 Stereo-Matching Methoden

Basierend auf der gegeben Taxonomie eines Stereo-Matching-Algorithmus werden im Folgenden verschiedene lokale und globale Methoden, so wie eine dazwischenliegende Methode vorgestellt.

Lokale Methoden

Eine Kategorie von lokalen Methoden sind *Block Matching* Methoden [Brown et al., 2003]. Hierbei werden die Disparitäten für einen Punkt im linken Bild durch den Vergleich einer

kleinen Region um den Punkt mit einer Reihe von Regionen im rechten Bild bestimmt. Für den Vergleich der Intensitäten in den Regionen werden Metriken wie beispielsweise normalisierte Kreuzkorrelation oder die Summer der Quadrate verwendet. Ein Alternative ist die Berechnung des *Rangs* (engl.: Rank) der Region. Der Rang einer Region ist die Anzahl der Pixel, deren Intensität geringer ist als die des Pixels im Zentrum. Diese Variante ist weniger anfällig gegenüber Ausreißern. Die einfache Implementation solcher Methoden ist aufgrund redundanter Berechnungen sehr ineffizient [Brown et al., 2003].

Eine andere Art lokaler Methoden sind *Gradienten*-Methoden. Diese nutzen Gradienten, welche den Verlauf der Pixelintensitäten des Bildes repräsentieren. Unter der Annahme, dass die Helligkeit von einem Punkt der Szene in beiden Bildern gleich ist, kann die horizontale Verschiebung mit der differenzierbaren Funktion 2.6 (aus [Brown et al., 2003, Gl. 3.1]) bestimmt werden.

$$(\nabla_x E)v + E_t = 0 \tag{2.6}$$

Dabei ist $\nabla_x E$ die horizontale Komponente des Gradienten, v die Verschiebung zwischen den Sichtpunkten (die Basis) und E_t der Unterschied der Intensitäten zwischen den beiden Bildern. Grundsätzlich können Gradienten-Methoden der Theorie nach nur Disparitäten von einem Pixel bestimmen. Deshalb ist es notwendig, hierarchische Verarbeitung der Bilder mit einzubinden und den Vorgang für verschiedene Auflösungen mit unterschiedlichem Detailgrad durchzuführen.

Belief Propagation

Belief-Propagation (deut.: "Glaubensweitergabe") ist ein Algorithmus zum Lösen bzw. Approximieren von Optimierungsproblemen auf graphischen Modellen. Stereo-Matching kann als solch ein Problem formuliert werden, in dem das Finden von Korrespondenzen als Minimierung einer Energiefunktion angesehen wird. Dadurch fallen Methoden, welche Belief-Propagation nutzen in die Kategorie der globalen Stereo-Matching-Methoden. Die im Folgenden vorgestellte Implementation von Xiang et al. [2012] ist nur eine von vielen.

Die Disparitätskarte wird als gerichteter Graph $G = (V, D)$ mit einer Menge von zufälligen Variablen V und einer Menge von Kanten D dargestellt werden. Für Stereo-Matching wird jedem Pixel $p \in V$ ein Label (deut.: Kennzeichnung) $f_p \in \Xi$, sodass die Energiefunktion 2.7 (aus [Xiang et al., 2012, Gl. 1])

$$\sum_{p \in V} E_p(f_p) + \sum_{(p,q) \in D} E_s(f_p, f_q) \tag{2.7}$$

minimiert wird. Der erste Term E_p stellt die Kosten für die Zuweisung eines Labels f zum Pixel p dar. Der zweite Term E_s gibt die Kosten für die Zuweisung der Label f_p und f_q zu zwei benachbarten Pixeln an.
Die einzelnen Knoten schicken Nachrichten an ihre Nachbarn, welche einen Wert beinhalten, den der jeweilige Nachbar ihrer „Meinung“ nach annehmen sollte. Der Inhalt einer Nachricht ist ein Vektor der Größe N mit der Anzahl der Label $N = ||f||$. Für eine Nachricht M_{pr} von einem Knoten p an sein Nachbarn r, wird jede Komponente $M_{pr}(f_r)$ des Vektors für ein Label f_r aller möglichen Label durch die Gleichung 2.8 bestimmt.

$$M_{pr}(f_r) = \underset{0<=i<N}{min}\Big(\omega_p E_p(f_i) + E_s(f_i, f_r) + \sum_{s \in N(p)} M_{sp}(f_i)\Big) \tag{2.8}$$

Der erste Term ist die Energiefunktion für die betrachteten Label f_i und f_r. Der zweite Term ist die Summe der erhaltenen Nachrichten aus der Nachbarschaft $N(p)\backslash p$, ausgenommen der Nachrichten vom Nachbarn r für das Label f_i. Für Label f_r werden so alle möglichen Kombination mit anderen Labeln durch den Term ausgewertet. Die Kombination mit der geringsten Energie wird im Vektor für das Label f_r an den Nachbarn r geschickt.
Nach einer gewissen Anzahl an Iterationen wird für jeden Knoten das optimale Label f_p^* durch Gleichung 2.9

$$f_p^* = arg \underset{f}{min}\Big(E_p(f) + \sum_{s \in N(p)} M_{sp}(f)\Big) \tag{2.9}$$

bestimmt. Dabei wird das Label ausgewählt, welches die geringste Summe aus den Zuweisungskosten $E_p(f)$ und der Summe der Energien aus den erhalten Nachrichten für das Label $M_{sp}(f)$ ergibt.

Eine weitere Möglichkeit für die Umsetzung einer globalen Methode für Stereo-Matching ist *Graph Cuts*.

Graph Cuts

Graph Cuts ist ein Verfahren zur Partitionierung eines gerichteten Graphen $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ mit einer Menge gewichteter Kanten $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. Es kann genau wie Belief-Propagation zum Approximieren einer Energiefunktion für die Zuweisung von Disparitäten verwendet werden. Die beschriebene Implementation ist von Kolmogorov [2004].

Der verwendete Graph besitzt zwei besondere Knoten, die Quelle s (engl.: source) und die Senke t (engl.: sink), zu denen die Menge der Knoten zugeordnet werden sollen. Dafür wird ein *s-t Cut* durchgeführt, welcher eine Menge von gewichteten Kanten $\mathcal{E}$ schneidet, um die Menge der Knoten $\mathcal{V}$ in zwei getrennte Mengen $\mathcal{S}$ und $\mathcal{T}$ zu teilen [Kolmogorov, 2004]. Die Kosten für den Schnitt ergeben sich aus der Summe der Gewichte aller durchtrennten Kanten von (p, q) mit $p \in \mathcal{S}$ und $q \in \mathcal{T}$. Der minimale Schnitt (engl.: *minimal cut*) ist der Schnitt mit den geringsten Kosten.

Durch das Verfahren lassen sich binäre Label-Probleme wie zum Beispiel Vorder- und Hintergrund Segmentierung lösen. Stereo-Matching stellt allerdings ein Multi-Label-Problem dar, bei welchem die Pixel einem Wert aus einer größeren Menge von Disparitäten zugeordnet werden müssen. Es wird eine Erweiterung benötigt, um GC für Stereo-Matching verwenden zu können. Bei dem *Expansion-Move*-Algorithmus geht es darum, ob ein Pixel sein Label behält oder ändert, wodurch das Multi-Label-Problem zu einem Binär-Label-Probleme wird [Kolmogorov, 2004]. Das Ziel ist es eine Konfiguration f von Label zu finden, welche jedem Pixel ein Label f_p zuweist und die niedrigste Energie ausweist.

Der Graph besitzt Knoten für alle Pixel der Disparitätskarte, welche mit Zufallsvariablen initialisiert werden. Für jede Disparitätsstufe existiert eine Konfiguration f, welche für jeden Pixel eine Zuweisung a (engl.: Assignment) eines Pixel-Paares (p, q) mit der Disparität $d(a) = d(p, q) := q - p$ besitzt. Eine Zuweisung kann entweder aktiv oder inaktiv sein.

Bei Optimierungsprozess wird über die Disparitätsstufen iteriert und für jede Zuweisung geprüft, ob sie aktiv ist und die betrachtete Disparität α dem Wert des Pixel-Paares entspricht $\alpha = d(p, q)$. Sollte sie aktiv sein und einen anderen Wert besitzen oder inaktiv und die betrachtete Disparität haben, wird ein Expansionsschritt *α-expansion move* durchgeführt, bei dem der jeweilige Aktivitätszustand gewechselt wird. Es wird der Expansionsschritt gesucht, welcher die Energiefunktion $E(f) = E_{data} + E_{smooth}$ über alle

Disparitätsstufen am meisten reduziert [Boykov et al., 2001]. Der Term Datenterm (Gleichung 2.10) stellt die Matching-Kosten für die Pixel der Zuweisungen dar. Der Glättungsterm (Gleichung 2.11) misst die Übereinstimmung der Pixel benachbarter Zuweisungen $a_1 \sim a_2$.

$$E_{data}(f) := \sum_{a, f(a)=1} D(a) \tag{2.10}$$

$$E_{smooth}(f) := \sum_{a_1 \sim a_2} V_{a_1,a_2} \tag{2.11}$$

Der optimale Expansionsschritt lässt sich durch das Bestimmen des minimalen s-t-Schnitts (*s-t cut*) anhand der Aktivitätszustände finden. Wenn keine Expansionsschritte mehr zur Reduzierung der Energiefunktion führen, ist das Optimum gefunden.

Semi-global Matching

Eine der verbreitetesten Stereo-Matching-Methoden ist *Semi-global Matching* von Hirschmuller [2008]. Sie bietet eine gute Balance zwischen Genauigkeit und Geschwindigkeit und wurde für die Verwendung in Echtzeit-Systemen auf Intel-Prozessoren, GPUs und FPGAs implementiert und wurde beispielsweise von Daimler für Stereokamerasysteme eingesetzt [Franke und Gehrig, 2015, S. 403].

Die Methode folgt den vier Schritten der Taxonomie von Scharstein et al. [2001], mischt aber Eigenschaften globaler und lokaler Methoden.

Die Berechnung der Kosten basiert auf „gemeinsamen“ (engl.: *mutual*) Informationen (MI), da diese von Beleuchtungsänderungen und Fehlern bei der Kalibrierung unbeeinflusst sind. Die gemeinsamen Informationen beschreiben die Ähnlichkeit des Bildinhaltes von zwei Bildern oder Ausschnitten dieser Bilder anhand ihres Informationsgehaltes. Sie werden aus der jeweilige *Entropie* der einzelnen Bilder H_{L_1},H_{L_2} und der gemeinsamen Entropie H_{L_1,L_2} bestimmt (Gleichung 2.12, aus [Hirschmuller, 2008, Gl. 1]) .

$$MI_{L_1,L_2} = H_{L_1} + H_{L_2} - H_{L_1,L_2} \tag{2.12}$$

Die Entropie ist ein Maß für den Informationsgehalt eines Bildes und wird anhand der Wahrscheinlichkeitsverteilung der Pixelintensitäten berechnet. Eine hohe Wahrscheinlichkeit für eine Pixelintensität steht für ein vermehrtes Vorkommen dieser Intensität.

Mit diesem Maß können Bildausschnitte verglichen werden, ohne direkte Pixel-zu-Pixel Vergleiche durchzuführen.
Die gemeinsame Entropie wird mit dem linken Bild und einer verzerrten (engl.: warped) Rekonstruktion des rechten Bildes und einer Disparitätskarte erstellt. Dadurch befinden sich zugehörige Pixel an der gleichen Stelle und können verglichen werden. Zu Beginn wird eine zufällige initiale Disparitätskarte erstellt, mit der die Rekonstruktion erstellt werden kann. Aus dem Vergleich der beiden Bilder mit den MI als Kostenfunktion wird eine neue Disparitätskarte erstellt. Es werden insgesamt drei Iterationen mit Bildern geringer Auflösung durchgeführt, die bei jeder Iteration um den Faktor 8 reduziert werden. Dies macht den Vorgang effizienter.

Auf der initialen Disparitätskarte werden daraufhin weitere Kostenaggregation durchgeführt. Hierfür wird die Energiefunktion $E(D)$ (Gleichung 2.13, aus [Hirschmuller, 2008, Gl. 11]) für die Disparitätskarte D verwendet.

$$E(D) = \sum_{p} \Big(C(\mathbf{p}, D_p) \sum_{q \in N_p} P_1 \mathrm{T}[|D_p - D_q| = 1] + \sum_{q \in N_p} P_2 \mathrm{T}[|D_p - D_q| > 1] \Big) \tag{2.13}$$

Der erste Term ist die Summer alle Kosten für die Disparitäten von D. Der Zweite Term wendet eine kleine, konstante Strafe P_1 an, für alle Pixel q der Nachbarschaft N_p von Pixel p, für die sich die Disparität um ein Pixel $\mathrm{T}[|D_p - D_q| = 1]$ ändert. Der Operator T ergibt 1, wenn die Bedingung stimmt und ansonsten 0.
Der Einsatz der kleinen Strafe ermöglicht die Anpassung an gekrümmte oder schräge Oberflächen. Der dritte Term ist eine große, konstante Strafe P_2 für alle größeren Abweichungen $\mathrm{T}[|D_p - D_q| > 1]$, durch welche Diskontinuitäten der Disparitäten verhindert werden sollen. Das Finden einer optimalen Minimierung über den gesamten 2D-Bereich des Bildes ist jedoch *NP-vollständig* das heißt (vermutlich) nicht in polynomialer Zeit lösbar [Hirschmuller, 2008].
Die Lösung ist die Anwendung der Funktion nur auf eindimensionale Pfade in 16 verschieden Richtungen um ein Pixel herum zu begrenzen. Die Abbildung 2.6 zeigt rechts die Verteilung der Pfade für einen Pixel und links einen Graphen des Verlaufes der Disparitäten entlang eines Pfades. Für einen Pixel p und die Disparität d wird die Disparität aus der Summe aller minimalen 1D Kosten-Pfade $L_r(p, d)$ berechnet, welche im Pixel $p - d$ enden. Die maximale Länge ist dabei $C_{max} + P_2$, der Summe der maximalen möglichen Kosten und der großen Strafe P_2.

Für jedes Pixel im linken Bild p wird dies für alle Pixel $p = (p - d)$ für alle möglichen Disparitäten d des rechten Bildes durchgeführt.

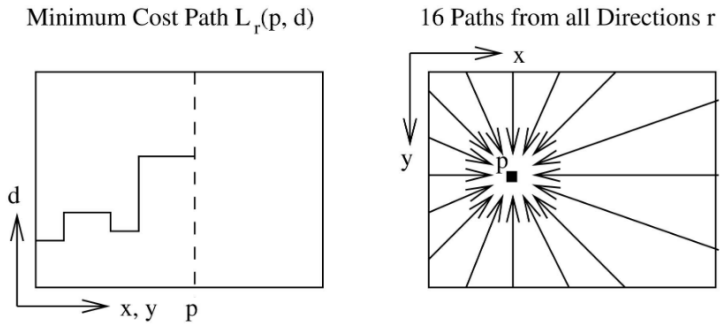


Abbildung 2.6: Links: Der minimale Kostenpfad $L_r(p, d)$. Rechts: Visualisierung aller Pfade für den Pixel p.

Daraus ergibt sich ein dreidimensionales Kostenvolumen, aus dem die endgültigen Disparitäten nach dem Winner-Takes-All Prinzip (s. Abschnitt 2.1.1 3. Schritt - Auswahl der Disparitäten) bestimmt werden. Nach verschiedenen Methoden zur Verbesserung der Disparitäten durch das Füllen von Lücken und Filtern von Ausreißern erhält man eine finale Disparitätskarte.

Die SGM-Methode wird viel verwendet und als Vergleich zu anderen Methoden herangezogen. Auch orientieren sich viele Arbeiten an dem Verfahren, um eigene Methoden zu entwickelten, darunter auch einige Deep Learning Ansätze. Bevor diese Methoden erklärt werden können, werden zuerst die nötigen Begriffe und Grundlagen erläutert.

2.2 Deep Learning

Der Begriff *maschinelles Lernen (ML)* (engl.: machine learning) ist ein Oberbegriff für den Themenbereich der Imitation von menschlichem Lernen von Computer-Systemen. Ein System besitzt die Fähigkeit zu lernen, wenn es durch Erfahrung eigenständig Muster und Zusammenhänge in Daten erkennen kann und diese Informationen nutzt, um sich selbst zu verbessern [Zhang, 2020].
Es existieren viele verschiedene ML-Methoden, von denen eine verbreitete das Lernen mit *künstlichen Neuralen Netzwerken (KNNs)* (engl.: artificial neural networks (ANNs)) ist.

2.2.1 Künstliche Neuronale Netzwerke

Mit künstlichen neurale Netzwerken wird der Aufbau des menschlichen Gehirns als komplexes Netz von Neuronen nachgeahmt. Sie bestehen aus künstlichen Neuronen, die aus einer Menge an Eingabewerten (engl.: inputs) X, den eigenen Gewichten (engl.: weights) W, einem Bias-Wert b und einer Aktivierungsfunktion θ eine Ausgabe (engl.: output) Y erzeugen. In Abbildung 2.7 ist eine bildliche Darstellung eines künstlichen Neurons zu sehen. Mathematisch lässt sich dies mit der Gleichung 2.14 beschreiben, in der die Anwendung der Aktivierungsfunktion auf eine gewichtete Summe der Eingaben und der jeweiligen Gewichte und dem zusätzlichen Bias die Ausgabe ergibt.

$$Y = \theta\Big(\sum_{i=1}^{n} W_i X_i + b\Big) \tag{2.14}$$

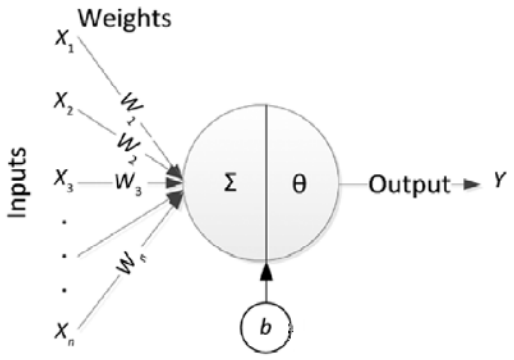


Abbildung 2.7: Darstellung eines Neurons. Aus [Awad und Khanna, 2015, Abb. 7-2]

Die Wahl der Aktivierungsfunktion beeinflusst das Aktivierungsverhalten des Neurons und sollte mit Hinblick auf das erwünschte Zielverhalten getroffen werden. Der Bias sorgt für die Verschiebung der Aktivierungsfunktion, sodass die Ausgabe beispielsweise nie null sein kann. Dieser wird aber nicht immer verwendet. Einzelne Neuronen können in Schichten (engl.: layer) mit unterschiedlichen Größen angeordnet werden.
In einem klassischen *feed-forward* Netzwerk sind die Ausgaben einer Schicht die Eingaben der folgenden Schicht. Die erste Schicht wird als Eingabeschicht (engl.: input layer), die letzte als Ausgabeschicht (engl.: output layer) und die dazwischen liegenden als versteckte Schichten (engl.: hidden layer) bezeichnet. Eine Eingabe wird von der ersten Schicht durch das ganze Netzwerk zur Ausgabeschicht propagiert und resultiert in der Ausgabe des Netzes.

Der Lernprozess findet hierbei durch das Verändern der Gewichte der Neuronen statt. Das Ziel ist es die optimalen Gewichte zu finden, um die bestmöglichen Ausgaben zu erhalten [Awad und Khanna, 2015, S. 133]. Dies wird durch die Minimierung einer Kostenfunktion erreicht, welche die Ausgabe des Netzes bewertete und zur Anpassung der Gewichte verwendet wird. Während des Trainingsprozesses lernt das Netz anhand von Beispielen aus Trainingsdaten. Diese Daten können beispielsweise einzelne Zahlen, Zahlenfolgen oder auch Pixel-Repräsentationen von Bildern sein.
Zusätzlich wird das Netz mit anderen Daten getestet, welche es nicht aus dem Training kennt. Gute Ergebnisse auf beiden Datensätzen zeigen, dass keine *Überanpassung* (eng.: overfitting) auf den Trainingsdaten stattgefunden hat und diese nur auswendiggelernt wurden.
Die Fähigkeit eines Netzes, auf unbekannten Daten richtige Ausgaben zu erzeugen, wird als *Generalisierungsfähigkeit* bezeichnet. Auf welche Art das Netz lernt, was es tun soll, hängt von dem verwendeten Lern-Algorithmus ab.

Lern-Algorithmen

Es existieren verschiedenen Lern-Algorithmen wie zum Beispiel *überwachtes Lernen*, *unüberwachtes Lernen* und *semi-überwachtes Lernen* [Awad und Khanna, 2015, S. 6-8]. Beim überwachten Lernen (engl.: supervised learning) bestehen die Trainingsdaten aus Eingabedaten und gekennzeichneten (engl.: labeled) Zieldaten. Das Netzwerk erzeugt mit den Eingabedaten eine Ausgabe, welche mit der Zielausgabe verglichen wird.
Der Fehler zwischen Ist- und Soll-Ausgabe wird zur Anpassung der Gewichte verwendet. Ziel ist es, diesen Fehler zu minimieren. Dafür wird zumeist der *Backpropagation*-Algorithmus eingesetzt, mit dem ein *Fehlergradient* bestimmt wird, mit welchem die Gewichte der Neuronen abhängig von ihrem Einfluss auf die Ausgabe angepasst werden. Dieser Lern-Algorithmus wird für das Training von Netzen für Klassifizierung oder Regression eingesetzt.

Das unüberwachte Lernen (engl.: unsupervised learning) eignet sich gut für das Lernen von versteckten Strukturen in Daten, für welche die Zielausgabe nicht bekannt ist. Das Netz soll die Zusammenhänge der Eingaben lernen, ohne das vorgegebenen wird, welche Eingabe zu welcher Ausgabe führen soll. Ein Beispiel hierfür ist das *Clustering* von Daten nach ihrem Ähnlichkeitsgrad ohne Vorgabe der Klassen. Der Vorteil von unüberwachtem Lernen ist, dass keine gekennzeichneten Trainingsdaten benötigt werden.

Semi-überwachtes Lernen (engl.: semi-supervised) ist eine Zwischenstufe von überwachtem und unüberwachtem Lernen. Dabei werden die ungekennzeichneten Trainingsdaten um eine kleine Menge an gekennzeichneten Daten erweitert. Dies führt zu besserer Genauigkeit im Vergleich zu unüberwachtem Lernen und erfordert weniger Aufwand in der Beschaffung der Trainingsdaten als bei überwachtem Lernen.

Die Wahl des Lern-Algorithmus ändert zwar die Anforderung an die für das Training bereitgestellten Daten ändert jedoch nichts an der Komplexität der zu erlernenden Muster der Daten. Für das Erlernen von komplexeren Daten stoßen herkömmliche Netze oft an ihre Grenzen.

Tiefe Netzwerke

Die Fähigkeit eines Netzes, Daten durch die Verwendung nicht-linearer Aktivierungsfunktionen linear trennen zu können, wird mit steigender Anzahl der Schichten besser [Aggarwal, 2018, S. 34]. Flache Netze sind deshalb oftmals nicht in der Lage eine gute Generalisierung für komplexere Daten zu lernen. Die Verwendung von tiefen neuronalen Netzwerken (engl.: deep neural networks) wird als *deep learning* bezeichnet. Tiefe Netze verwenden viele versteckte Schichten, durch deren hierarchischen Aufbau die Daten mit unterschiedlichen Abstraktionsgeraden verarbeitet werden. Sie sind in der Lage, komplexere Muster zu erkennen, welche sich aus den simpleren Mustern der oberen Schichten ergeben [Aggarwal, 2018, S. 34].

Im Bereich der Computer Vision ist Deep Learning eine effektive Methode für die Verarbeitung von Bildern im Hinblick auf Aufgaben wie Bildklassifizierung, Objekterkennung oder semantische Bildsegmentierung. Die Verwendung von herkömmlichen Netzwerken für diese Ausgaben bringt aber mehrerer Probleme mit [Qiu et al., 2021]. Für die Verarbeitung müssen die Bilder als eindimensionaler Vektor in das Netz gegeben werden, wodurch räumlich Informationen verloren gehen. Zudem benötigen die Netze zu viele Parameter, um etwas lernen zu können, was wiederum leicht zu *overfitting*, der Überanpassung auf die Trainingsdaten führen kann. Eine spezielle Art tiefer neuronaler Netzwerke, welche dafür entwickelt wurden, sind *Convolutional Neural Networks (CNNs)* (deut.: Faltungsnetzwerke). Die grundsätzliche Struktur und deren Kernelemente werden im folgenden Abschnitt kurz erläutert.

2.2.2 Convolutional Neural Networks

Der Kernbaustein von CNNs sind Faltungsschichten (*convolutional layers*), in welchen die Neuronen in einem dreidimensionalen Raster angeordnet sind [Aggarwal, 2018, S. 318]. Die drei Dimensionen sind die räumlichen Dimensionen der Höhe und Breite und der Tiefe bzw. Anzahl der *feature maps (FM)* (deut.: Merkmalskarten). Eine Feature-Map oder auch *activation map* ist die Ausgabe einer Faltungsschicht, welche unterschiedliche Merkmale des Bildes repräsentiert. Diese Ausgabe wird durch die namensgebende Faltungsoperation erzeugt. Eine Faltung ist die Berechnung des Skalarprodukts von einem Filterkern und dem durch ihn bestimmten Bildausschnitt bzw. Ausschnitt einer Feature-Map.
Der Filterkern oder auch Kernel ist eine Anordnung von Gewichten in einem Raster, welche die trainierbaren Parameter der Faltungsschicht ausmachen. Dieser ist in der Höhe und Breite meistens kleiner als die Schicht selber, doch hat er dieselbe Tiefe wie die Schicht.
Die Eingabe-Feature-Maps der Schicht werden mit dem Filter gefaltet, wobei der Filter auf jede gültige Position der Feature-Map angewendet wird. Die resultiert in einer Reduktion der Größe der Ausgabe-Feature-Map, die abhängig von der Größe und Schrittweite (engl.: stride) des Filters ist. In dem in 2.8 dargestellten Beispiel wird eine Feature-Map der Größe 7×7 mit einem 3×3 Filterkern und einer Schrittweite von 1 gefaltet. Die ent-

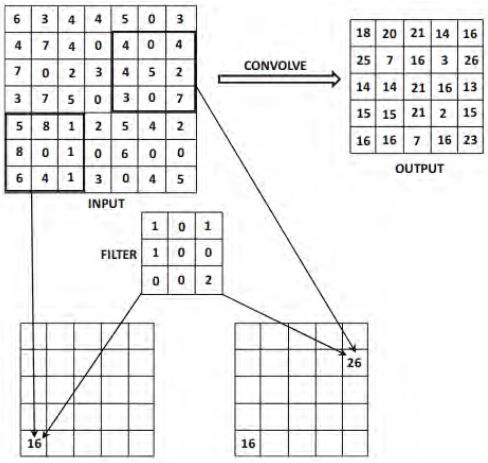


Abbildung 2.8: Visualisierung des Vorgangs einer Faltung von einer 7×7 Feature-Map mit einem 3×3 Filter.

stehende Feature-Map besitzt die Größe 5×5, da die äußersten Pixel durch die Größe des Filterkerns keine gültigen Positionen sind. Durch diese Reduktion gehen Informationen

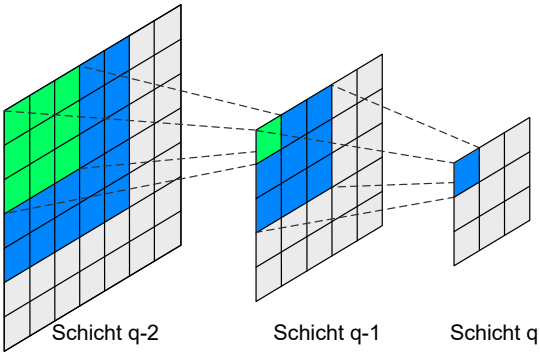


Abbildung 2.9: Visualisierung des rezeptiven Feldes eines Neurons in Schicht q.

an den Bildrändern verloren. Um dies zu verhindern, kann *padding* (deut.: Polsterung) eingesetzt werden. Dabei wird die Feature-Map rundherum mit Nullen aufgefüllt, sodass die originalen Randwerte gültige Positionen sind. Im Falle des Beispiels aus Abbildung 2.8 müsste ein *padding* von 1 für eine Schicht von Nullen verwendet werden.

Durch die Kaskadierung mehrerer Faltungsschichten werden die Information des Eingabebildes komprimiert und auf kleinere Repräsentationen heruntergebrochen. Wie groß der „Sichtbereich“ eines Neurones einer Schicht in dem Originalbild ist, wird als das *rezeptive Feld* (engl.: receptive field) bezeichnet [Aggarwal, 2018, S. 322]. Die Größe ist von den Faltungen der vorherigen Schichten und den dabei verwendeten Konfigurationen abhängig. Das rezeptive Feld eines Neurons der Schicht q nach Anwendung von drei vorherigen Faltungen mit einem Filterkern der Größe 3×3 besitzt die Größe 7×7, wie in Abbildung 2.9 dargestellt ist. Durch die erste Faltung ist das Feld 3×3 groß, nach der zweiten 5×5 und nach der dritten 7×7. Ein großes rezeptives Feld, in welchem viel des Eingabebildes enthalten ist, ermöglicht das Erkennen von komplexeren Strukturen [Aggarwal, 2018, S. 322].

Eine Methode zur effektiven Erweiterung des rezeptiven Feldes sind *dilated Convolutions (deut.: geweitete Faltungen)* (manchmal auch *atrous* Faltungen [Xu et al., 2022]). Sie ermöglichen eine simple Vergrößerung des rezeptiven Feldes ohne die Reduzierung der Größe und mit weniger Parametern und Operationen als mit normalen Faltungen [Wang et al., 2019]. Dafür wird der Kernel um die *Rate* r „geweitete“, sodass die vom Kernel abgedeckten Pixel nicht direkt aneinandergrenzen, sondern Lücken entstehen.
In Abbildung 2.10 sind zwei geweitete Filterkerne mit einer Rate von 2 und 3 im Vergleich zu einem normalen Filterkern mit einer Rate von 1 dargestellt. Eine Faltung mit einem

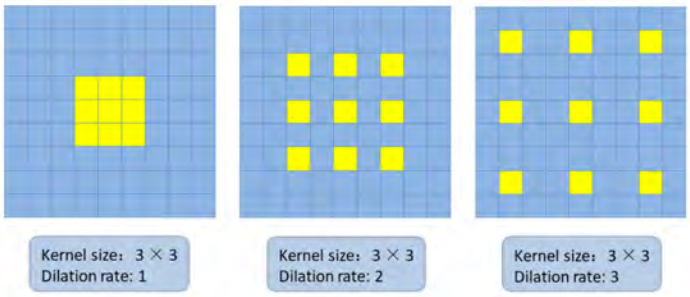


Abbildung 2.10: Drei Filterkerne der Größe 3×3 mit unterschiedlichen Raten von 1,2 und 3.

$3x3$ Kernel resultiert in einem rezeptiven Feld von $3x3$. Eine geweitete Faltung mit dem gleichen Kernel und einer *Rate* von 2 ergibt ein rezeptives Feld der Größe $5x5$ und mit einer Rate von 3 ein $7x7$ großes rezeptives Feld.

Des Weiteren lässt sich das rezeptive Feld auch durch die Verwendung von *max pooling* erweitern [Aggarwal, 2018, S. 324,326]. Durch die Max-Pooling-Operation wird wie bei einer Faltung für jede gültige Position ein rechteckiger Ausschnitt der Eingabe-FM betrachtet. Anstatt den Wert der aktuellen Position für die Ausgabe-FM durch die Faltung zu bestimmen, wird der maximale Pixelwert des Ausschnittes übernommen. Dies hebt herausstechende Features weiter hervor und verringert die räumliche Distanz zwischen diesen Features. Die standardmäßig verwendete Größe für diesen Ausschnitt ist 2×2.

Einige Anwendungen wie zum Beispiel Bildgeneration oder Bildrekonstruktion erfordern es, dass aus der heruntergebrochenen Repräsentation eine Ausgabe rekonstruiert wird. Dafür werden *transponierte Faltungen* (engl.: transposed convolutions) eingesetzt, welche zum *Upsampling* der Eingabe-FM führt. Als Gegenstück zur normalen Faltung wird dafür ein transponierter Kernel verwendet. Für die Eingabe-FM wird padding angewendet, wobei auch zwischen den Pixeln Nullen oder interpolierte Werte eingefügt werden. Dadurch kann mit dem Filterkern eine größere Fläche abgelaufen werden als bei der originalen Feature-Map, weshalb die resultierende Ausgabe-Feature-Map größer ist.
Die Visualisierung dieses Vorgangs sieht grundsätzlich wie die in Abbildung 2.10 geweitete Faltung aus, nur das die gelben Kästchen hierbei die Pixel der originalen Eingabe-FM und die blauen Kästchen die durch das padding hinzugefügten Werte sind. Bei dem mittleren Beispiel wird ein padding von 1 und im rechten ein padding von 2 verwendet. Zusätzlich werden auch *unpooling*-Operationen als Gegenstück zur Pooling-Operation eingesetzt, um eine einfache Vergrößerung der Feature-Maps zu erreichen.

Für tiefe CNNs existiert allerdings die Gefahr des *vanishing* (deut.: verschwindenen) oder *exploding gradient* (deut.: explodierenden) Problems, bei welchem durch den Backpropagation-Algorithmus bestimmte Gradient exponentiell sinkt oder steigt. Als Folge werden Gewichte gar nicht oder viel zu stark angepasst. Eine Lösung existiert in dem Ansatz des *residual learning* mit *residual neural networks (ResNets)* [He et al., 2016].

Residual Learning

Anstatt einer festen hierarchischen Struktur, ist es dem Netz möglich, Schichten mit *skip-connections* zu überspringen, wodurch die Eingabe ohne Änderung als Ausgabe weitergegeben wird [Aggarwal, 2018, S. 349]. Dies wird auch als *Identity-Mapping* bezeichnet, da die Identität der Eingabe nicht verändert wird. Durch das Überspringen beim Erzeugen der Ausgabe können diese Schichten auch bei der Backpropagation ausgelassen werden, da sie nicht zur Ausgabe beigetragen haben. Dies verringert die Gefahr vom Verschwinden oder Explodieren des Gradienten.

Dieser Ansatz erlaubt dem Netz zu „wählen“, welchen Grad an Abstraktion eine Eingabe benötigt, sodass nicht jede Eingabe von denselben Schichten verarbeitet wird. Daher eignet sich der Ansatz zudem sehr gut für Anwendungen, bei denen die Bilder unterschiedlich komplexe Merkmale aufweisen. ResNets werden in vielen Architekturen als Teilmodule integriert.

Als Teilbereich der Computer-Vision wurde auch für Stereo-Vision an dem Einsatz von Convolutional Neural Networks für die Erstellung von Disparitätskarten geforscht. Die Besonderheit ist hierbei, dass mit zwei Eingabebildern gleichzeitig gearbeitet werden muss, deren zusammenhängende Merkmale erkannt werden müssen. Die Entwicklung verschiedener Netzwerkarchitekturen und der verwendeten Methodiken, um Korrespondenzen zwischen den Bildern zu finden, werden im Folgenden erläutert.

2.3 Stereo-Matching mit Deep Learning

Zu Beginn der Forschung für Stereo-Matching mit neuronalen Netzwerken lag der Fokus auf der Verbesserung der einzelnen Schritte von der herkömmlichen Stereo-Pipeline (s. 2.1.1) [Poggi et al., 2021]. Ein der wichtigsten Arbeiten, welche für viele folgende Arbeiten

den Grundstein legt, ist die in 2015 von Žbontar und LeCun [2015] zur Verwendung von CNNs für Stereo-Matching. Sie verwendeten für die Berechnung der Matching-Kosten ein 8-schichtiges CNN, welches die Ähnlichkeit von zwei Bildausschnitten verglich.
Dieses Netzwerk war noch keine *End-to-End* Lösung für die Bestimmung von Disparitäten, sondern realisierte den ersten Schritt eines Stereokorrespondenz-Algorithmus, der Berechnung der Matching-Kosten. Mit dem Begriff End-to-End wird ein Ansatz oder Netzwerk bezeichnet, welches eine Aufgabe vollständig erfüllt und keine zusätzlichen Schritte in der Vor- oder Nachbereitung benötigt werden.
Die Ausgabe des Netzes war einen Ähnlichkeitsgrad (engl.: *similarity score*), welcher zur Initialisierung der Matching-Kosten genutzt wurde. Mit den Kosten wurden die weiteren drei Schritte mit traditionellen Methoden durchgeführt, um eine endgültige Disparitätskarte zu generieren. In einer folgenden Arbeit in 2016 [Žbontar und LeCun, 2016], stellten sie basierend auf der ihrer Arbeit eine schnelle (MC-CNN-fst) und eine genaue (MC-CNN-acrt) Architektur vor.
Die durch überwachtes Training trainierten Netzwerke erreichten in der Auswertung auf dem KITTI 2012 Datenset eine maximale Fehlerrate von 2,61 % und einer minimalen Fehlerrate von 2,43 % und schlugen damit alle anderen nicht CNN-basierten Methoden [Žbontar und LeCun, 2016]. Diesen Ergebnissen folgten viele weitere Arbeiten, welche andere Methoden und Strukturen hinzufügten oder veränderten, um Genauigkeit und Geschwindigkeit zu verbessern. Nach den Erfolgen einiger End-to-End Netzwerke wechselte der Fokus zu dieser Art von Netzwerken[Poggi et al., 2021].

Für die Bewertung der Qualität von Stereo-Matching-Algorithmen werden verbreitet die Metriken des *End-Point-Error* (EPE) und des *D1*-Errors verwendet [Zhou et al., 2020]. Der EPE ist der durchschnittliche absolute Fehler (engl.: mean absolute error (MAE)) zwischen der Schätzung und dem Groundtruth in Pixeln. Beim D1-Error wird überprüft, ob die Abweichung des EPE größer als 3 px und größer als 5% der maximalen Disparität ist (Gl. 2.15).

$$D1 := 3\ px < |D_{gt} - D_{est}| > |D_{gt} \cdot 0.05|\ px \tag{2.15}$$

Dabei kann noch zwischen dem Fehler für verdeckte und nicht verdeckte Bereiche, welche nur in einem Bild zu sehen sind, unterschieden werden. Mit Objekt-Masken können die Fehler zudem noch separat für Vordergrund- und Hintergrund-Objekte berechnet werden.

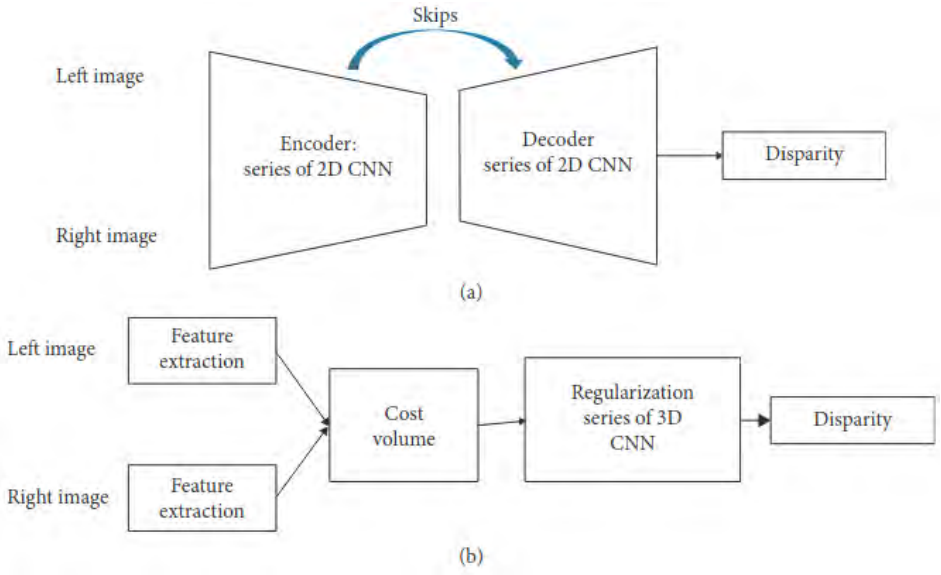


Abbildung 2.11: Vereinfachte Darstellung der zwei verbreiteten Ansätze für End-to-End Stereo-Matching Netzwerke. (a) 2D-Encoder-Decoder Architekturen. (b) 3D-Architekturen mit Kostenvolumen. Abb. aus [Zhou et al., 2020, Abb. 4]

End-to-End Netzwerke für Stereo-Matching lassen sich grundsätzlich in die zwei Kategorien 2D- und 3D-Architekturen einteilen [Poggi et al., 2021]. In den nächsten Abschnitten werden die Eigenschaften und Beispiele der jeweiligen Architekturen beschrieben.

2D-Architekturen

Bei den 2D-Architektur werden zumeist 2D-Encoder-Decoder Strukturen eingesetzt, welche eine u-förmige Anordnung von 2D-CNNs ist. Eine vereinfachte Darstellung solcher Netze bietet Abbildung 2.11a. Die Eingabe wird zuerst mit abnehmender Schichtgröße in der Dimension reduziert, um danach durch *Upsampling* (deut.: Abstastratenerhöung) eine Disparitätskarte zu erzeugen. Durch den Encoder werden die wichtigsten Informationen der Eingabebilder extrahiert und in eine Repräsentation mit geringer Auflösung gebracht. Wichtig ist es hierbei, das rezeptive Feld zu erweitern, um genug Bildkontext mit einzubeziehen. Im Decoder wird aus der Repräsentation mit zusätzlichen Informationen aus dem Encoder die Ausgabe erzeugt.
Das erste *End-to-End* Netzwerk für Stereo-Matching mit dieser Architektur war DispNet von Mayer et al. [2016b]. Sie entwickelten zudem eine sogenannte Korrelationsschicht

(eng.: correlation layer), mit welcher die Ähnlichkeit zwischen zwei den Feature-Maps der beiden Bilder berechnet wurden. Das Netz mit dieser Schicht wurde zur DispNet-C Variante.
Eine weitere Architektur, welche darauf aufbaute, ist die *cascade residual learning (CRL)* Architektur von Pang et al. [2017], welche aus zwei Phasen besteht, die beide der Struktur von DispNet-C folgen. Zuerst wird eine initiale Schätzung der Disparitäten erstellt, für welche darauffolgend die Verfeinerung durchgeführt wurde. Nach der ersten Phase wurde mit dem linken Bild und den geschätzten Disparitäten eine verzerrte (engl.: warped) Rekonstruktion des rechten Bildes erstellt. Das zweite Netzwerk berechnete, welche Korrekturen für die Disparitäten nötig sind, damit die Rekonstruktion besser zum Originalbild passt.
Netzwerke mit 2D-Architekturen sind grundsätzlich leichtgewichtiger als die Varianten mit 3D-Architekturen. Dafür verlieren sie im Vergleich oftmals bei der Genauigkeit der Ergebnisse [Poggi et al., 2021].

3D-Architekturen

Netzwerke, welche dem Ansatz von 3D-Architekturen folgen, setzten auf 3D-Faltungen, um ein *cost volume* (deut.: Kostenvolumen) zu filtern und die endgültigen Disparitäten zu formen. Das Kostenvolumen bietet eine initiale Messung der Ähnlichkeit zwischen dem linken und rechten Bild, welche in den folgenden Netzwerk-Modulen verfeinert werden kann. Mit diesem Ansatz orientiert man sich an dem Vorgehen der lokalen Methoden (s. Abschnitt 2.1.2), wo ein Kostenvolumen die Matching-Kosten des Bildes für alle Disparitäten darstellt. Die Netzwerke setzten in der Regel die drei Schritte Feature-Extraction, Erstellung des Kostenvolumens und Kostenaggregation (bzw. Regularisierung) des Kostenvolumens um. Es existieren verschiedene Arten von Kostenvolumen, welche sich durch ihre Erstellung unterscheiden lassen und unterschiedliche Vor- und Nachteile mit sich bringen [Xu et al., 2022].

Die erste Art ist das *Korrelationsvolumen* (engl.: *correlation volume*), mit welchem direkte Ähnlichkeiten zwischen den Eingabebilder bestimmt werden können. Standardmäßig wird es durch das Zusammenrechnen beider Bilder für alle Disparitäten erstellt. Als Resultat erhält man eine leichtgewichtige Messung der Ähnlichkeit der Größe $S_{FM} \times W_{FM} \times H_{FM} \times D$ für die der Breite W_{FM}, Höhe H_{FM}, der Anzahl der Feature-Maps S_{FM} und den Disparitäten D. Diese Art von Volumen benötigt wenig Speicher und hat einen vergleichsweise niedrigen Rechenaufwand, da nur 2D Faltungen angewendet werden

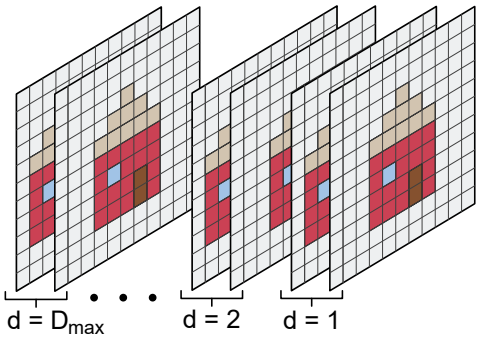


Abbildung 2.12: Visualisierung der Erstellung eines Konkatenationsvolumens durch Konkatenation der Feature-Maps für alle Disparitäten.

müssen. Nachteil ist, dass durch die starke Komprimierung auf wenige Dimensionen viele Informationen verloren gehen.

Für das GC-Net (Geometry and Context Network) [Kendall et al., 2017] wurde die Variante des *Konkatenationsvolumens* (engl.: *concatenation volume*) verwendet, bei welchem die Feature-Maps der beiden Bilder zu einem 4D Volumen konkateniert werden. Im Vergleich zum Korrelationsvolumen erhält man dadurch ein 4D Volumen mit den Dimensionen $2 \cdot S_{FM} \times W_{FM} \times H_{FM} \times D$. Durch das Volumen bleiben viele geometrische Informationen erhalten, da beide Feature-Maps voll verfügbar sind. Eine simple Illustration eines 4D Volumen ist in Abbildung 2.12 dargestellt.
Der Nachteil dieser Variante ist, dass ein Konkatenationsvolumen initial nicht den Grad der Ähnlichkeit zwischen den Bildern repräsentiert. Dafür ist es nötig, Kostenaggregation durch die Nutzung von 3D-Faltungen durchzuführen. Zusätzlich zu der größeren Speicheranforderung des Volumens kommt somit auch einen höheren Rechenaufwand.

Die dritte Kategorie kombiniert beide vorherigen Ansätze zu einem *kombinierten Volumen* (engl.: *combined volume*), um die Vorteile beider Varianten auszunutzen. Bei diesem Ansatz, vorgestellt in GwcNet [Guo et al., 2019b], wird ein gruppenweises Korrelationsvolumen (engl.: *group-wise correlation volume*) mit einem kompakten Konkatenationsvolumen konkateniert, welches dann in einem *3D aggregation Netzwerk* für Kostenaggregation verarbeitet wird. Das gruppenweise Korrelationsvolumen entsteht durch Aufteilung der Features in Gruppen, für die einzeln die Korrelation der Feature-Maps berechnet werden und dann in ein Volume zusammengefügt werden, wodurch mehr Informationen erhalten bleiben als bei einem normalen Korrelationsvolumen.

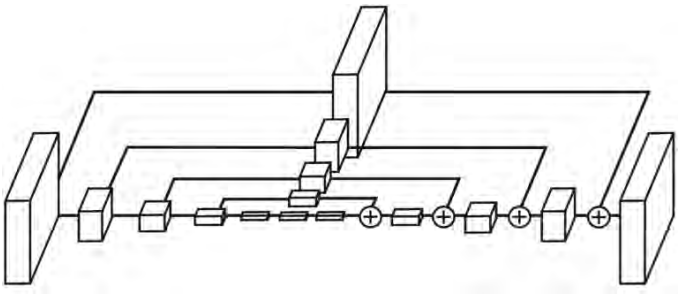


Abbildung 2.13: Einfache Darstellung eines einzelnen Hourglass-Modules. Aus [Newell et al., 2016, Abb. 3].

Um die besten Disparitäten aus dem Volumen bestimmen, werden dieselbe zu einem Wahrscheinlichkeitsvolumen (*probability volume*) umgewandelt. Dafür wird die Softmax-Funktion (Gleichung 2.16, [Aggarwal, 2018, Gl. 1.12]) auf alle geschätzten Kosten c_d angewendet.

$$\Phi(\overline{v})_i = \frac{\exp(v_i)}{\sum_{j=1}^{k} \exp(v_j)} \forall \in \{1, ..., k\} \tag{2.16}$$

Dabei ist $\Phi(\overline{v})_i$ die i-te Ausgabe aller Ausgaben der Schicht $\overline{v}$ und k die Anzahl der Klassen, welche in diesem Fall alle möglichen Disparitäten sind. Das Wahrscheinlichkeitsvolumen gibt für jede Kombination aus Pixel und Disparität die Wahrscheinlichkeit an, dass die Disparität die richtige für das Pixel ist. Um gleichmäßige Übergänge zwischen den Pixeln zu erhalten, wird in den meisten Fällen die *soft-armgin* Funktion (Gleichung 2.17, [Kendall et al., 2017, Gl. 1])

$$soft\ argmin := \sum_{d=0}^{D_{max}} d \times \Phi(-c_d) \tag{2.17}$$

verwendet.

Eine bei den 3D-Architekturen oft verwendete Subnetz-Architektur ist die von *stacked 3D-hourglass* CNNs. Die Hourglass-Architektur (engl.: Sanduhr) beschreibt einen Aufbau mit einer symmetrischen Anordnung von Faltungs-, Pooling- und transponierten Faltungs- und Unpooling-Schichten in Form einer Sanduhr [Newell et al., 2016]. Ein vereinfachte Darstellung ist in Abbildung 2.13 zu betrachten. Dabei werden die Feature-Maps durch Faltungen- und Max-Pooling-Schichten stark in der Größe reduziert. Dies ermöglicht den Vergleich von Features über den gesamten Bildbereich und das Erfassen

von Informationen verschiedener Größen.
Gleichzeitig gibt es vor jeder Max-Pooling-Schicht eine Abzweigung, bei der die Feature-Map separaten mit weiteren Faltungen verarbeitet wird. Beim Upsamling werden diese separaten Feature-Maps mit den jeweiligen vergrößerten Feature-Map der transponierten Faltungen zusammengebracht. Eine gestapelte (engl: stacked) Hourglass-Architektur setzt mehrere einzelne Hourglass-Module hintereinander ein.

In den meisten Netzwerken werden *stacked hourglass* CNNs für die Kostenaggregation auf den Kostenvolumen verwendet. In ihrem Netzwerk *PSMNet* verwendeten Chang und Chen [2018] drei gestapelte Hourglass-Module, um die endgültige Disparitätskarte als auch zwei Zwischenausgaben nach jeweils einem Modul zu erzeugen. Der Fehler für die Zwischenausgaben und der Endausgabe ergeben als gewichtete Summe den endgültigen Fehler und ermöglichen die direkte Überwachung der einzelnen Module. Für die Feature-Extraction verwendeten die Autoren das *spatial pyramid pooling* Modul, durch welches die Verbindung zwischen Objekten und deren zugehörigen Regionen wie z. B. Fenster oder Reifen eines Autos erlernt werden sollten.
Durch die Verwendung von Average-Pooling wurden vier Feature-Maps mit unterschiedlicher Größe erzeugt, welche nach jeweils einer 1×1 Faltungen zur Reduzierung der Dimensionen konkateniert wurden. Nach weiteren Faltungen wurden diese Feature-Maps dann zu einem Volumen konkateniert, und mit dem Stacked-Hourglass-Netzwerk verarbeitet. Das Netzwerk erzielte zum Zeitpunkt der Veröffentlichung herausragende Ergebnisse auf den KITTI-Benchmark Datensätzen (s. 5.1.1) und wurde in vielen anderen Arbeiten als Vergleich verwendet. Im Vergleich zum *GC-Net*, welches zu der Zeit eines der besten Netzwerke war, lieferte PSMNet bessere Ergebnisse bei zusätzlich geringerer Ausführungszeit [Hamid et al., 2022].
Gegensatz zu den Autoren von PSMNet verwendeten Zhang et al. [2019] für ihr *GA-Net (global aggregation network)* ein Stacked-Hourglass für die Feature-Extraction, in welchem beide Bilder zusammen verarbeitet wurden, um daraus ein 4D Kostenvolumen zu erstellen. Für die Kostenaggregation orientierten sie sich an der Semi-global-Matching-Methode (s. Abschnitt 2.1.2). Sie entwickelten eine *semi-global guided aggregation (SGA)* Schicht mit lernbaren Parametern für die benutzerdefinierten Größen von SGM, wie z. B. den Strafen der Energiefunktion. Die Aggregation wurde dabei in 4 Richtungen über die gesamte Feature-Map durchgeführt und das Maximum ausgewählt, welches die höchste Wahrscheinlichkeit für die richtige Korrespondenz angibt.
Zum Schluss wurde eine *local guided aggregation (LGA)* Schicht eingesetzt, welche zur Verbesserung von feinen Strukturen und Kanten führen soll. Diese wird durch das Er-

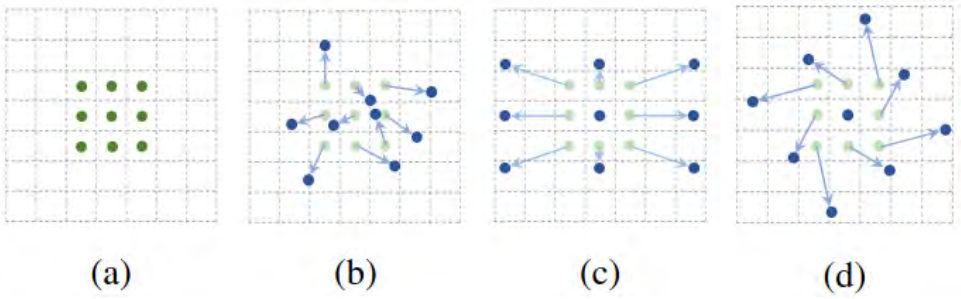


Abbildung 2.14: Darstellung von verformbaren Faltungen mit einem 3×3 Filter. (a) normale Faltung (grüne Punkte). (b) verformte Faltung (blaue Punkte). Generalisierung verschiedener Bildtransformation wie Skalierung (c) und Rotation (d). Abb. aus [Dai et al., 2017, Abb. 1].

lernen mehrere geführter (engl.: guided) Filter umgesetzt, die derselben Kostenfunktion folgen wie auch die SGA-Schicht. Ein *guidance* Subnetzwerk, bestehend aus der Faltung des linke Bildes und dessen Ergebnis aus der Feature-Extraction, stellt die Gewichte für die SGA- und LGA-Schichten bereit. Nach Testen verschiedener Konfigurationen erreichten sie die besten Ergebnisse mit drei SGA-Schichten und insgesamt 15 3D-Faltungen. Im Vergleich zu PSMNet erreichten sie bessere Genauigkeiten in ihren Ergebnissen, aber mit höherer Ausführungszeit.

Einen ganz anderen Ansatz verfolgten Xu und Zhang [2020] mit ihrem *Adaptive Aggregation Network (AANet)*. Sie verwendeten eine *feature-pyramid* für die Feature-Extraction, welche je Seite in drei Feature-Maps unterschiedlicher Größe resultierte. Durch die Korrelation der linken und rechten Feature-Maps für jede Größe wurde ein *multi-scale cost volume* erstellt. Für die Kostenaggregation entwickelten die Autoren ein *adaptive intra-scale aggregation* Modul, welches mit verformbaren (engl: deformable) Faltungen die Ergebnisse von Kanten verbessern sollte.
Bei verformbaren Faltungen [Dai et al., 2017] werden die Sample-Punkte des Filterkerns durch einen Offset versetzt, sodass Informationen aus unterschiedlichen Bereichen der Umgebung zusammengebracht werden. Die Offsets können dabei für alle Punkte gleich oder völlig unterschiedlich sein, wie in Abbildung 2.14 dargestellt ist. Pixelwerte werden durch bilineare Interpolation bestimmt, da die Punkte durch die Offsets nicht an dem Pixel-Raster ausgerichtet sind.

Für die Verbesserung der Disparitäten von Regionen mit wenig Details wurde das *cross-scale aggregation* Modul eingesetzt, welches Korrespondenzen über die unterschiedlichen Größen des Multi-Scale-Kostenvolumens bestimmt und diese adaptive zusammenfügt.

Das Netz erzielte im Vergleich mit Netzen wie PSMNet und GA-Net ähnliche Genauigkeiten, benötigte dafür aber bedeutend weniger Ausführungszeit.

Die Entwicklung von Netzwerkarchitekturen für Stereo-Matching ist geprägt von dem Zusammenführen und Verbessern vorheriger Ansätze, um den aktuellen Stand der Technik voranzutreiben. Auch die Autoren Xu et al. [2022] entwickelten auf diese Weise ihre *ACVNet*-Architektur, welche in diesem Projekt verwendet wurde. Die Struktur und Besonderheiten der Architektur als auch die Gründe für die Auswahl werden in Kapitel 4 beschrieben.

3 Stereokorrespondenz in suboptimalen Bedingungen

In diesem Kapitel wird der Einfluss von umweltbedingten Störeffekten auf Verfahren der Tiefenbestimmung erläutert. Es werden existierende Deep-Learning Ansätze für die betrachteten Störeffekte vorgestellt und ihre Eignung für diese Arbeit besprochen.

3.1 Einfluss von Störeffekten auf Verfahren zur Tiefenbestimmung

Schlechte Wetterbedingungen stellen besonders im Kontext von Fahrassistenzsystem und autonomen Fahren eine große Herausforderung dar [Zang et al., 2019]. Regen, Schnee und Hagel erschweren die Sichtverhältnisse und die Handhabung eines Fahrzeuges für den Fahrer, wodurch dieser noch stärker auf diese Systeme angewiesen ist. Statistiken zeigen, dass die Unfallrate bei Regen 70 % höher ist als unter normalen Bedingungen und dass verschneite und vereiste Straßen in zum Beispiel den USA zu jährlich 30.000 Unfällen führen [Zhang et al., 2023]. Die Problematik ist, dass die Sensoren wie LiDAR, Kameras, GPS und Radar, welche diese Systeme verwenden, durch diese Bedingungen ebenfalls gestört sind.
LiDAR-Sensoren sind wichtig für die Bestimmung von Distanzen. LiDAR-Sensoren (**Li**ght **D**etection **A**nd **R**anging) sind Abstandssensoren, welche nach dem *Time-of-Flight*-Prinzip Distanzen in der Umgebung messen [Gotzig und Geduld, 2015]. Dafür wird die Zeitdauer zwischen dem Aussenden eines Lichtimpulses und dem Empfangen der Reflexionen gemessen und mit dem Wissen über die Geschwindigkeit von Licht die Distanz zur Stelle der Reflexion bestimmt. 2D-LiDAR messen die Distanzen entlang einer horizontalen Linie, wogegen 3D-LiDAR, durch unterschiedliche Winkel beim Aussenden auch Distanzen auf unterschiedlichen Höhen messen. Regentropfen, Schneeflecken oder Aerosole wie Nebel, Rauch oder Staub können diese Lichtimpulse beeinflussen und zu reduzierter Reichweite

oder verfälschten Messungen führen. Tests zeigen, dass leichter Regen wenig Einfluss auf die Messungen eines LiDARs hat, doch bei einer Stärke von 30 mm/h die Reichweite um mehr als 50 % reduziert werden kann [Yoneda et al., 2019]. Zudem können Pfützen zusätzlichen Fehlmessungen führen, wodurch sie schwer von wirklichen Hindernissen unterscheidbar sind.

Auch Kameras sind von diesen Bedingungen in der Wahrnehmungsfähigkeit beeinträchtigt. In regnerischen Bedingungen können anhaftende Regentropfen auf der Linse oder der Scheibe davor das Sichtfeld blockieren oder die eigentlichen Informationen verzerren [Zhang et al., 2023]. In kälteren Gebieten oder Jahreszeiten können ebenso Schneeflocken und gefrorene oder beschlagene Scheiben die Sicht verschlechtern oder ganz versperren. Nebel dagegen führt zu einer Reduzierung der Sichtweite, welche abhängig der Dichte das Erkennen der Umgebung erschwert. Zudem führt die weiß-gräuliche Färbung der in Nebel gehüllten Umgebung zu Kontrastverlust in den resultierenden Bildern. Abseits der Straße führt starker oder lang anhaltender Niederschlag zu schlammigem Boden, was zu Verdeckung durch Dreck führen kann. Auch Schneematsch kann dieses Problem verursachen.
Diese Effekte beeinflussen bei Nutzung von Stereokameras auch die Qualität der erstellten Disparitätskarten. In Rahmen dieser Arbeit soll untersucht werden, ob neuronale Netzwerke, die für diese Zwecke trainiert wurden, eine verlässliche Methode sind, um unter diesen Bedingungen Disparitätskarten erstellen zu können. Es wird sich dabei speziell auf **Nebel**, **anhaftende Regentropfen** und allgemeine **Verdeckung** durch Objekte fokussiert. Bei der Verdeckung werden Teilverdeckungen durch beispielsweise Blätter und großflächiger Verdeckungen wie durch Eis, Dreck oder eine beschlagene Scheibe betrachtet.

3.2 Deep Learning Ansätze

Die grundsätzlich simpelste Lösung für verbesserte Disparitätsschätzung mit verunreinigten Bildern ist die Entfernung der Störeffekte im Voraus. Methoden zur Entfernung von Regen (deraining) oder Nebel (dehazing) existieren über den Kontext von Stereo-Vision hinaus besonders für Einzelbilder. Diese Methoden könnten verwendet werden, um die Bilder zu bereinigen, mit denen „normale“ Stereo-Matching-Methoden die Disparitätsschätzung durchführen können. Der Vorteil dieses Ansatzes ist die Flexibilität, welche der Umfang von Kombinationsmöglichkeiten beider Methoden bietet. Jedoch wird in den

folgenden betrachteten *simultanen* Ansätzen gezeigt, dass das Korrespondenzproblem und die Rekonstruktion verdeckter Informationen sehr ähnlich sind. Für diese Ansätze werden End-to-End Netzwerke verwendet, welche beide Teilaufgaben erledigen. Die Ausgabe kann dabei nur die Disparitätskarte oder auch die bereinigten Bilder umfassen. Aus diesem Grund werden *sequenzielle* Lösungen nicht betrachtet.

3.2.1 Stereokorrespondenz in Nebel

Das Entfernen von Nebel aus Bildern ist eine schwierige Aufgabe, da die für die Bereinigung nötigen „Unbekannten“ nicht einfach zu finden sind [Song et al., 2020]. Der Effekt von Nebel oder ähnlichen Bedingungen kann durch das mathematische Modell (Gleichung 3.1, aus [Song et al., 2020, Gl. 1]) für die *atmosphärische Streuung* beschrieben werden, welches die Streuung von Licht durch in der Luft befindliche Aerosole beschreibt.

$$\mathbf{I}(x) = \mathbf{J}(x)\mathbf{T}(x) + A(1 - \mathbf{I}(x)) \tag{3.1}$$

Die *transmission map* $\mathbf{T}$ beschreibt die Lichtdurchlässigkeit der Szene und der Faktor A das globale atmosphärische Licht (engl.: airlight), welches die Färbung des Nebels bestimmt. Die Transmission-Map ist abhängig von der Tiefe der Szene Z und kann bei homogenem Nebel mit der Gleichung 3.2 (aus [Song et al., 2020, Gl. 2]) beschrieben werden.

$$\mathbf{T}(x) = e^{\beta \mathbf{Z}(x)\mathbf{N}(x)} \tag{3.2}$$

Umgekehrt kann das Bild ohne Nebel aus der inversen Rechnung (Gl. 3.3) erhalten werden, wobei ϵ eine kleine Konstante für die numerische Stabilität ist.

$$\mathbf{J}(x) = \frac{\mathbf{I}(x) - A(1 - \mathbf{T}(x))}{max(\epsilon, \mathbf{T}(x))} \tag{3.3}$$

Die Beziehung zwischen der Tiefe und der Lichtundurchlässigkeit kann sowohl für Stereo-Matching als auch für die Entfernung von Nebel ausgenutzt werden. Dichterer Nebel lässt auf eine größere Tiefe schließen. Die Transmission-Map gibt Hinweise über entferntere Objekte und die Tiefe aus der Disparität ist für nähere Objekte verlässlicher.

Aus diesem Grund verwendeten Song et al. [2020] in ihrer Arbeit genau dieses Modell, um ein Netzwerk (SSMD) für die gleichzeitige Entfernung von Nebel und Disparitätsschätzung zu entwickeln. Das Netzwerk erlernte die Parameter für die Transmission-Map, die Informationen über die Tiefe in Bezug auf die Dichte des Nebels gibt. Dies wurde genutzt,

um damit eine verbesserte Disparitätskarte zu erstellen. Gleichzeitig lernte es auch das atmosphärische Licht der Szene zu erkennen, um mit der Transmission-Map nach dem inversen Modell ein bereinigtes Bild zu erzeugen. Das Netz kann sowohl zur Erstellung von Disparitätskarten als auch für die Bereinigung von Nebel in Stereobildern verwendet werden. Zudem ist es möglich mit einer Disparitätskarte eines Lehrer-Netzwerkes ein eigenes Nebelbild zu erzeugen, welches ein Schüler-Netzwerk als Eingabe erhält und dadurch ein selbst-überwachtes Training durchzuführen. Das Netzwerk war in der Lage auf synthetischen und realen (meistens aber mit künstlichem Nebel) Datensätzen Nebel zu entfernen und gute Disparitätskarten zu erzeugen.

Dasselbe mathematische Modell verwendeten auch Yao und Yu [2022] in ihrer Arbeit. Sie entwickelten Netzwerk, welches anhand des Modells für atmosphärische Streuung ein *fog volume* für verschiedene Tiefen erstellt. Das Netzwerk erlernte die Parameter des Modells anhand der nebeligen Bilder, um mehrere bereinigte Bilder für unterschiedliche Tiefen zu erzeugen. Aus den einzelnen bereinigten Bildern wurde das *fog volume* erstellt (s. Abbildung 3.1). Dies wurde mit dem normalen Kostenvolumen zusammengebracht, wobei das *fog volume* Hinweise zur korrekten Auswahl von Disparitäten liefert. Im Vergleich zum SSMD-Netzwerk erreichte es auf realistischen Daten bessere Ergebnisse. Für die Erzeugung der bereinigten Bilder für das *fog volume* ist jedoch die Brennweite und die Basis notwendig, wodurch sich das Netzwerk nicht komplett unabhängig einsetzten lässt.

Beide vorgestellten Arbeiten nutzen das Modell der atmosphärischen Streuung und bauen auf dem Zusammenhang zwischen Tiefe und Nebeldichte auf. Aufgrund von fehlendem Zugang zum Quellcode und zeitlichen Einschränkungen werden diese entwickelten Netze in dieser Arbeit nicht verwendet.

3.2.2 Stereokorrespondenz mit Regentropfen

Für das Entfernen von *haftende* (engl.: adherent) Regentropfen und „Regenstreifen/-schlieren“(engl.: *rain streaks*) in Bildern wurden in den letzten Jahren viele Methoden vorgestellt [Zhang et al., 2021]. Der Fokus liegt jedoch hauptsächlich nur auf der Bereinigung von Einzel- oder Stereobildern. Direkte Ansätze zur Disparitätsschätzung auf Stereobildern mit dem Ziel eine „fehlerfreie“ Disparitätskarte zu erhalten, werden nicht untersucht.

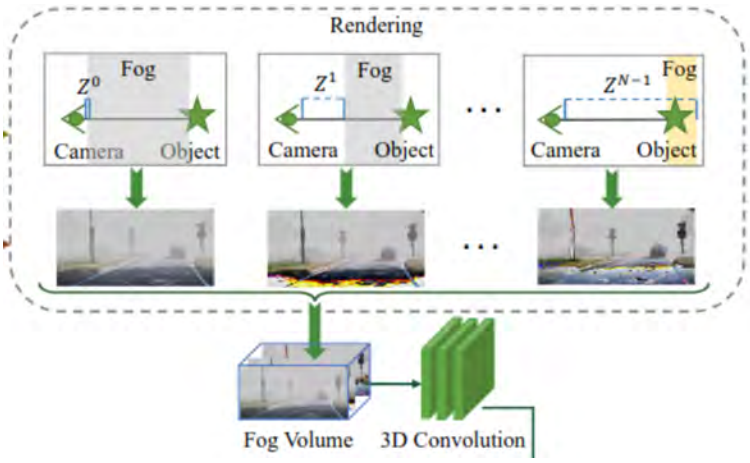


Abbildung 3.1: Erstellung des *fog volume* aus mehreren „ent-nebelten“ Bilder als Teil der *Foggy Stereo* Architektur. Ausschnitt aus [Yao und Yu, 2022, Abb. 2].

Das Problem bei der Bereinigung von Stereobildern ist grundsätzlich dasselbe wie bei der Erstellung einer Disparitätskarte: das Finden von Korrespondenzen. Die Bereiche einer Szene, die in einem Bild verdeckt sind, können gegebenenfalls aus dem anderen Bild extrahiert werden. Dafür müssen die korrespondierenden Pixel gefunden werden.

In ihrer Arbeit verwenden Shi et al. [2021] diesen Ansatz, um Wassertropfen aus Stereobildern zu entfernen. Da der Hintergrund durch die Wassertropfen verdeckt bzw. verformt ist, ist es wichtig, ein großes rezeptives Feld zu nutzen. Die Position der verdeckten Pixel muss aus dem Kontext der umliegenden, nicht verdeckten Pixel bestimmt werden können. Sie nutzen verformte Faltungen auf Bildreihen, um Ausschnitte aus den Bildern zu vergleichen und die Ähnlichkeit zu bestimmen. Dies ist sehr ähnlich zu lokalen Stereokorrespondenz-Methoden, die ein Fenster für die Kostenaggregation verwenden.
Auf ähnliche Weise nutzen Yan et al. [2020] in ihrer Arbeit Disparität zur Restaurierung von Stereobildern. Mit einer erstellen Disparitätskarte und dem linken Bild wurde eine Rekonstruktion des rechten Bildes erzeugt, um gemeinsame Bildinformationen zu finden. Dafür integrieren sie die Architektur eines Stereo-Matching-Netzwerkes. Ihre Methode wird für die Rekonstruktion von Bildern mit Rauschen und Unschärfe eingesetzt.

Diese vorgestellten Arbeiten zeigen, dass Disparität als Grundlage zum Wiederherstellen verdeckter Bildinformationen einsetzbar ist. Das Finden der Korrespondenzen ist dabei nur ein Teilschritt und dient zur Erzeugung von bereinigten Eingabebildern. Es lässt sich vermuten, dass ein Netzwerk, das auf das Erkennen von Disparitäten ausgelegt ist,

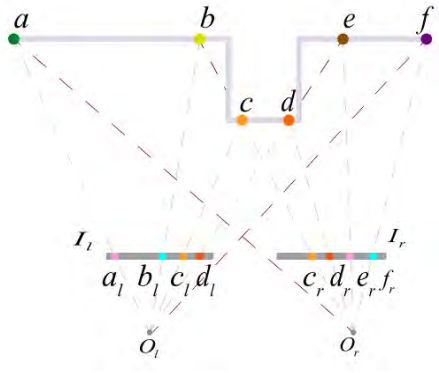


Abbildung 3.2: Darstellung der Verdeckung in einer Szene. Für Kamera O_l liegt Punkt f außerhalb des Sichtbereiches und e ist verdeckt. Für Kamera O_r liegt Punkt a außerhalb des Sichtbereiches und b ist verdeckt. Abb. aus [Li et al., 2022, Abb. 3].

demnach auch in der Lage sein sollte, Disparitäten anhand von Bilder mit Regentropfen erkennen zu können.

3.2.3 Stereokorrespondenz bei Verdeckung

Im Kontext von Stereo-Matching bezieht sich Verdeckung (engl.: occlusion) auf Bereiche der Szene, die nur aus der Sicht einer Kamera zu sehen sind und aus Sicht der anderen verdeckt sind. Dazu zählen vor allem die Bereiche des linken Bildrandes im linken Bild und des rechten Bildrandes im rechten Bild. Objekte in der Szene können die Sicht auf andere Objekte im Hintergrund für die Sicht einer Kamera verdecken. Abhängig vom Abstand der Kameras ist der Bereich, der nur im Sichtfeld von einer Kamera liegt, unterschiedlich groß. Der verdeckte Bereich in einem Bild kann im anderen aber sichtbar sein. Eine simple Darstellung ist in Abbildung 3.2 zu sehen. Die Punkte e und f sind für die linke Kamera O_l nicht sichtbar. Dagegen sind für die rechte Kamera O_r die Punkte e und f nicht sichtbar.

Für die Verbesserung der Qualität dieser Bereiche in Disparitätskarten mit Deep-Learning-Methoden existieren vereinzelte Arbeiten, welche meist unter dem Schlüsselwort *occlusion-aware* zu finden sind. Ein Beispiel ist die Arbeit der Autoren Li et al. [2022] die versuchen, dieses Problems durch explizite Erkennung der Verdeckungen zu lösen. Im Gegensatz zu anderen Arbeiten, die nur zwischen verdeckten und nicht verdeckten Pixeln unterscheiden, wurde eine ternäre Klassifikation angewendet. Ein Pixel kann entweder durch ein

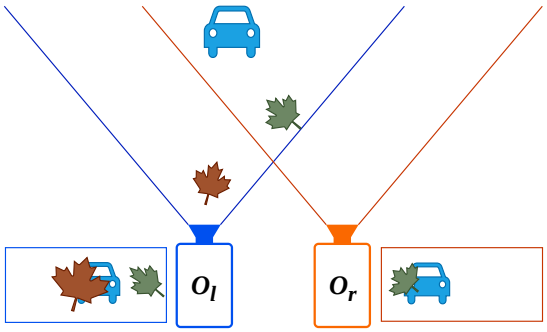


Abbildung 3.3: Skizze einer exemplarischen Szene, die zwei Arten von Verdeckung zeigt. In blau ist der Sichtbereich der linken Kamera O_l und in orange der Bereich der rechten Kamera O_r.

benachbartes Objekt durch den Bildrand oder gar nicht verdeckt sein. Diese Unterscheidung wird bei der Berechnung der Disparitäten berücksichtigt und am zum Schluss für eine gerichtete Glättung der Disparitäten verwendet. So werden für die linke Kamera verdeckte Bereiche von rechts geglättet und die für die rechte Kamera verdeckten Bereiche von links.

Die damit behandelten Verdeckungen sind grundsätzlich durch die Verschiebung des Sichtbereiches der linken und rechten Kamera verursacht. In den meisten Fällen ist diese nicht größer als 50 cm, wie es beispielsweise bei den KITTI-Datensätzen der Fall ist. Die resultierenden Verdeckungen sind nicht so große, wie es durch eine Objekt-Verdeckung nahe vor der Kamera wäre.

In Abbildung 3.3 ist eine Szene dargestellt, in der zwei Blätter die Sicht auf ein Auto verdecken. Dabei ist das braune Blatt näher an der linken Kamera O_l und damit nur im Sichtbereich der linken Kamera. Ohne das grüne Blatt wäre im linken Bild die Sicht auf das Auto vollständig frei.

Im Falle des grünen Blattes ist dies im Sichtfeld beider Kameras. Für ein Stereo-Matching-Algorithmus besteht hierbei die Schwierigkeit, dass Blatt als störenden Element zu erkennen und zu ignorieren. Als Hinweis könnte die extrem große Disparität dienen, die durch die geringe Distanz besteht. Mit einem ausreichend großen Kontext könnte der verdeckte Teil des Autos aus dem linken Bild geholt werden, der dort (ohne das braune Blatt) nicht verdeckt ist. Für diese Art von Verdeckungen, die durch störende Objekte in der Szene verursacht werden, existieren keine Arbeiten, in denen eine Architektur für diesen Zweck entwickelt wurde. Zudem wurden auch keine Untersuchungen dazu angestellt, wie die Ergebnisse herkömmlichen Netzwerkarchitekturen in diesen Bedingungen sind.

Unter der Betrachtung der erläuterten Zusammenhänge zwischen der Bestimmung von Disparitäten und der Wiederherstellung verdeckter Daten in Stereobildern wurde eine Netzwerkarchitektur ausgewählt. Es wurde ein allgemeines, nicht speziell für die genannten Bedingungen entwickeltes Stereo-Matching-Netzwerk als Grundlage gewählt. Auch die Verfügbarkeit von Arbeiten für diese speziellen Fälle, auf denen man hätte aufbauen können, spielte bei der Entscheidung eine Rolle. Zudem ermöglicht es die Evaluierung der Eignung eines solchen Netzes für diese herausfordernden Bedingungen, die sonst nur für optimale Bedingungen getestet wurden.

Im folgenden Kapitel wird die Architektur des gewählten Netzes beschrieben.

4 Netzwerkarchitektur

Die Auswahl von neuronalen Netzwerken für Stereo-Matching ist groß. In den Evaluationsranglisten von Datensätzen wie KITTI 2015[1] oder Middleburry[2] lassen sich viele Netzwerke mit unterschiedlichen Ansätzen oder erweiterten Versionen von anderen einsehen. Diese Ranglisten erleichterten das Finden von wissenschaftlichen Arbeiten und dienten als Orientierung für die Leistung der Netzwerke. In den meistens Arbeiten werden dabei Datensätze wie KITTI 12/15, Middleburry, Sceneflow und ETH3D[3] verwendet. Dies ermöglicht den direkten Vergleich mit anderen Arbeiten, welche mit denselben Daten gearbeitet haben. Allerdings umfassen diese Datensätze nur Aufnahmen bei gutem Wetter oder nur von Objekten innerhalb eines Gebäudes. Der Fokus liegt daher auf der Leistung bei optimalen Bedingungen und mit optimalen Daten. Die Eignung für den Einsatz unter schwierigeren Bedingungen wird nie betrachtet, weshalb sie nicht als Kriterium für die Auswahl eines Netzwerkes für diese Arbeit verwendet werden konnte.

Im Rahmen dieser Arbeit wurden drei Netzwerke getestet, von den das letztendlich für die endgültigen Ergebnisse verwendete Netz ausgewählt wurde. Als erstes wurde *Pyramid Stereo Matching Network* (PSMNet) von Xu und Zhang [2020] verwendet, das bereits in Abschnitt 2.3 kurz beschrieben wurde. Aufgrund des besonderen Ansatzes für die Feature-Extraction wurden gute Ergebnisse erwartet. Zudem wurde es in der Literatur häufig als Referenz verwendet.
Das zweite betrachtete Netz war *Adaptive Aggregation Network* (AANet) von Xu und Zhang [2020], welches eine geringe Ausführungszeit und verbesserte Qualität für Kanten bot.
Als letztes und finales Netzwerke wurde das *Attention Concatenation Volume Network* (ACVNet) von Xu et al. [2022] getestet. Zum Zeitpunkt der Auswahl war das Netzwerk eines der zehn besten Netzwerke in den Ranglisten der KITTI-Stereo-Evaluierung. Von

[1] https://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo
[2] https://vision.middlebury.edu/stereo/eval3/
[3] https://www.eth3d.net/overview

diesen zehn war es die einzige Arbeit, wofür der Quellcode zugänglich war. Es kombinierte Ansätze der anderen Netze und bot sowohl eine geringe Ausführungszeit als auch beeindruckende Qualität. Auch der Umfang des Projektes für den Trainingsprozess, Protokollierung und Auswertung stellten eine gute Grundlage für die Nutzung dar.

Im Folgenden werden die Architektur und die Leistung des verwendeten ACVNet beschrieben.

4.1 ACVNet

Das Ziel der Autoren Xu et al. [2022] für die Entwicklung dieser Architektur war eine effizientere Form des *cost volumes* zu finden, um die Kostenaggregation effizienter zu machen. Dabei basiert das Model auf zwei Beobachtungen aus der Analyse der verschiedenen Varianten von Kostenvolumen. Die erste Beobachtung war, dass ein Konkatenationsvolumen sehr viele, aber redundante Informationen enthält. Die zweite Beobachtung war, dass ein Korrelationsvolumen die Ähnlichkeit der beiden Bildern misst und dadurch implizit Beziehungen zwischen benachbarten Pixeln widerspiegelt. Aus diesen Beobachtungen entstand die Idee, ein Korrelationsvolumen zu verwenden, um ein Konkatenationsvolumen zu erstellen, bei welchem redundante Informationen unterdrückt und gleichzeitig wichtige Informationen behält. Darauf basierend wurde die Architektur entwickelt, bei der ein Korrelationsvolumen verwendet wird, um sogenannte *Attention-Weights* zu generieren, welche zur Filterung des Konkatenationsvolumens genutzt werden.

Die folgenden Abschnitte erläutern die Eigenschaften der Architektur.

Extraktion der Feature

Die Eingabe des Netzwerks besteht aus zwei RGB Bildern, welche parallel verarbeitet werden. Beide Stränge sind identisch aufgebaut und teilen sich die Gewichte. Zum Extrahieren der bedeutsamen Bildinformationen wird ein Aufbau verwendet, der einer dreischichtigen Architektur von einem Residual Neural Network (ResNet) ähnelt (s. Abschnitt 2.2.2 Residual Learning).
Mit drei 2D-Faltungen findet zuerst ein Downsampling der Bilder statt. Darauf folgt ein ResNet-Block mit 16 Schichten, welcher die Ausgabe $l1$ von unären Features bei einem

viertel der Eingabe-Auflösung erzeugt. Die unären Features sind in diesem Fall die Intensität der einzelnen Pixel. Zwei weitere ResNet-Blöcke mit jeweils drei Schichten ergeben die Ausgaben $l2$ und $l3$. Diese drei Feature-Maps werden konkateniert und bilden eine Feature-Map mit 320 Kanälen. Zwei weitere Faltungen komprimieren die Feature-Map 32 Kanäle und ergeben die endgültigen Feature-Maps f_l und f_r.

Generierung der Attention-Weight

Die Attention-Weights (deut.: Aufmerksamkeitsgewichte) sollen dem Netz ermöglichen, für jede Eingabe die wichtigsten Informationen zu finden und diese für die Kostenaggregation hervorzuheben. Durch Unterdrücken der redundanten Informationen wird der Schritt zudem effizienter, da die Filterung nicht durch die 3D-Faltungen auf dem Kostenvolumen stattfinden muss [Xu et al., 2022].
Als erster Schritt findet die Generierung des initialen Korrelationsvolumens statt. Die herkömmliche Herangehensweise ist, die Korrelation Pixel für Pixel durchzuführen. Dieser Weg führt zu unzuverlässigen Ergebnissen bei Oberflächen mit wenig Struktur. Die hierbei verwendete Lösung ist die Anwendung vom *multi-level adaptive patch matching (MAPM)*, welches durch zwei Kernaspekten charakterisiert wird.

Zum Ersten wir bei dieser Methode die Korrelation mit *Patches*(deut.: „Stellen“) Ausschnitten unterschiedlicher Größen durchgeführt, um Merkmale mit unterschiedlichem Detailgrad zu berücksichtigen. Hierfür werden geweitete Faltungen (s. Abschnitt 2.2) angewendet. Diese vergrößern das rezeptive Feld und ermöglicht das Einbeziehen eines größeren Kontextes für die Berechnung der Matching-Kosten. Die Ähnlichkeit zweier Pixel ergibt sich aus der gewichteten Summe der Pixel, welche vom Kernel einbezogen werden. Diese stellt die Matching-Kosten dar.

Zusätzlich zu der geweiteten Faltung wird die Idee der *group-wise Correlation*(deut.: gruppenweisen Korrelation) des GwCNet von Guo et al. [2019a] übernommen. Dabei werden die 320-Kanal Feature-Maps bestehend aus den einzelnen Feature-Maps l_1,l_2 und l_3 aus der Feature-Extraction in 40 Gruppen aufgeteilt. Letztendlich sind von diesen 40 Gruppen die ersten 8 aus l_1, die folgenden 16 aus l_2 und die letzten 16 aus l_3.
Für die drei Ebenen (die drei Teilgruppen) wird die geweitete Faltung mit unterschiedlichen Raten durchgeführt: $Rate_{l_1} = 1$,$Rate_{l_2} = 2$ und $Rate_{l_3} = 3$. Die daraus resultierenden Feature-Maps werden konkateniert und bilden das *multi-level patch matching volume* der Größe $40 \times D/4 \times H/4 \times W/4$, mit D der maximalen Disparität, H und W

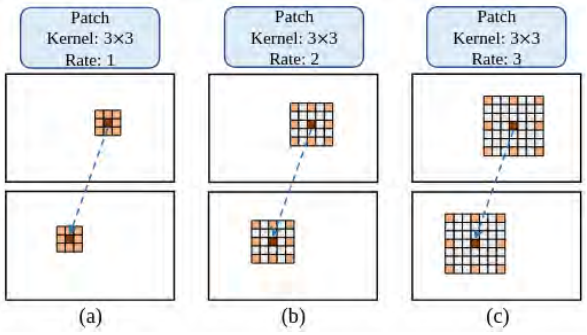


Abbildung 4.1: Visualisierung von MAPM mit der Anwendung von geweiteten Faltungen für die drei Ebenen l_1(a), l_2(b) und l_3(c). Das zentrale Pixel ist rot und die vom Kernel einbezogenen Pixel sind orange. Abbildung aus [Xu et al., 2022, Abbildung 3].

der Breite und Höhe des Eingabebildes. Abbildung 4.1 zeigt die drei geweitete Faltungen für die jeweiligen Ebenen. Die oberen Bilder sind die jeweiligen Feature-Maps des linken Bildes und die unteren die Feature-Maps des rechten Bildes.

Das erstellte Volumen wird darauf durch zwei 3D-Faltungen und ein 3D-*Hourglass* Netzwerk gegeben und letztendlich mit einer weiteren Faltung auf eine einzige Dimension komprimiert, um die endgültigen Attention-Weights zu erhalten.
Beim Training werden diese separat trainiert, indem eine eigene Schätzung gemacht wird, welche mit der eingegeben Disparitätskarte verglichen wird.

Attention Concatenation Volume (ACV)

Das namensgebende *attention concatenation volume* wird aus der initialen Konkatenation der linken und rechten Feature-Map f_l und f_r für jede Disparität gebildet. Bei einer jeweiligen Größe der Eingabebilder von $H \times W \times 3$ und der maximalen Disparität D ergibt sich ein Volumen der Größe $2 * 32 \times D/4 \times H/4 \times W/4$. Die separat generierten Attention-Weights werden nun zur Filterung des initialen Konkatenationsvolumens eingesetzt. Das ACV ergibt sich aus der elementweise Multiplikation des Konkatenationsvolumens $\mathbf{C}_{concat}$ mit den Attention-Weights $\mathbf{A}$ pro Kanal i:

$$C_{ACV} = \mathbf{A} \odot \mathbf{C}_{concat}(i) \tag{4.1}$$

Aus diesem Volumen müssen die endgültigen Disparitäten bestimmt wird, was durch ein Subnetzwerk für die Kostenaggregation umgesetzt wird.

Kostenaggregation

Für die Aggregation der Matching-Kosten verwenden die Autoren wie viele andere *stacked 3D-Hourglass* Netzwerke (s. 2.2). Zu Beginn werden zwei normale 3D-Faltungen durchgeführt, auf welche zwei gestapelte 3D-Hourglass-Netze folgen. Beide 3D-Hourglass Netzwerke bestehen aus vier 3D-Faltungen und zwei transponierten Faltungen durch 3D-Transpose Schichten. Hierbei werden insgesamt drei Ausgaben generiert: $output_1$ nach den ersten zwei Faltungen, $output_2$ nach dem ersten Hourglass-Netzwerk und $output_3$ nach dem zweiten Hourglass-Netzwerk. Jede der Ausgaben wird auf ein 4D Volumen mit einem Kanal komprimiert und mit der *softmax*-Funktion in ein Wahrscheinlichkeitsvolumen für jede Disparität konvertiert. Die endgültige Disparität wird mit der *soft-argmin*-Funktion (Gleichung 4.2, aus [Xu et al., 2022, Gl. 5])

$$d = \sum_{k=0}^{D_{max}-1} k \cdot p_k \tag{4.2}$$

berechnet, mit der Disparität k und der Wahrscheinlichkeit der Disparität an diesem Pixel p_k. Die letzte der drei Disparitätskarten ist die finale Schätzung des Netzes für die Eingabebilder. Die zwei vorherigen Ausgaben als auch die Schätzung der Attention-Weights werden zur Berechnung des Fehlers verwendet und werden dabei unterschiedlich gewichtet.

Die Autoren haben zusätzlich zu der beschriebenen Architektur noch eine schnellere Variante entwickelt, die sie *ACVNet-Fast* nannten. Sie besitzt dieselbe Struktur wie die normale Variante, verwendet aber weniger Faltungen in der Feature-Extraction und der Kostenaggregation. Das Multi-Level-Adaptive-Patch-Matching wird zudem bei 1/8 der Auflösung durchgeführt, um den Vorgang zu beschleunigen. Dieses Netz kann für den Einsatz in Echtzeit genutzt werden, findet in dieser Arbeit aber keine Anwendung.

Leistung des Netzwerks

Für die Evaluierung des Netzwerkes wurde zuerst ein Grundtraining mehreren Phasen mit dem Sceneflow-Datensatz durchgeführt. Als erstes wurden die Gewichte der *Attention Weight Generation* und dann die des restlichen Netzwerkes für jeweils 64 Epochen trainiert. Danach wurde das gesamte Netz für weitere 64 Epochen trainiert. Dies stellt

das trainierte „Grundnetzwerk“ dar, welches auch von den Autoren zur Verfügung gestellt wird. Für die Evaluierung auf den KITTI-Datensätzen wurde das Grundnetzwerk 500 Epochen auf den gemischten KITTI 12/15 Daten trainiert. Die jeweiligen finalen Versionen ergaben sich aus weiteren 500 Epochen auf den einzelnen Datensätzen.
Verglichen mit den zu dem Zeitpunkt besten Netzwerken erreichte ACVNet auf dem Sceneflow-Datensatz eine Verbesserung des End-Point Fehlers (EPE) von 34%. Auf den KITTI-Datensätzen schlug das Netz die Referenz-Netze im Hinblick auf den D1- und EPE-Fehler fast in jeder Kategorie und war zu dem Zeitpunkt das zweitbeste Netzwerk in der Rangliste.

Mit dem beschriebenen Netzwerk als Grundlage werden verschiedene Trainingsläufe auf unterschiedlichen Daten mit Regentropfen, Nebel oder Verdeckungen durchgeführt. Die dafür verwendeten Datensätze und angewandte Methoden zur Augmentation werden im nächsten Kapitel beschrieben.

5 Datensätze

Gute Datensätze sind wichtig für das Training von neuronalen Netzwerken. Das Training auf einem Datensatz, welcher nicht genug Abwechslung in den Daten bietet, resultiert in einer meist schlechten Generalisierungsfähigkeit.
Die meisten Datensätze für das Trainieren oder Evaluieren von Stereokorrespondenz-Algorithmen sind anwendungsspezifisch und bieten meist Aufnahmen von Verkehrszenarien oder aus Gebäuden. Das wichtigste Kriterium ist jedoch der Realismus der in den Bildern darstellten Szenen. Einige Datensätze enthalten Aufnahmen, welche mit unterschiedlichen Stereokamerasystemen in der realen Welt durchgeführt wurden.
Andere Datensätze bestehen aus synthetisch erzeugten Bildern, die durch die Verwendung von Simulation-Software oder Game-Engines generiert wurden. Der Vorteil von künstlichen Datensätzen sind die vollständigen Groundtruth-Daten, welche für das Training mit überwachtem Lernen äußerst geeignet sind. Der Nachteil ist die oftmals schlechte Generalisierungsfähigkeit mit realen Daten und die Notwendigkeit für *domain-adaption* [Zhang et al., 2020]. Dies liegt an den Unterschieden der *Domänen* in Bezug auf Beleuchtung, Farbe, Kontrast und Texturen in den Bildern. Es ist notwendig, ein zusätzliches Training zum *fine-tuning* auf den Daten der Domäne durchzuführen, damit das Netzwerk das Gelernte auf die neue Domäne übertragen kann. Dafür werden jedoch Groundtruth-Daten der Domäne benötigt, welche nicht immer verfügbar sind. Daher sind vollständig synthetische Datensätze in vielen Fällen nicht ohne Verbesserung der Architektur direkt nutzbar. In dieser Arbeit werden synthetischen Datensätze deshalb nicht für die letztendliche Evaluierung der Netzwerke verwendet.

Im Folgenden werden weitere Eigenschaften der Datensätze beschrieben.

Damit ein Datensatz für das Training bei überwachtem Lernen und die Evaluierung von Netzwerken verwendet werden kann, muss er gewisse Anforderungen erfüllen:

1. Stereobilder vorhanden
2. Groundtruth vorhanden

3. Extrinsische und intrinsische Parameter

Die grundlegendste Voraussetzung ist das Beinhalten von Stereobildern. Diese können sowohl RGB-Bilder als auch simple Schwarz-Weiß-Bilder sein. Die Bilder müssen bereits rektifiziert sein oder durch Bereitstellung der nötigen intrinsischen und extrinsischen Kameraparameter selbst rektifiziert werden können. Unter der Voraussetzung, dass extrinsische Parameter und Synchronisationsinformationen wie zum Beispiel Zeitstempel und Frame-Nummer vorhanden sind, können auch Bildsequenzen einer einzelnen Kamera verwendet werden. Anhand der mitgelieferten Informationen können dann Bildpaare in gleichmäßigen Abständen extrahiert werden. Eine Rektifizierung muss zudem auch noch stattfinden.

Die zweite wichtige Anforderung ist das Bereitstellen von Groundtruth. Dies kann in Form von Disparitätskarten, Tiefenkarten oder LiDAR-Daten passieren. Da die Ausgabe der Netzwerke Disparitätskarten sind und diese auch für die Überwachung beim Training genutzt werden, ist diese Form die simpelste. Bietet ein Datensatz nur Tiefenkarten, werden die Werte der Brennweite und des Abstandes der Kameras benötigt, um die zugehörigen Disparitätskarten zu berechnen.
Bei der Bereitstellung von Groundtruth in Form von LiDAR-Daten als Punktwolke werden zusätzlich zur Brennweite und dem Abstand der Kameras, die gesamten extrinsischen Parameter benötigt. Um die LiDAR-Daten als Groundtruth für Bilder nutzen zu können, müssen zuerst die Punkte der Punktwolke, welche im Sichtbereich der Kamera liegen als Disparitätskarte extrahiert werden. Dabei werden die extrinsischen Parameter verwendet, um die Punkte so zu transformieren, dass diese auf das linke Bild projiziert werden können und als eigenes Bild gespeichert werden kann. Da die LiDAR-Daten auch Tiefendaten sind, müssen diese auch noch zu Disparitäten konvertiert werden.

Unter diesen Voraussetzungen wurden viele Datensätze untersucht. Es stellte sich schnell heraus, dass für die betrachteten Störeffekt wenig bis gar keine Datensätze existieren, die diese Anforderungen ausreichend erfüllen. Besonders für Datensätze mit Nebel, von denen es grundsätzlich sehr wenige gibt, sind nicht alle für das überwachte Training geeignet. Dies liegt zumeist an der schlechten Qualität der Groundtruth-Daten, welche mit LiDAR-Sensoren aufgenommen wurden und demnach keine verlässliche Grundlage darstellten.
Stereo-Datensätze mit Aufnahmen, bei denen die Kamera durch unterschiedliche Objekte verdeckt ist, wurden keine gefunden.

Daher wurden verschiedene Methoden verwendet, um diese Daten künstlich zu erzeugen. Als Grundlage wurden dafür Datensätze verwendet, welche gute Groundtruth-Daten bereitstellen.

5.1 Betrachtete Datensätze

In den folgenden Abschnitten werden die in dieser Arbeit betrachteten Datensätze erläutert. Es wird der Datensatz selbst beschrieben und dann abhängig von seiner Verwendung entweder wofür er eingesetzt wird oder warum er für eine Verwendung nicht geeignet ist. Des Weiteren werden das Vorgehen bei der eigenen Datensammlung und die entwickelten Methoden für die Augmentation beschrieben.

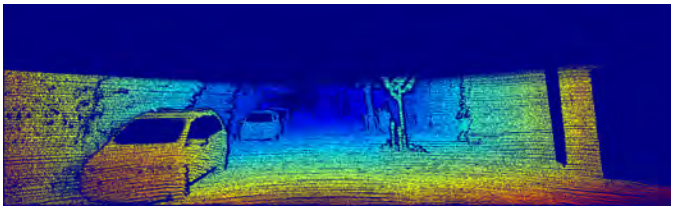
5.1.1 KITTI 2012/2015

Zwei verbreitete Datensätze werden durch die *KITTI Vision Benchmark Suite*[1] bereitgestellt. Die Suite ist eine Sammlung von Datensätzen für Anwendungen wie *SLAM*, *3D Objekterkennung*, *Scene Flow* und Stereo-Vision und bietet zudem die Möglichkeit zur Evaluierung eigener Ergebnisse auf diesen Datensätzen und das Eintragen der Lösung in eine Rangliste für die jeweilige Kategorie.

Der *KITTI 2012 Stereo* [Geiger et al., 2012] Datensatz enthält Aufnahmen von innerhalb der Stadt Karlsruhe, in ländlichen Gebieten und auf Autobahnen aus Sicht eines Fahrzeugs. Dabei werden sowohl rektifizierte RGB-Bilder als auch Schwarz-Weiß-Bilder und Groundtruth in Form von Disparitätskarten bereitgestellt. Die Disparitätskarten wurden aus LiDAR-Daten erstellt, welche über mehrere Frames akkumuliert und auf die Bilder projiziert wurden. Zudem wurden die Punkte aus schwierigen Regionen entfernt, wo die Daten nicht klar und uneinheitlich waren, wie beispielsweise Fahrzeugfenster und Zäunen [Geiger et al., 2012]. Diese Modelle sind nicht millimeter-genau, wie in (Abbildung 5.1) zu erkennen ist, verbessern die verfügbare Disparität aber trotzdem erheblich und lösen damit besonders das Problem mit reflektierenden Oberflächen. Für die Datensammlung wurde ein Auto mit zwei Farb- und Schwarz-Weiß-Kameras und einem Velodyne 3D LiDAR ausgestattet. Das Stereokamera-System mit den verwendeten Farbkameras, deren Bilder in dieser Arbeit verwendet werden, besitzen eine Brennweite von 21 mm und eine Basis von 0.54 m.

[1] https://www.cvlibs.net/datasets/kitti/

(a) RGB Bild (links)



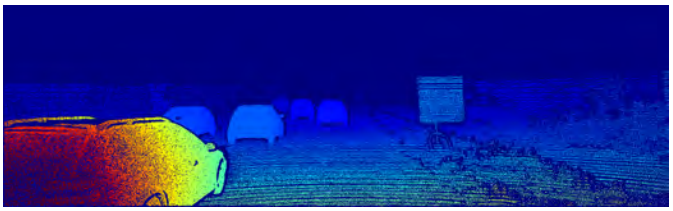
(b) Groundtruth (normalisiert)

Abbildung 5.1: Beispielbilder aus dem KITTI 2012 Datensatz.

Ein weiteres Merkmal ist, dass es keine Aufnahmen von Objekten gibt, welche sich zum Zeitpunkt der Aufnahme bewegt haben, wie zum Beispiel fahrende Autos, sondern nur statistische, beispielsweise parkende Autos. Dies wurde für KITTI 2015 geändert.

Der *KITTI 2015 Stereo* [Menze und Geiger, 2015; Menze et al., 2018, 2015] Datensatz ist grundsätzlich wie der KITTI 2012 Datensatz und bietet Aufnahmen von derselben Datensammlung nur von anderen Szenen. Der große Unterschied ist, dass die im Gegensatz zu KITTI 2012 auch Aufnahmen von sich „bewegenden“ Objekte vorhanden sind. Die Schwierigkeit dabei ist, dass Objekte, welche während der Aufnahme in Bewegung waren, aufgrund der geringen Bildrate (10 Bilder pro Sekunde) und des *Rolling-Shutter-Effekts* nicht genau bestimmbar waren. Für die Verbesserung der Disparitäten dieser Objekte, welche hauptsächlich Fahrzeuge sind, wurden 3D Modelle in die Disparitätskarte mit eingebaut [Menze und Geiger, 2015] (s. Abbildung 5.2b). Dafür mussten die passenden Transformationsparameter gefunden werden, um das Objekt korrekt in der Szene zu platzieren. Für das Finden dieser Parameter werden drei Anhaltspunkte verwendet. Zuerst wurden die verfügbaren Messpunkte für ein Objekt über mehrere Frames auf ca. 3000 Punkte akkumuliert. Durch die Verwendung von *Semi-Global-Matching* wurde dann eine Schätzung der Disparität für das Objekt erstellt. Zuletzt werden manuell 5 bis 10 Korrespondenzen für auffällige, geometrische Teile des 3D Modells herausgesucht. Diese 3 Anhaltspunkte sind Terme einer Energiefunktion, deren Optimierung die passenden Parameter ergibt.

(a) RGB Bild (links)



(b) Groundtruth (normalisiert)

Abbildung 5.2: Beispielbilder aus dem KITTI 2015 Datensatz.

Beide Datensätze sind bereits in einer für das Training neuraler Netzwerke gedachten Struktur mit der Aufteilung in Trainings- und Testdaten. Der Unterschied besteht darin, dass für die Testdaten kein Groundtruth verfügbar ist, sondern nur die linken und rechten RGB-Bilder.
Wichtig zu beachten ist, dass die Werte der Disparitätskarten mit 256 skaliert sind und nach Einlesen der Daten deshalb noch durch 256 geteilt werden muss, um die wirklichen Werte zu erhalten.
Für beide Datensätze existiert zudem jeweils ein *Development Kit (DevKit)*[2,3], mit welchem eine eigenständige Evaluierung eigener Ergebnisse auf den verfügbaren Daten möglich ist. Dabei werden Fehler verschiedener Fehlermetriken pro Bild als auch für den gesamten Datensatz berechnet und Fehlerkarten erstellt.

Sowohl das KITTI 2012 als auch KITTI 2015 wurden für das Training von Netzen verwendet. Zudem wurden augmentierte Version erstellt (siehe Abschnitt 5.3).

[2]KITTI 2012 DevKit https://www.cvlibs.net/datasets/kitti/eval_stereo_flow.php?benchmark=stereo
[3]KITTI 2015 DevKit https://www.cvlibs.net/datasets/kitti/eval_scene_flow.php?benchmark=stereo

5.1.2 Virtual KITTI 2

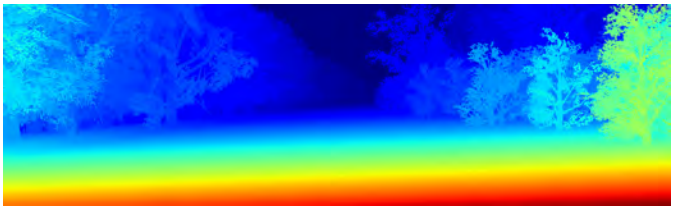
Das *Virtual KITTI 2 (VKITTI2)*[4] [Cabon et al., 2020; Gaidon et al., 2016] ist die zweite Version eines künstlichen, dynamischen und fotorealistischen Nachbildung einiger *KITTI Multi-Object Tracking and Segmentation (MOTS)* Video-Sequenzen in der *Unity Game-Engine*. Das Ziel für die Erstellung des Datensatzes war die Bereitstellung von vollständigem Groundtruth für Training und Evaluierung verschiedene Computer Vision Aufgaben wie Segmentierung, Objekterkennung und Tiefenbestimmung.
Um robuste und repräsentative Ergebnisse liefern zu können, existieren für jede nachgebaute Szene eine Variante mit Nebel, Regen, sonnigem und bewölktem Wetter als auch zur Morgen- und Abenddämmerung. Dies wurde durch unter anderem entwickelte Methoden zum „Klonen“ realer Videodaten zu virtuellen Welten und der automatisierten Generierung von künstlichen Wettereffekten für künstliche Datensequenzen umgesetzt [Gaidon et al., 2016]. Die Referenz-Videosequenzen werden in der Engine nachgebaut, sodass Objekte, Kameraeinstellungen und Beleuchtung übereinstimmen. Durch die Änderung der Beleuchtung und der Verwendung von Partikelsystemen werden Variation von Wetter und Tageszeiten erstellt.

Der Datensatz liefert insgesamt ungefähr 17.000 Frames über 5 verschieden Sequenzen mit jeweils 10 Variationen. Zusätzlich zu den RGB-Stereobildern werden Tiefenkarten, Objekt-Label, Segmentierungs-Label, extrinsische und intrinsische Parameter bereitgestellt. Da in dieser Arbeit mit Disparitäten gearbeitet wird, mussten aus den Tiefenkarten die Disparitätskarte berechnet werden. Dafür kann die Gleichung 2.3 für die Berechnung der Tiefe anhand der Disparität umgestellt werden. Mit den verfügbaren Parametern der Brennweite und Basis konnte somit aus der Tiefe die Disparität errechnet werden.

Im Rahmen dieser Arbeit werden das Szenario mit Nebel und bewölktem Wetter für das Training von Netzwerken verwendet. Die Bilder des Regenszenarios enthalten keine Verdeckung durch Regentropfen, weshalb sie nicht für diese Arbeit geeignet sind.

[4] https://europe.naverlabs.com/research/computer-vision/proxy-virtual-worlds-vkitti-2/

(a) RGB Bild (links)



(b) Groundtruth (normalisiert)

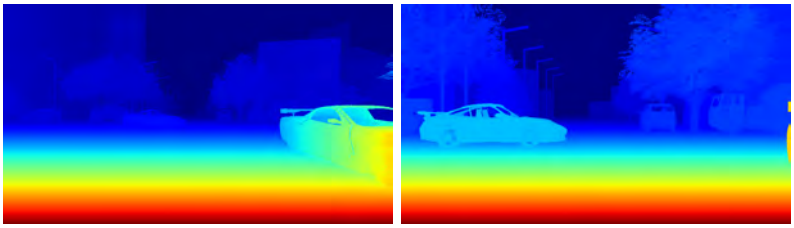
Abbildung 5.3: Beispielbilder aus dem Virtual KITTI 2 Datensatz | Scene18 - Fog

5.1.3 Sceneflow

Der *Sceneflow*-Datensatz[5][Mayer et al., 2016a] der Universität Freiburg ist eine Sammlung künstlich erzeugter Stereobilder mit zugehöriger Groundtruth von unterschiedlichen Szenarien. Der Hauptfokus liegt dabei auf *Scene Flow*, der dreidimensionalen Bewegung zwischen sequentiellen Bildern, wofür aber auch die Disparität notwendig ist und somit auch diese als Groundtruth mit bereitstellt. Der Datensatz wurde mit der 3D Computergrafik-Software *Blender*[6] erstellt. Der Gesamtdatensatz enthält drei Teil-Datensätze unterschiedlicher Szenarien, von dem eines mehrere Sequenzen aus einer simulierten Straßenverkehrssituation darstellt. Das Szenario umfasst eine Version der Aufnahmen mit einer simulierten Brennweite von 35 mm und einer mit 15 mm Brennweite. In Abbildung 5.4 sind für beide Versionen das linke RGB-Bild und die zugehörige Disparitätskarte für dieselbe Szene dargestellt. Die 15 mm Brennweite ist vergleichbar mit den verwendeten Kameraeinstellungen der KITTI Datensätze. Im Vergleich mit den KITTI Datensätzen sind die Bilder weniger detailreich und weniger realistisch, bieten dafür aber als Groundtruth eine volle Disparitätskarte anstatt von einzelnen LiDAR Messpunkten.

[5] https://lmb.informatik.uni-freiburg.de/resources/datasets/SceneFlowDatasets.en.html
[6] https://www.blender.org/

(a) Sceneflow RGB Bild - 15 mm (b) Sceneflow RGB Bild - 35 mm



(c) Sceneflow Groundtruth - 15 mm (d) Sceneflow Groundtruth - 35 mm

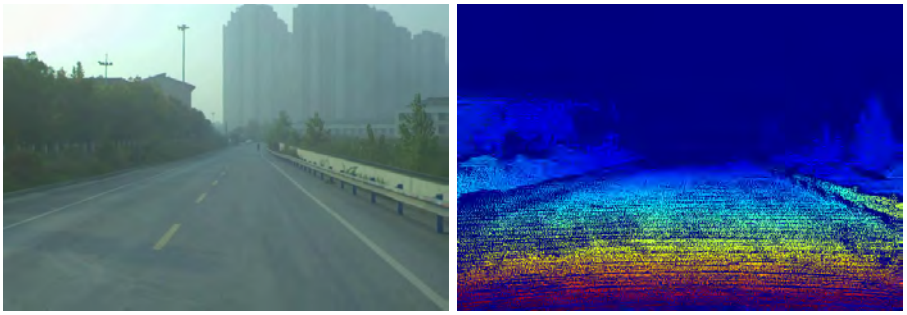
Abbildung 5.4: Beispielbilder aus dem Sceneflow Datensatz.

5.1.4 DrivingStereo

DrivingStereo[7] [Yang et al., 2019] ist ein Stereo-Datenset mit über 180.000 Bildern von städtischen, vorstädtischen, ländlichen Gebieten und Autobahnabschnitten. Darunter befinden sich Aufnahmen bei unterschiedlichen Wetterbedingungen wie Regen, Nebel/Smog, sonnig, bedeckt und bei Abenddämmerung. Der „Nebel“ in diesem Datensatz entspricht mehr dem Smog, der viel in asiatischen Großstädten vorkommt und keinem wirklich dichten Nebel. Eine Szene des Nebelszenarios ist in Abbildung 5.5 dargestellt. In den Aufnahmen sind nicht nur Straßen und Fahrzeuge, sondern auch viele Fußgänger und Natur zu sehen. Bereitgestellt werden RGB-Bilder bei großer unter kleiner Auflösung mit Disparitäts- und Tiefenkarten aus LiDAR-Daten. Durch den Einsatz einer geführten Filterungsstrategie (*guided-filter strategy*) mit dem neuronalen Netzwerk *GuideNet*, bei welcher die RGB Bilder als Orientierung für die Filterung der initialen LiDAR-Daten verwendet werden, wurden Disparitäts- und Tiefenkarten mit hoher Auflösung erstellt [Yang et al., 2019].

Von diesem Datensatz wurde die Daten mit Nebel und bewölktem Wetter für das für das Training von Netzwerken verwendet. In den Aufnahmen in regnerischen Bedingungen finden sich jedoch keine haftenden Regentropfen im Sichtfeld der Kameras, weshalb sie für die Arbeit nicht relevant sind.

[7] https://drivingstereo-dataset.github.io/



(a) RGB Bild - links (b) Groundtruth (Normalisiert)

Abbildung 5.5: Beispielbild aus dem Nebelszenario (Foggy) des DrivingStereo-Datensatzes. (Das Bild ist zugeschnitten.)

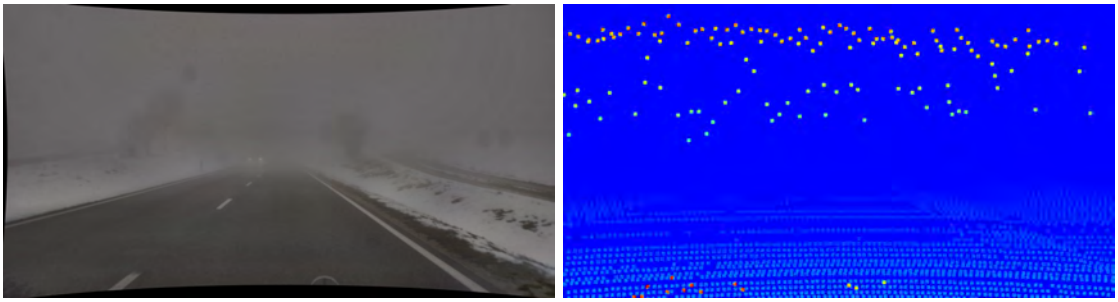
5.1.5 DENSE Datensätze

Im Zentrum des *DENSE* Projektes [8] aus der Zusammenarbeit einiger europäischer Universitäten steht ein System für verlässliches autonomes Fahren in allen möglichen Wetterbedingungen und zu allen Tageszeiten. Der Fokus liegt hierbei auf der Objekterkennung von Fahrzeugen durch die Nutzung verschiedener Sensordaten wie LiDAR, RADAR und Stereo RGB Bildern sowie Bilder von *gated* Kameras. Für diesen Zweck wurde mit diesen Sensoren eine große Menge Aufnahmen in unterschiedlichen Wetterbedingungen von Verkehrsszenarien in Schweden, Finnland und Deutschland gemacht. Aus der gesamten Menge dieser Aufnahmen wurden mehrere Datensätze erstellt.

Einer dieser Datensätze ist der *SeeingThroughFog*[9] Datensatz [Bijelic et al., 2020b], welcher sich in 12.000 Aufnahmen realer Verkehrszenarien und 1500 Aufnahmen kontrollierter Wetterbedingungen in einer Nebelkammer aufteilt. Der Teil der realen Bedingungen umfasst Aufnahme mit Schnee, Regen und Nebel von 10.000 km europäischen Straßen. Die dabei bereitgestellten LiDAR-Messungen sind sehr spärlich, sodass sie nicht als Groundtruth für das Training von Netzen verwendet werden können. Zudem sind diese Daten unter nebeligen Bedingung nicht zuverlässig, wie in 5.6b zu sehen ist. Die Aufnahmen werden deshalb nur für die Evaluierung ohne Groundtruth verwendet.

Zusätzlich zu den Aufnahmen in der Nebelkammer, die Teil des SeeingThroughFog-Datensatzes sind, wurden weitere Aufnahmen für die Evaluierung verschiedener Methoden für Tiefenbestimmung gemacht. In der Nebelkammer wurden die Szenarien einer

[8] https://www.dense247.eu/home/index.html
[9] https://www.uni-ulm.de/en/in/driveu/projects/dense-datasets



(a) Aufnahme einer Szene mit Nebel aus dem SeeingThroughFog-Datensatz.

(b) Disparität basierend auf in Nebel aufgenommenen LiDAR-Daten.

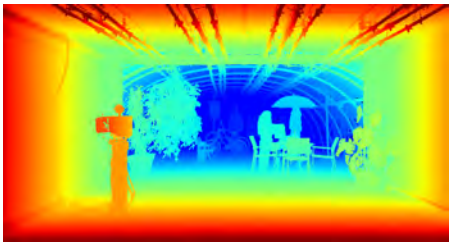
Abbildung 5.6: Aufnahme einer nebeligen Szene und der zugehörigen Disparitätskarte. Die Datenpunkte wurden für bessere Sichtbarkeit stark vergrößert.

Fußgängerzone, Straßenbaustelle, Autobahn und Straße eines Wohngebietes nachgestellt und Aufnahmen für Nebel- und Regenbedingungen mit variirender Intensität und zugehörige hochauflösende Groundtruth-Daten gesammelt. Durch die Verwendung eines hochauflösenden *Leica ScanStation P30 3D-Laserscanners* und die Akkumulation mehrere Punktwolken wurden endgültige Punktwolken mit ca. 50 Millionen Datenpunkten erstellt. Zusätzliche Kontrolle über den Lichteinfall ermöglichte die Szenarien auch für die Bedingungen bei Nacht nach zu simulieren. In Abbildung 5.7 sind zwei Szenen mit Regen und Nebel und den Tiefendaten der jeweiligen Szene ohne Störeffekte zu sehen.
Die Sichtweite bei Nebel wurde mit der meteorologischen Sichtweite verfolgt, die durch $V = -ln(0.05)/\beta$ definiert ist, wobei β die atmosphärische Absorption ist. Die Aufnahmen sind mit Sichtweiten von 20 bis 100 m. Für die Aufnahmen mit Regen wurde Regenfall mit Intensitäten von 15 und 55 mm/h/m^2 simuliert, welche für leichten und starken Regenfall stehen.
Diese Aufnahmen sind im *PixelAccurateDepth*-Datensatz zusammengefasst. Diese Daten können mit dem Projekt *PixelAccurateDepthBenchmark*[10] [Bijelic et al., 2020b] verwendet werden, um eigene Methoden zu evaluieren.

Im Rahmen dieser Arbeit werden die Aufnahmen aus der Nebelkammer und zur Evaluierung der Netzwerke in Bedingungen mit Regen und Nebel genutzt. Vom SeeingThroughFog-Datensatz werden einige Aufnahmen in gutem Wetter und mit Nebel für die Evaluierung verwendet.

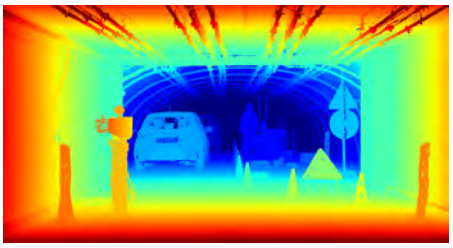
[10] https://github.com/gruberto/PixelAccurateDepthBenchmark

(a) Scene1 Nebel - RGB



(b) Scene1 Nebel - GT

(c) Scene3 Regen - RGB



(d) Scene3 Regen - GT

Abbildung 5.7: Beispielbilder aus dem PixelAccurateDepth-Datensatz.

5.1.6 Waterdrop-Removal Datensatz

Für ihre Arbeit zur Entfernung von Regentropfen aus Stereobildern erstellen Shi et al. [2021] einen eignen Datensatz von Aufnahmen unterschiedlicher innen und außen Szene mit und ohne Wassertropfen. Für die Aufnahmen nutzen sie eine Stereolabs ZED 2 Stereokamera und eine MYNT EYE Kamera mit weitem Sichtfeld. Die Glasscheibe wurde dabei zufällig zwischen 0 und 45°angewinkelt und die Distanz zur Kamera variiert von 2 bis 10 cm. Für jede Szene wurde zuerst eine Aufnahme ohne Tropfen für Groundtruth-Daten gemacht. Daraufhin wurde Aufnahmen mit unterschiedlich vielen Tropfen durchgeführt. Die Szenen umfassen Aufnahmen in kleinen und großen Räumen sowie verschiedene Szenen im Freien. In Abbildung 5.8 ist ein Beispiel einer Szene ohne und mit Wassertropfen abgebildet.
Für die aufgenommenen Szenen existieren zwar Groundtruth-Daten, deren Qualität für Training oder Evaluation nicht ausreichend ist. Im Rahmen dieser Arbeit wurden eigene Aufnahmen solcher Szenen mit einem ähnlichen Vorgehen erstellt. Diese dabei aufgenommen Groundtruth-Daten sind besser für diese Zwecke geeignet. Aus diesem Grund wird dieser Datensatz letztendlich nicht verwendet.

(a) Aufnahme der Szene 91 ohne Wassertropfen.

(b) Eine Aufnahme der Szene 91 mit Wassertropfen.

Abbildung 5.8: Szene 91 des Waterdrop-Removal Datensatzes.

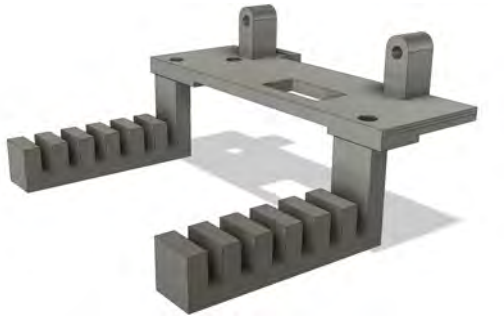
5.2 Eigene Datensammlung

Im Fokus dieser Arbeit steht die Evaluierung des Einsatzes von neuronalen Netzwerken für die Bestimmung von Tiefeninformationen mit durch Umwelteinflüsse gestörten Stereobildern. Im Zuge dessen soll auch ein Vergleich mit alternativen Möglichkeiten stattfinden, welche in dieser Arbeit die Stereokameras Stereolabs ZED 2i und der Luxonis OAK-D Pro sind. Für einen repräsentativen Vergleich der Qualität der Ergebnisse der trainierten Netzwerke und den verwendeten Stereokameras ist es nötig, dass die Bestimmung der Tiefe auf denselben Stereobildern ausgeführt wird. Dafür werden mit den Stereokameras Testbilder für dieselben Szenen unter den gleichen Bedingungen aufgenommen. Zudem ermöglicht dies die Aufnahme von Groundtruth für die Szene ohne Störeffekte und welche als Referenz verwendet werden kann.
Mit diesem festen Aufbau wurden mehrere Szenarien mit unterschiedlichen Störeffekten simuliert. Diese Testbilder umfassen ein Stereobildpaar, eine Disparitätskarte und eine Tiefenkarte der jeweiligen Kamera. Die Stereobildpaare werden für die Bestimmung einer Disparitätskarte durch die trainierten Netzwerke verwendet.

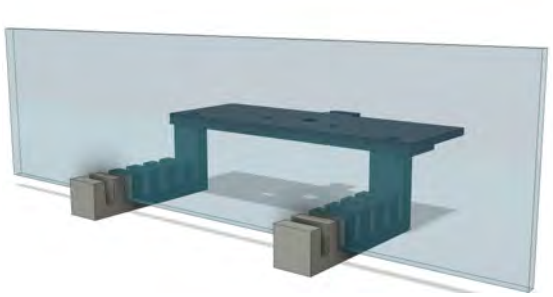
5.2.1 Vorbereitung

Für die Simulation der betrachteten Störeffekte ist es nötig, dass vor den Kameras Wassertropfen, Blätter oder ähnliches zu platzieren. Dafür wird eine Plexiglasscheibe vor den Kameras positioniert, auf welcher diese Objekte „angebracht“ werden können. Dadurch werden die Linsen der Kameras nicht beschmutzt oder beschädigt. Zudem ermöglicht es die Platzierung von Objekten im Sichtfeld beider Kameralinsen und der Verstellung der Distanz.

Dafür werden speziell entwickelte Halterungen verwendet, welche die einfache Positionierung der Plexiglasscheibe vor den Kameras zu ermöglichen. Da die verwendeten Kameras unterschiedliche Größen und Befestigungsmöglichkeiten besitzen, existieren zwei unterschiedliche Halterungen (s. Abbildung 5.9). Diese besitzen zwei Streben mit sechs Steckplätzen in einem Abstand von je 5 mm, in denen die Plexiglasscheibe platziert werden kann. Im von der Kamera aus ersten Steckplatz ist die Scheibe direkt vor der Kamera und im letzten besteht ein Abstand von 5 cm.



(a) Halterung für Luxonis OAK-D Pro



(b) Halterung für ZED2i (mit Plexiglasscheibe)

Abbildung 5.9: Modelle der Kamerahalterungen

Die Stereolabs ZED 2i besitzt ein horizontales Sichtfeld von 110° und eine Basis von 12 cm. Die Aufnahmen werden in Farbe und mit einer Auflösung von 1280×720 gemacht.
Für die Verwendung der ZED 2i ist eine NVIDIA GPU mit CUDA-Unterstützung oder eine TPU notwendig. Zu diesem Zweck wird ein NVIDIA Jetson Nano Entwickler-Kit[11] verwendet. Stereolabs stellt für die Nutzung der ZED-Kamera eine Software-Development-Kit (SDK)[12] zur Verfügung mit einer C++ und Python-API.

[11] https://developer.nvidia.com/embedded/jetson-nano-developer-kit
[12] https://www.stereolabs.com/developers/

Die Luxonis OAK-D Pro besitzt ein horizontales Sichtfeld von 77° und eine Basis von 7.5 cm. Die Stereobilder der OAK-D Pro sind in Schwarz-weiß mit einer Auflösung von 1280×720. Die OAK-D Pro lässt sich wie eine normale Webcam anschließen und verwenden. Für die fortgeschrittene Nutzung kann die AI-Plattform *DepthAI*[13] verwendet werden, welche die Steuerung mit Python erlaubt.

Für den Prozess der Datensammlung wurde ein GUI-Programm entwickelt, welches den Aufnahmeprozess der unterschiedlichen Szenen mit den verschiedenen Kameras erleichtert. Mit einer CSV-Datei können Dateinamen und Unterverzeichnisnamen für unterschiedliche Szenen definiert werden, für welche dann sequenziell die Bilder aufgenommen werden können. Dies ermöglicht die simple Abarbeitung von im Voraus geplanten Szenen und der direkten Speicherung in der richtigen Ordnerstruktur. Die Nutzung der Kameras wird für den Benutzer abstrahiert und lässt sich mit der Stereolabs ZED und der Luxonis OAK benutzen. Es ist auch möglich, die Bilddaten über das Netzwerk zu schicken. Dies war besonders für die Nutzung der ZED 2i mit dem Jetson Nano hilfreich.
Zusätzlich zu der Aufnahme von Einzelbildern ist eine Aufnahme von Bildsequenzen möglich, um schneller eine größere Menge an Bildern aufnehmen zu können.

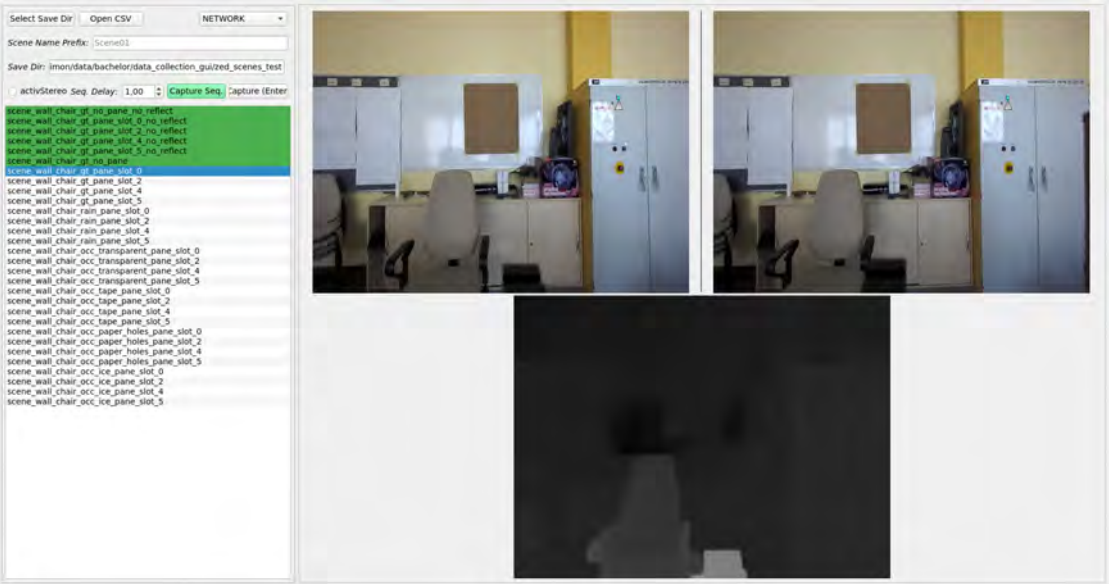


Abbildung 5.10: Screenshot der DataCollectionGui.

[13] https://docs.luxonis.com/en/latest/

5.2.2 Aufnahme der Szenarien

Die Aufnahmen der Szenarien wurden in einem geschlossenem Raum, mit zugezogenen Vorhängen durchgeführt, um möglichst wenig Probleme mit Spiegelung auf der Plexiglasscheibe zu haben. Die für die Aufnahmen gewählte Szene ist in Abbildung 5.11 zu sehen. Sie bietet viele kleine Objekte sowie größere Flächen mit wenig Textur. Im Zentrum ist ein Whiteboard, welches durch das Licht der Fenster hinter der Kamera Spiegelung erzeugt. Ein Teil wurde mit einem rechteckigen Papierstück abgedeckt und dient als eine Referenz für die eigentliche Distanz des Whiteboards. Um einen weiteren festen Referenzpunkt auf mittlerer Entfernung zu haben, wurde ein Stuhl platziert, dessen Anfang der Sitzfläche 100 cm und die Lehne 140 cm Abstand zur Kamera hat. Die Kameras wurden

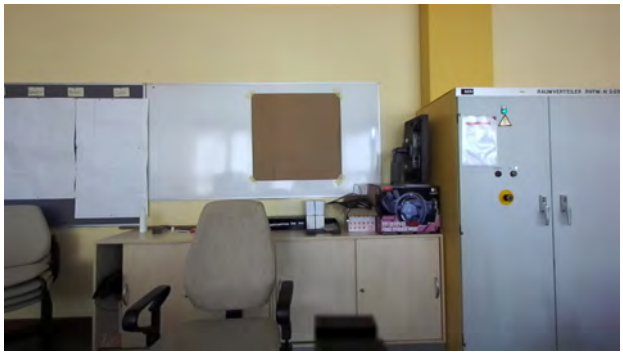


Abbildung 5.11: Szene für die Aufnahmen der simulierten Szenarien.

auf der jeweiligen Halterung montiert und auf einem Stativ auf 120 cm Höhe und mit einem Abstand von 230 cm zur Wand aufgestellt (Abbildung 5.12a). Die Positionierung der Stativ-Beine wurden auf dem Boden markiert, um die gleiche Ausrichtung zu ermöglichen.
Bei der Verwendung der Plexiglasscheibe kam es zu starken Spiegelungen, besonders bei der Platzierung in den letzten Steckplätzen. Als Lösung wurde ein Karton hinter den Kameras befestigt, welcher die Spiegelungen verhindert hat (Abbildung 5.12b).

Zu Beginn wurden erst Referenzaufnahmen der Szene ohne jeglichen Störeffekt gemacht. Danach wurden die Störeffekte aufgenommen. Je Effekt wurde eine Aufnahme mit der Plexiglasscheibe in den Steckplätzen 1, 3, 4 und 6 gemacht. Eine Ausnahme stellt Szenario 4 dar.

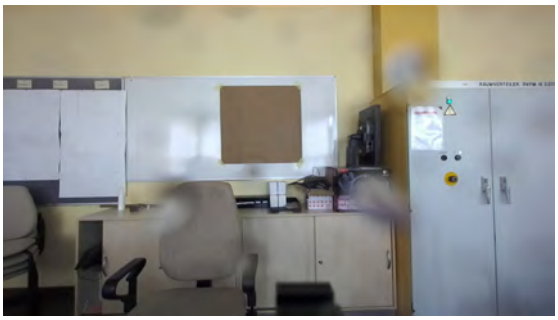
(a) Aufbau mit Stativ und Maßstab für die Aufnahme der Szene.

(b) ZED 2i mit Halterung, Plexiglasscheibe und Karton gegen Spiegelungen

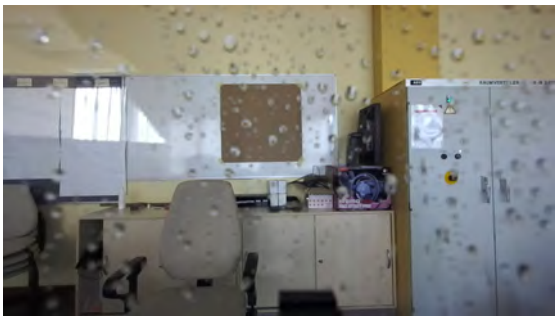
Abbildung 5.12: Szeneaufbau.

Szenario 1: Regentropfen

Der Effekt von Regentropfen wurde durch das Verteilen von Wassertropfen auf der Plexiglasscheibe simuliert. Hierfür wurde eine Sprühflasche verwendet, um die Wassertropfen zu verteilen. Es wurde dabei versucht, so viele Tropfen wie möglich zu platzieren, ohne dass diese verlaufen. Auf den resultierenden Bildern bei Steckplatz 1 sind vereinzelnde, verschwommene Tropfen zu sehen. Je weiter hinten die Plexiglasscheibe ist, desto mehr kleinere Tropfen sind im Bild.
Die Schwierigkeit dieses Szenarios ist, dass der Hintergrund durch vielen Tropfen verdeckt und teilweise verwischt wird. Da die Tropfen willkürlich platziert sind und je nach Distanz nicht in beiden Bildern zu erkennen, bieten sie keine Möglichkeit für das Finden von Korrespondenzen. Die Gefahr ist, dass der verwendete Algorithmus versucht, Disparitäten für die Tropfen zu finden und somit eine „fleckige“ Disparitätskarte entsteht. Bei der Evaluierung wird deshalb auf diese Flecken geachtet.



(a) ZED Szenario 1 - Steckplatz 1



(b) ZED Szenario 1 - Steckplatz 6

Abbildung 5.13: Beispiele vom ZED Szenario 1.

Szenario 2: Einzelne Verdeckung

Eine Variante der Verdeckung ist die durch ein einzelnes Objekt, welches nur einen Teil des Bildes verdeckt. Für diesen Zweck wurde ein Stück Klebeband einer zufälligen Form auf der Plexiglasscheibe platziert, sodass dieses im linken Bild den Stuhl verdeckt. Diese Verdeckung ist abhängig vom Steckplatz der Scheibe unterschiedliche groß. Beim Steckplatz 1 ist das linke untere Viertel des Bildes verdeckt und beim Steckplatz 6 ist nur ein kleiner Teil der Lehne verdeckt.
Die Herausforderung dieses Szenarios ist Erkennung der Disparitäten für die verdeckten Bildausschnitte mit den Informationen aus nur einem Bild. Das gewünschte Verhalten ist demnach, dass das verdeckende Objekt „ignoriert“ wird. Der Fokus bei der Evaluierung liegt auf der Erkennung der Stuhllehne.

(a) ZED Szenario 2 - Steckplatz 1 (b) ZED Szenario 2 - Steckplatz 6

Abbildung 5.14: Beispiele vom ZED Szenario 2.

Szenario 3: Großflächige Verdeckung

Bei der großflächigen Verdeckung wird die Sicht der linken Kamera mit einem größeren, löchrigen Papierstück verdeckt. Durch die Löcher wird nicht die gesamte Sicht verdeckt, sondern nur Teile. Welche Abschnitte verdeckt und wie groß diese sind, ist abhängig von der Distanz zur Kamera. Bei kurzer Distanz in Steckplatz 1 ist ein großer Teil der Bildmitte verdeckt. Je größer die Entfernung wird, desto kleiner werden die verdeckten Flächen, aber auch die Löcher. Das Papierstück ist nur im Bild zu sehen und ist somit einer Erweiterung von Szenario 2. Durch die größere Verdeckung des Hintergrundes im linken Bild gibt es weniger gleiche Bildinformationen für das Finden der Korrespondenzen. Bei der Evaluierung wird auf die Erkennung der Stuhllehne und auf Spuren vom Muster des Papierstückes geachtet.

(a) ZED Szenario 3 - Steckplatz 1 (b) ZED Szenario 3 - Steckplatz 6

Abbildung 5.15: Beispiele vom ZED Szenario 3.

Szenario 4: Stereo-Verdeckung

Bei der Stereo-Verdeckung ist das verdeckende Objekt im Gegensatz zu Szenario 2 in beiden Bildern zu erkennen. Als verdeckendes Objekt wurde hierbei ein Ahornblatt verwendet.
Damit das Blatt in beiden Bildern auftaucht, musste dieses in größerer Distanz „platziert“ werden, als es mit Halterung und der Plexiglasscheibe möglich ist. In der ersten Aufnahme ist das Blatt bei ca. 15 cm Distanz zur Kamera und in der zweiten bei ca. 7 bis 8 cm. Durch die größere Entfernung ist in der zweiten Aufnahme die Lehne des Stuhls zum Teil verdeckt.
Die besondere Herausforderung ist, dass Blatt als störendes Objekt im Vordergrund zu erkennen. Wie in Szenario 3 soll hierbei die Disparität für die verdeckenden Bildinformationen bestimmt werden. Die zusätzliche Schwierigkeit kommt davon, dass das Blatt in beiden Bildern vorhandenen ist und somit die Möglichkeit für das Finden von Korrespondenzen bietet. Bei der Evaluierung dieses Szenario wird besonders auf die Erkennung der Stuhllehne geachtet.

Szenario 5: Transparente Verdeckung

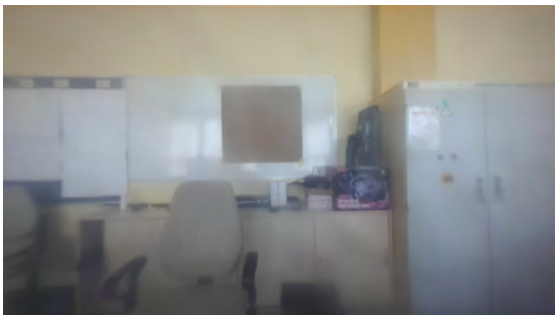
Durch die Verwendung einer kleinen, transparenten Plastiktüte wird eine Bildverunreinigung erzeugt, welche Ähnlichkeit zu einer verschmierten oder beschlagenen Scheibe hat. Die Verdeckung ist dabei sowohl im linken und rechten Bild vorhanden. Bei Steckplatz 1 sind beide Bilder komplett überdeckt. Mit steigender Distanz wird nur noch die gemeinsame Bildmitte verdeckt und die jeweiligen äußeren Bildränder bleiben frei. Durch die Tüte sind Teile des Hintergrundes verschwommen oder werden durch Falten in der Tüte


(a) ZED Szenario 4 - ca. 8 cm Entfernung


(b) ZED Szenario 4 - ca. 15 cm Entfernung

Abbildung 5.16: Beispiele vom ZED Szenario 4.

leicht verdeckt bzw. verzerrt.
Durch diesen Effekt werden Bildinformationen verdeckt und ungleichmäßig verändert. Bei der Evaluierung wird auf die allgemeine Glätte der resultierenden Disparitätskarte geachtet.


(a) ZED Szenario 5 - Steckplatz 1

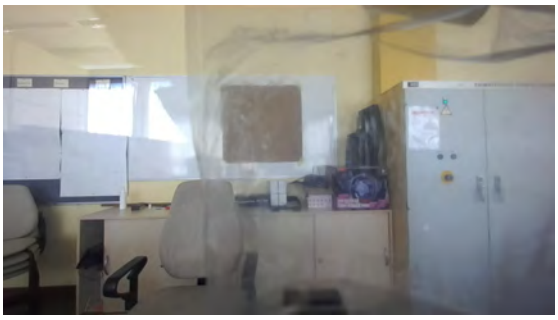

(b) ZED Szenario 5 - Steckplatz 6

Abbildung 5.17: Beispiele vom ZED Szenario 5.

Szenario 6: Frost/Eis

Durch die Verwendung eines Kältesprays wurde der Effekt einer gefrorenen bzw. mit Eis bedeckten Glasscheibe simuliert. Dafür wurde das Spray für ein paar Sekunden auf die Plexiglasscheibe angewendet, um die Bildung von Eiskristallen zu ermöglichen. Da die bei Zimmertemperatur sehr schnell auftaut, mussten die Aufnahmen sehr schnell gemacht werden. Trotzdem musste darauf geachtet werden, dass die Scheibe nicht zu stark bedeckt ist, sodass überhaupt noch etwas zu erkennen ist. Das Resultat ist stark von Dauer und Bewegung beim Auftragen als auch der Temperatur der Scheibe selber abhängig. Auf-

grund dieser schwer kontrollierbaren Faktoren sind die Aufnahmen der beiden Kameras nicht identisch.
Durch das Eis werden Teile der Bilder zum Teil vollständig und teils lückenhaft verdeckt. Zudem führte der Unterschied zwischen der Raumtemperatur und Temperatur der Scheibe dazu, dass die Scheibe in manchen Bildern zusätzlich beschlagen ist. Die dahinterliegenden Bildausschnitte sind dadurch leicht verschwommen. Dieses Szenario ist das schwierigste von allen, da meist das ganze Bild und nicht nur Ausschnitte betroffen sind. Es verbindet zudem den Effekt von transparenter Verdeckung von Szenario 5 und der flächendeckende, aber lückenhaften Verdeckung von Szenario 3.

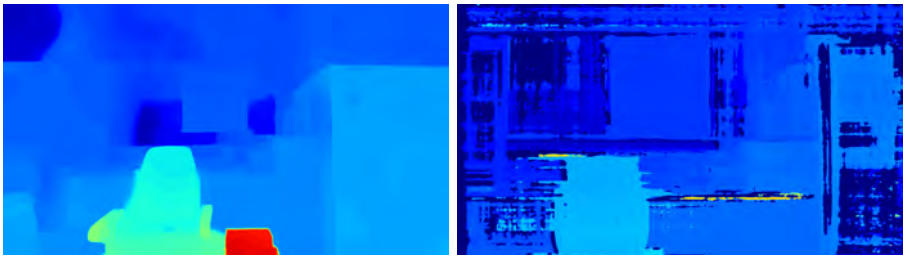
(a) ZED Szenario 6 - Steckplatz 1 (b) ZED Szenario 6 - Steckplatz 6

Abbildung 5.18: Beispiele vom ZED Szenario 6.

5.2.3 Nachbereitung

Die Qualität der Disparitätskarten der Stereokameras für die Aufnahme der Szene ohne Störeffekt sind leider nicht ausreichend gut, um sie als Groundtruth für die Aufnahmen mit Störeffekt zu verwenden. Besonders die spiegelnden Stellen des Whiteboards und die Wand darüber führen zu Fehlschätzungen, wie in Abbildung 5.19 zu sehen ist. Deshalb wurden diese Aufnahmen nachbearbeitet, um die fehlerhaften Stellen zu korrigieren. Dafür wurden Stellen mit zu geringen oder zu extremen Disparitäten durch eine Kombination aus manuell bestimmten Masken und der Nutzung von Median-Filtern an die Umgebung angepasst.

Für die Disparitätskarten der Aufnahmen mit der Stereolabs ZED 2i war eine weitere Bearbeitung nötig. Aufgrund eines Fehlers bei der Erstellung der Halterung für die ZED, waren die Streben, welche die Plexiglasscheibe halten, nicht tief genug befestigt. Dadurch ist das Ende der Streben in den Bildern und demnach auch in den Disparitätskarten



(a) ZED Groundtruth ohne Nachbereitung.

(b) OAK Groundtruth ohne Nachbereitung.

(c) ZED Groundtruth mit Nachbereitung.

(d) OAK Groundtruth mit Nachbereitung.

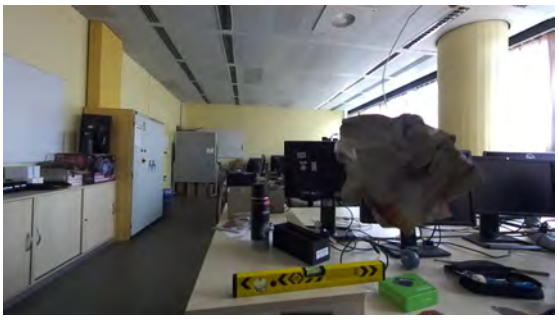
Abbildung 5.19: Nachbereitung der Groundtruth-Disparitätskarten für die Szenarien.

sichtbar. Da dies letztendlich auch einer Verdeckung entspricht, dürften die Strebe in der Groundtruth-Disparitätskarte ohne Störeffekte eigentlich nicht zu sehen sein. Deshalb wurde eine zweite Variante der Groundtruth-Disparitätskarte ohne Störeffekte erstellt, wo die Streben entfernt wurden (s. Abbildung 5.19c). Diese wird nur für die Auswertung der Verdeckungsszenarien verwendet. Für das Regentropfenszenario wird die Variante mit Strebe verwendet.

Aufnahme von Sequenzen

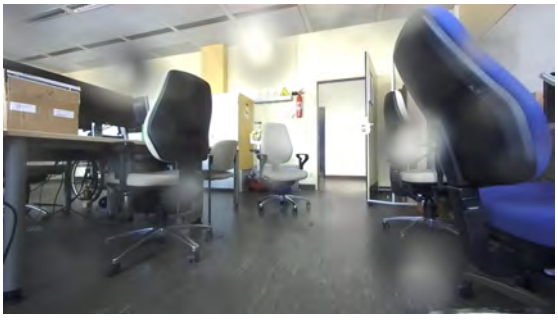
Für das Training eines Netzes werden grundsätzlich viele Daten benötigt. Noch wichtiger ist eine große Varianz der abgebildeten Szenen besteht, sodass eine bessere Generalisierungsfähigkeit erreicht werden kann. Deshalb eignen sich die Aufnahmen der Szenarien nicht für das Training.
Daher wurden insgesamt drei Sequenzen für unterschiedliche Effekte aufgenommenen, welche unterschiedliche Szenen zeigen. Eine Sequenz wurde ohne jegliche Störeffekte aufgenommen. Die zwei anderen beinhalten Aufnahmen mit Wassertropfen und mit Verdeckung. Für die Aufnahme wurde die Stereolab ZED 2i verwendet.

An verschieden Orten wurde zuerst eine Aufnahme ohne Tropfen oder Verdeckung gemacht und danach eine mit dem jeweiligen Störeffekt. Für die Wassertropfen wurde wie in Szenario 2 eine Sprühflasche verwendet, um die Wassertropfen zu verteilen. Einige Beispiel der Aufnahmen mit Regen sind im Anhang in Abbildung A.1 zu betrachten. Für die Verdeckungen wurden unterschiedliche Objekte bei verschiedenen Entfernungen vor der Kamera platziert. Diese konnten entweder mit der Hand oder an einem Faden ins Sichtfeld der Kamera gehalten werden. Einige Beispiel sind im Anhang in Abbildung A.2 zu betrachten.
Im Nachhinein wurden die Groundtruth-Daten der ungestörten Bilder mit den verunreinigten zu je einem Trainingsdatensatz kombiniert. Im Verlauf der Arbeit werden sie mit *ZED_Seq* bezeichnet, mit dem Postfix *clean* für die ohne Störeffekte, *rain* für Regentropfen und *occ* für Verdeckung. Die erste Sequenz ohne Störeffekte umfasst 763 Bilder und die Regentropfen- und Verdeckungs-Sequenzen jeweils 60 Bilder.

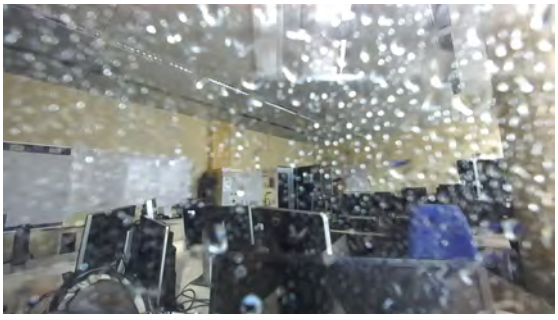


(a) Bild der ZED OCC Sequenzen mit zerknülltem Papier. | Links

(b) Bild der ZED OCC Sequenzen mit zerknülltem Papier. | Rechts



(c) Bild der ZED RAIN Sequenzen mit vielen kleinen Tropfen.



(d) Bild der ZED RAIN Sequenzen mit größeren Tropfen.

Abbildung 5.20: Beispielbilder der aufgenommenen ZED-Sequenzen mit Tropfen und mit Verdeckungen.

5.3 Augmentation von Datensätzen

Eine große Problematik bei Datensätzen realer Aufnahmen von schlechten Wetterbedingungen ist die Beschaffung von korrekten Groundtruth-Daten. Von den im Abschnitt 5.1 beschriebenen Datensätze mit realen Daten stellen alle Groundtruth auf Basis von LiDAR-Daten zur Verfügung. Bei den KITTI Datensätzen ist dies auch kein Problem, da nur Aufnahmen bei gutem Wetter gemacht wurden. Wie bei der Beschreibung des SeeingThroughFog-Datenseatzes erläutert wurde, sind LiDAR für Aufnahmen mit nebeligem Wetter oder liegendem und fallendem Schnee dagegen nicht so gut geeignet.

Eine Lösung ist die Aufnahme von Daten in einer kontrollierbaren Umgebung, wie durch den Einsatz einer Nebelkammer, wie bei dem *Dense*-Projekt (s. Abschnitt 5.1.5). Diese Methode erlaubt zwar Aufnahmen von mehr oder weniger realistische Szenen, ist dabei aber sehr aufwendig und bietet wenig Datenvariation, da die Umgebung immer sehr ähnlich ist. Ein Alternative ist die Nutzung von synthetischen Daten wie zum Beispiel VKITTI 2, wo mit im Vergleich wenig Aufwand verschiedene Wettereffekte simuliert werden können. Der Vorteil ist hierbei, die Verfügbarkeit von ungestörten Groundtruth-Daten. Leider führt das Training auf vollständig synthetischen Daten oftmals zu einer schlechten Generalisierungsfähigkeit für reale Daten.

Eine dritte Möglichkeit ist das Augmentieren von existierenden Daten, welche unter guten Bedingungen aufgenommen wurden. Der Idee ist hierbei Daten zu verwenden, welche standardmäßig gute Groundtruth-Daten mitbringen und die Bilder zu verändern. Durch verschiedene Methoden können Effekte wie Regen oder Nebel den Bildern hinzugefügt werden. Somit erhält man Bilder von schlechten Bedingungen mit korrekten Groundtruth-Daten.

Im Folgenden werden drei verwendete Methoden zur Augmentation von Bildern mit künstlichem Nebel, Regentropfen und Verdeckungen vorgestellt.

5.3.1 Synthetischer Nebel

Als Grundlage für die Erzeugung von künstlichem Nebel, wurde eine Methode aus der Arbeit von Song et al. [2020] verwendet. Inspiriert von Ansätzen anderer Arbeiten und eigenen Ideen, wurde diese Methode über den Verlauf der Arbeit erweitert.

Mit dem bereits erwähnten und viel verwendeten mathematischen Modell (Gleichung 5.1) für die *atmosphärische Streuung*, kann synthetischer Nebel für ein Bild erzeugt werden.

$$\mathbf{I}(x) = \mathbf{J}(x)\mathbf{T}(x) + A(1 - \mathbf{I}(x)) \tag{5.1}$$

Ein Nebel-Bild $\mathbf{I}$ kann mit einer *transmission map* $\mathbf{T}$, welche die Lichtdurchlässigkeit der Szene beschreibt, für ein Bild $\mathbf{J}$ erzeugt werden. Der Faktor A beschreibt dabei das globale atmosphärische Licht, welches die Färbung des Nebels bestimmt. Ein höherer Wert führt zu weißem Nebel und ein niedriger führt zu dunklem Nebel bzw. Rauch. Der Streuungskoeffizienten beeinflusst die Dichte des Nebels.
Die Transmission-Map $\mathbf{T}$ kann durch die Gleichung 5.3 mit der Distanz zur Kamera $\mathbf{Z}$ und dem Streuungskoeffizienten β erhalten berechnet werden. Für die Transmission-Map wird eine Tiefenkarte des Bildes benötigt. Synthetischen Datensätzen wie z. B. VKITTI sind dafür sehr gut geeignet, da sie meist eine vollständige Disparitätskarte mit hohem Detailgrad bereitstellen. Für Datensätze wie KITTI 2015, bei dem nur spärliche Disparitätskarten vorhanden sind, wurde ein bereits trainiertes Netzwerk verwendet, um eine volle Karte zu generieren. In Ausnahmefällen kommt es dabei in manchen Disparitätskarten in Bereichen mit Himmel zu Fehlern.

Um diese zu entfernen, wird eine simple Methode zum Ausschneiden vom Himmel aus der Arbeit von Liu und Klette [2016] verwendet. Sie nutzen dasselbe Modell für die Erzeugung von ästhetischem Nebel für die Aufwertung von Fotos. Da in Bildbereichen mit Himmel oder Wolken der blaue Farbanteil bedeutend höher ist als für Objekte im Vordergrund, kann man einem Schwellwert für den blauen Farbanteil als Filter verwenden, wie in Gleichung 5.2 (aus [Liu und Klette, 2016]) gezeigt ist. Dort, wo der blaue Farbanteil $B_{blue}(p)$ eines Pixel p in der oberen Bildhälfte Ω_{upper} des RGB-Bildes größer ist als der Schwellwert T_{sky}, wird der korrespondierenden Pixel der Disparitätskarte auf ∞ bzw. 0 gesetzt (0 für Disparität, ∞ für Tiefe). Der Wert für den Schwellwert ergab sich aus $T_sky = 0.9 \cdot max(RGB)$.

$$D(p) = \infty \quad if \quad p \in \Omega_{upper} \wedge B_{blue}(p) > T_{sky} \tag{5.2}$$

Zusätzlich wird die Idee der zusätzlichen Verwendung einer Textur aus der Arbeit von Liu und Klette [2016] übernommen. Die führt zu mehr Variation der Nebeldichte im ganzen Bild. Hierfür wird 1/f-Rauschen (engl.: *fractal noise*)(oder auch rosa Rauschen) verwendet, um eine Textur $\mathbf{N}$ für das Bild zu generieren. Diese wird normalisiert, je nach

(a) Farbliches Rauschen in den Bildern des SeeingThroughFog-Datensatzes.

(b) Stark erleuchteter Nebel in Bildern des PAD-Datensatzes.

Abbildung 5.21: Eigenschaften von Aufnahmen von echtem Nebel.

gewünschter Stärke skaliert und mit der Tiefenkarte multipliziert (Gl. 5.3).

$$\mathbf{T}(x) = e^{\beta(\mathbf{Z}(x)\mathbf{N}(x))} \tag{5.3}$$

Bei der Betrachtung einiger Real-Aufnahmen von Nebel aus den PixelAccurateDepth und SeeingThroughFog Datensätzen können zwei Beobachtung gemacht werden. Wie in Abbildung 5.21a zu sehen ist, ist viel farbliches Rauschen im Nebel vorhanden. Dieser Effekt ist zwar durch die Kameras verursacht, ändert nichts daran, dass ein Netzwerk mit solchen Daten funktionieren muss. Um diesen Effekt nachzubilden, wird Gaußsches Rauschen verwendet. Dies wird mit anhand einer Normalverteilung mit einem Erwartungswert $\mu = 0$ und einer Standardabweichung von $\sigma = 1$ erzeugt und auf die Transmission-Map hinzuaddiert.
Die zweite Beobachtung ist der Effekt von Lichtquellen, die den Nebel stellenweise stark erhellen, wie in 5.21b zu sehen ist. Hierfür wird eine Methode aus der Arbeit von Kokubo et al. [2021] verwendet, die eine *glare probablity map* (deut.: Blendung) nutzen, die Lichtbrechung für künstliche Regentropfen zu simulieren. Diese *glare probablity map* gibt für eine vorgeben Position einer simulierten Lichtquelle (x, y) für jedes Pixel des Bildes an, wie große der Einfluss der Lichtquelle für diesen Punkt ist. Wird diese mit dem Faktor A für das atmosphärische Licht kombinierte, wird der Nebel im Einflussbereich der Lichtquelle erhellt.

Ein Beispiel eines Nebel-Bildes ohne und mit Blendung ist in Abbildung 5.22 sehen.

Bei der Augmentation von realistischen Datensätzen, für die erst eine Disparitätsschätzung durchgeführt werden muss, besteht das Problem, dass manche Objekte durch Fehler in der Disparitätskarte stark hervorgehoben werden. Besonders Bäume im Vordergrund

(a) Unverändertes Originalbild.

(b) Mit künstlichem Nebel versehenes Bild ohne Blendung.

(c) Mit künstlichem Nebel versehenes Bild mit Blendung.

Abbildung 5.22: Generierung des künstlichen Nebels - Beispiel an einem Bild des KITTI 2015 Datensatzes.

des Bildes stechen oft stark aus dem Nebel hervor. Dies könnte durch die Verwendung eines besseren Netzwerkes oder Verbesserungen der Disparitätskarten behoben werden.

5.3.2 Synthetische Regentropfen

Es wurde das Projekt *ROLE - Raindrop on Lense Effect*[Chang, 2019] verwendet, welche auf Github verfügbar ist. Die Form der Tropfen wird durch die Kombination eines Ovals und eines Kreises vorgegebenen. Abbildung 5.23a zeigt eine Zeichnung dieser Grundform. Diese Form wird als Maske verwendet, um die dahinter liegenden Bildbereiche durch Weichzeichnen unscharf zu machen. Zudem kann eine die Färbung der Umrandung der Tropfen variiert werden. Dunklere Ränder lassen die Tropfen mehr im Fokus erscheinen.

(a) Grundform eins künstlichen Wassertropfens. Von Chang [2019].

(b) Bild des KITTI15-Datensatzes mit künstlichen Regentropfen.

Abbildung 5.23: Generierung von künstlichen Regentropfen.

Durch Angabe von Wertebereichen für die Anzahl und die Größen werden verschiedene Positionen bestimmt. Wenn zu starke Überlagerung der Positionen bestehen, werden die Tropfen weiter auseinandergesetzt. Ein Resultat dieser Augmentation ist in Abbildung 5.23b zu sehen, wo ein Bild des KITTI15-Datensatzes mit künstlichen Regentropfen erweitert wurde.
Für die Nutzung waren ein paar Verbesserungen für Prüfung der Gültigkeit von Positionen und Überprüfung von Kollisionen nötig.

5.3.3 Verdeckung durch Objekte

Für Erzeugung von Objekten, welche Bildabschnitte verdecken, wurden zwei Methoden entwickelt. Die erste arbeitet mit simplen 2D-Objekten in Form von Bildern, welche mit zusätzlicher Veränderung durch Rotation und rosa Rauschen den Originalbildern hinzugefügt werden. Der Nachteil dieser Variante ist die fehlende Varianz der verdeckenden Objekte. Ein reales Objekt, welches aus unterschiedlichen Perspektiven betrachtet wird, besitzt eine unterschiedliche Form und je nach Beleuchtung auch eine andere Färbung. Zudem ist auch die Textur und Farbe von beispielsweise Blättern nicht immer gleich. Im Versuch, realistischere Daten zu erzeugen, wurde für die zweite Variante die 3D-Computergrafik-Software Blender verwendet. Dadurch konnte das Kamera-System in einer 3D-Szene nachgestellt werden, welches für den Datensatz verwendet wurde, der augmentiert werden soll.
Die erzeugten Verdeckungen lassen sich in zwei Kategorien einteilen. Zur ersten Kategorie gehören Verdeckungen durch Objekte, welche die Sicht auf dahinterliegende Bildinformationen vollständig verdecken. Die zweite Kategorie umfasst lückenhafte Verdeckung mit

transparenten Stellen, wie beispielsweise durch eine gefrorene Scheibe, wo das Originalbild nicht vollständig verdeckt ist. Beide Methoden erzeugen Verdeckung beider Kategorien.
Durch die zufällige Positionierung der verdeckenden Objekte ist das Objekt bei Ergebnissen mal in beiden und mal nur in einem der beiden Bilder zu sehen. Anmerkung: Im Folgenden ist mit dem Begriff *zufällig* ein *pseudozufälliges* Ergebnisse eines Zufallsgenerators gemeint, welches nicht „echt“ zufällig ist.

Methode 1: 2D-Verdeckung

Bei der ersten Methode werden 2D-Bilder von Blättern, Farbflecken, einer Plastiktüte, einem Papierstück und einer Textur von dreckigem Boden als *Formen* für die Verdeckungen der ersten Kategorie verwendet. Einige der verwendeten Bilder sind PNG-Bilder, welche zusätzlich zu den Farbkanälen einen Alpha-Kanal haben, der die Transparenz angibt. Für Bilder anderer Formate wurde vorab der Vordergrund vom Hintergrund getrennt und als PNGs gespeichert.

Zuerst wird für das Objekt eine zufällige Position im linken Bild bestimmt. Mit einer ausgewählten Tiefe und den Kameraparametern des jeweiligen Datensatzes wird die Disparität für diese Position errechnet, mit welcher die Position im rechten Bild bestimmt werden kann. Es wird eine zufällige Form ausgewählt, für welche die zwei Verarbeitungsschritte durchgeführt werden können, um die Varianz der Ergebnisse zu erhöhen.
Der erste Schritt ist die Rotation der Form. Beim zweiten Schritt gibt es zwei Möglichkeiten, bei welchen eine mit 1/f-Rauschen erzeugte Maske verwendet wird.
Eine mit 1/f-Rauschen erzeugtes Bild ist nicht uniform verrauscht, sondern besitzt zusammenhängende Strukturen mit Übergängen zwischen den Pixelintensitäten. Die durch die Maske vorgegebenen Stellen werden entweder ausgeschnitten, um eine willkürliche Form zu erstellen (Beispiel in Abbildung 5.24a) oder transparent gemacht, sodass der Hintergrund noch stellenhaft durchscheint (Beispiel in Abbildung 5.24). Eine Darstellung der einzelnen Schritte ist in Abbildung A.4 zu sehen.

Für die Verdeckung der zweiten Kategorie wurden Texturen von Eis oder eines Glassprungs in einer Scheibe verwendet. Die Bilder der Glassprünge werden nur mit zufälliger Rotation den Originalbildern hinzugefügt. Ein Beispiel ist in Abbildung 5.25b zu sehen. Für die Bilder mit Eis-Texturen wurde ein Schwellenwert verwendet, um Teile der Textur transparent zu machen. Vorab wurde jeweils manuell ein Bereich von Schwellenwerten

(a) Form eines Blattes mit ausgeschnittenen Stellen.

(b) Form eines Blattes mit transparenten Stellen.

Abbildung 5.24: Beispiele für das Ausschneiden und Hinzufügen von Transparenz einer Verdeckungsform.

bestimmt, mit welchem variierende Grade an Transparenz erzeugt werden können. Da die Texturen standardmäßig rechteckig sind, wird eine zufällige Kontur erzeugt, welche auf die Form übertragen wird. Die transparente Form wurde dann über das Originalbild gelegt (Abbildung 5.25a). Die einzelnen Schritte für die Erstellung von Eis-Verdeckungen sind in Anhang in Abbildung A.4 bildlich dargestellt.

(a) Transparente Verdeckung mit einer Eis-Textur

(b) Verdeckung durch ein Glassprung in der Scheibe.

Abbildung 5.25: Beispiele für das Ausschneiden und Hinzufügen von Transparenz einer Verdeckungsform.

Die Augmentation der Bilder wurde mit selbst erstellten Python-Scripten durchgeführt.

Methode 2: 3D-Verdeckung

Für die Erstellung von 3D-Verdeckungen wurde in Blender eine Szene mit einer Stereokamera erstellt, deren Einstellungen dem verwendeten Stereokamera-System des Datensatzes entsprechen, welcher gerade augmentiert wird. Anstatt zweidimensionalen Bildern

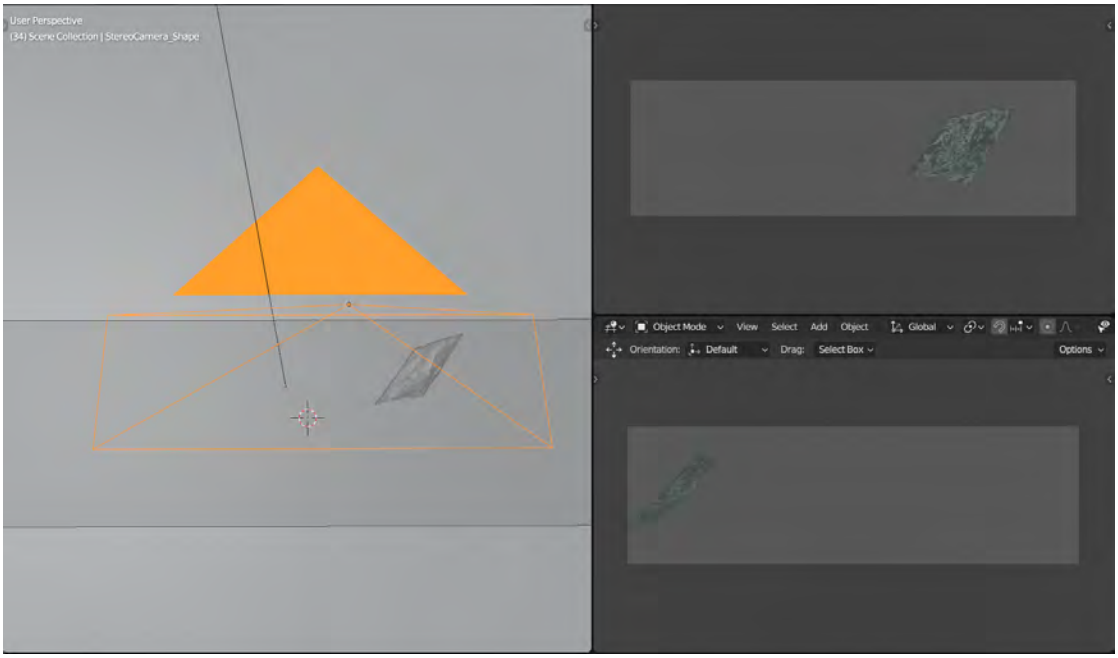


Abbildung 5.26: Aufbau der Szene zur Erstellung von 3D-Verdeckung. Linke Bildhälfte ist die 3D-Sicht der Szene. Rechts-Oben die linke Kamera, rechts-unten die Rechte Kamera.

wurden 3D-Objekte verwendet, deren Aussehen und geometrischen Eigenschaften zufällig verändert wurden. Dieser Aufbau resultierte in unterschiedlicher Erscheinung der Objekte im linken und rechten Bild, wie in Abbildung 5.26 zu erkennen ist.

Die genutzten 3D-Objekte sind von unterschiedlichen Erstellern von der Website Sketchfab[14], welche unter der *Creative Commons Namensnennung 4.0 International Public License*[15] frei nutzbar sind, sofern die Ersteller angegeben werden (siehe Kapitel 9 - Namensnennung). Von dort wurden mehrere Modelle von Blättern und eines zerknitterten Papierstücks ausgewählt. Auch hier werden Teile der Objekte mit einer durch 1/f-Rauschen erzeugten Maske aufgeschnitten, um mehr Varianz in den Ergebnissen zu erreichen. Ein beispielhaftes Ergebnis dieser Operation mit einem Blatt ist in 5.27 dargestellt. Für das Objekt des Papierstücks wurde zusätzlich noch die Färbung zufällig bestimmt.

Es wurden Animationen festgelegt, welche die grobe Position der Objekte bestimmt und diese durch die Sichtbereiche der beiden Kameras bewegen. Zusätzlich werden die Ob-

[14] https://sketchfab.com/
[15] https://creativecommons.org/licenses/by/4.0/legalcode.de

Abbildung 5.27: 3D-Objekt eines Blattes mit hinzugefügten „Ausschnitten“.

(a) Linkes Bild mit 3D-Verdeckung (b) Rechtes Bild mit 3D-Verdeckung

Abbildung 5.28: Beispiele für die 3D-Verdeckung auf dem KITTI 2015 Datensatz.

jekte zufällige rotiert und auf der vorgegebenen Position entlang aller Achsen zufällig verschoben. Dies reduziert die Wahrscheinlichkeit, dass die Ergebnisse selbst bei mehreren Durchgängen unterschiedlich sind. Pro Frame ändert sich das aktuell sichtbare Objekt zufällig, sodass immer nur ein Objekt zusehen ist.
Die Augmentation eines Datensatzes beginnt mit dem Laden der linken und rechten Bilder als Bildsequenzen. Für jedes Bildpaar der Sequenzen wird ein Frame der Animation genutzt. Pro Frame wird ein Bild der Szene für die linke und rechte Kamera erstellt und im Compositor mit den Originalbildern zusammengefügt. Dies funktioniert wie bei Methode 1 anhand der Transparenz der Verdeckungsformen. In Abbildung 5.28 ist ein Ergebnis dieser Augmentation zu sehen. Der Unterschied in der Form des Blattes im linken und rechten Bild ist sehr groß, da die Kameras beim KITTI Datensatz recht weit auseinander sind.

Die Hoffnung bei der Augmentation mit dieser Methode ist, dass sie die in der Realität auftretenden Situationen gut nachgestellt werden. Diese Methode ist zeitaufwendiger als die erste, da das *Rendern* der Szenen je nach Objekt einiges an Zeit beanspruchen.

5.3.4 Verwendete augmentierte Datensätze

Mit den beschriebenen Methoden zur Augmentation von Bildern für verschiedene Arten von Störeffekten wurden einige der in Abschnitt 5.1 beschriebenen Datensätze augmentiert. In Tabelle 5.1 sind die Variationen aufgelistet. Die verwendeten Bezeichner sind *rain* (Regentropfen), *fog* (Nebel), *occ* (2D Verdeckung) und *occ_3d* (3D Verdeckung). Da es für Regentropfen und Nebel auch Datensätze mit realen Aufnahmen gibt, werden diese mit dem weiteren Postfix *real* oder *aug* für „augmentiert“ angegeben.
Eine Übersicht der augmentierten Datensätze ist in 5.1 dargestellt.
Aufgrund des hohen Zeitaufwandes der 3D-Verdeckungs-Methode wurden für den ZED_-Seq-Datensatz nur 200 der 762 Bilder augmentiert.

Datensatz	*fog_aug* (Nebel)	*rain_aug* (Tropfen)	*occ* (2D Verd.)	*occ_3d* (3D Verd.)
KITTI 2015	✓	✓	✓	✓
KITTI 2012	✓	✓	✓	✓
Sceneflow	✓	✓	–	–
ZED_Seq	–	✓	✓	✓

Tabelle 5.1: Übersicht der augmentierten Datensätze.

6 Training der Netzwerke

Für die Beantwortung der Fragestellung, ob neurale Netzwerke für Stereokorrespondenz für den Einsatz in ungünstigen Bedingungen einsetzbar sind, musste die gewählte Netzwerkarchitektur für diese Situationen trainiert werden. Dafür wurden einige der im Kapitel 5 vorgestellten Datensätze und die genannten augmentierten Varianten verwendet. Um die Leistung des Netzwerkes für die einzelnen Störeffekte auswerten zu können, wird das Training dafür separat durchgeführt und keine Variante für alle Effekte trainiert.
Die Trainingsdatensätze für die verunreinigten Daten wurden mit 20 % bis 30 % und bei Nebel teilweise mit 50 % normalen Bilder gemischt. Dies soll verhindern, dass die Netze nichts verlernen, auch auf normalen Daten gute Ergebnisse zu erbringen oder sich etwas Neues anlernt, was nicht auf den normalen Daten funktioniert.

Für die Umsetzung des Trainings wurde das mit der Netzwerkarchitektur bereitgestellte Projekt mit bereits implementierten Trainingsablauf verwendet. Es wurden aber einige Erweiterungen hinzugefügt. Zuerst mussten für die unterschiedlichen Datensätze passende *Dataloader*-Klassen hinzugefügt werden, welche die Daten korrekt einlesen und für das Training aufbereiten. Beim Training werden die Eingabebilder nicht in voller Größe verwendet. Es wurden zufällige Ausschnitte der Größe 512×256 erstellt, wodurch der Speicherbedarf als auch die Gefahr von Überanpassung gesenkt wird.
Zudem bestand eine 70%ige Chance, dass die Bilder horizontal gespiegelt werden. Da bei Datensätze mit LiDAR-Groundtruth die Datenpunkte nur bis zu einer bestimmten Höhe vorhanden sind, kann der Fehler einer Schätzung nie für die Disparitäten in den oberen Bildbereichen bestimmt werden. Damit ein Netz auch lernen kann, für die obere Bildhälfte akkurate Schätzungen zu erbringen, werden die Bilder gespiegelt. So ändert sich die Position des Bildbereiches, für den es kein Groundtruth gibt. Gleichzeitig wird dadurch der Trainingsdatensatz theoretisch erweitert, da für jedes Bild eine gespiegelte Variante existiert.

Das Training wurde hauptsächlich auf einer NVIDIA GTX 1060 6GB ausgeführt. Aufgrund der geringen Menge an VRAM war es nicht möglich, eine Batch-Size größer als 1

für das Training zu wählen. Als Lösung wurde die Möglichkeit hinzugefügt, mit *gradient accumulation* (deut.: Gradienten-Akkumulation)[1] zu trainieren. Mit dieser Technik kann das Vorgehen des normalen Mini-Batch-Verfahrens imitiert werden. Anstatt in einem Trainingsschritt mehrere Bild-Paare zu verwenden, wird immer nur ein Paar genommen. Der Gradient wird nach einem Trainingsschritt aufsummiert, bis eine dafür festgelegte „Batch-Size“ erreicht wurde. Erst dann werden die Gewichte angepasst. Auf diese Weise war es möglich, trotz geringem VRAM eine größere Batch-Size zu simulieren. Für alle endgültigen Trainingsläufe wurde eine Batch-Size von 32 verwendet.

Die Dauer der Trainingsläufe, besonders für Datensätze mit größeren Bildern, war aufgrund der verfügbaren Hardware recht hoch. Zu Unterstützung wurde die Cloud-Plattform *Colaboratory* (kurz colab) von Google genutzt, die es erlaubt, für eigenen Code auf einer GPU auszuführen. Zudem bietet die Netzwerkarchitektur die Möglichkeit, die maximale Disparität zu wählen. Dies verändert die Größe des Kostenvolumens, wie in Abschnitt 4.1 beschrieben wurde. Da eine größere maximale Disparität auch den Speicherbedarf des Netzwerkes erhöht, wurde sich auf eine maximale Disparität von 192 begrenzt, was ein verbreiteter Standard ist.

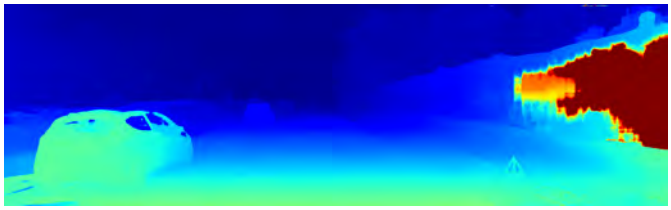
Über den Verlauf der Arbeit wurden viele Trainingsläufe mit verschiedenen Datensätzen und verschiedenen Versionen der Augmentationsmethoden durchgeführt. Da neuer Versionen für die Augmentationen entwickelt wurden, waren die bisherigen Trainingsläufe zu dem Zeitpunkt obsolet. Die hier vorgestellten sind die Trainingsläufe auf den finalen Varianten. Zudem werden auch die Trainingsläufe mit den zuvor verwendeten Netzwerke AANet und PSMNet nicht betrachtet.
Im Folgenden wird erläutert, welche Trainingsläufe mit welchen Daten durchgeführt wurden.

6.1 Basisnetzwerke

Als Grundlage für alle trainierten Netze wurde das von den Autoren bereitgestellte vortrainierte Netz genutzt. Dieses Netz wurde mit dem künstlichen Sceneflow-Datensatz trainiert. Dieses Training resultierte in herausragender Leistung für die Erstellung detaillierter und kontinuierlicher Disparitätskarten. Das ausschließliche Training auf synthetischen Daten führt aber zu suboptimalen Ergebnissen auf realen Daten, sofern das Netzwerk oder die Trainingsdaten nicht dafür angepasst sind.

[1] https://pytorch.org/docs/stable/notes/amp_examples.html

(a) RGB Bild (links)



(b) Schätzung des Netzwerkes.

Abbildung 6.1: Ergebnis des Grundnetzwerkes für das Bild 000197_10 (Training) dem KITTI 2015 Datensatz.

Diese Problematik wurde im Kapitel 5 bereits erläutert. Ein Beispiel einer Schätzung des Grundnetzwerkes auf echten Daten ist in Abbildung 6.1 (und weitere Ergebnisse in A.3) zu sehen. Das Auto im linken Bildabschnitt und das Warnkreuz auf der rechten Seite sind mit einem sehr hohen Detailgrad dargestellt. Die Probleme sind an der Wand auf der rechten Seite zu erkennen, wo die schlechte Beleuchtung dem Netzwerk eine korrekte Schätzung erschwert.

Deshalb war es für die Basisnetzwerke, die nicht auf den verunreinigten Daten trainiert wurden, nötig mit den sauberen Daten einen *fine-tuning* Traininingslauf durchzuführen. Dadurch sollte das vortrainierte Netzwerk, das bisher Gelernte auf die echten Daten transferieren. Somit musste kein komplett neues Netzwerk von Grund auf trainiert werden. Im Folgenden werden diese Netzwerke als *Basisnetzwerke* bezeichnet, da sie nicht auf den Nebel-, Regentropfen- oder Verdeckungsdaten trainiert wurden. Die anderen Netzwerke werden dagegen allgemein als *Störungsnetzwerke* bezeichnet.
Die Anzahl der Epochen wurde dabei abhängig der trainierten Störungsnetze bestimmt. Dies ist wichtig, um aussagekräftige Ergebnisse zu erhalten. Ansonsten könnte man nicht mit voller Gewissheit sagen, dass die Störungsnetze ggf. besser sind, weil sie gelernt haben, trotz Störeffekten Disparitäten korrekt zu bestimmen, sondern sich einfach grundsätzlich besser an die Daten des Datensatzes angepasst haben als das vortrainierte Netzwerk.

Für den Vergleich aller Netzwerk, die auf einer Variante der ZED-Datensätze trainiert haben, wurde ein Netzwerk für 100 Epochen auf den Daten von ZED_Seq_clean trainiert.

Die beste Epoche war am Ende die letzte mit einem durchschnittlichen Validierungs-Loss (Val-Loss) von 2,170. Als Referenz-Netz für alle auf den KITTI-Datensätzen trainierten Netzen wurde das KITTI_Basisnetz für 115 Epochen auf den normalen KITTI-Daten trainiert. Mit eine Val-Loss von 0,643 ist die Epoche 115 dabei die beste. Für beide Trainingsläufe wurde eine Lernrate von 0.001 und eine Batch-Size von 32 verwendet. Die Parameter sind in Tabelle 6.1 dargestellt.

Netzwerk-Name	**Datensatz**	**Epochen**	**Lernrate**	**Beste Epoche & Val-Loss**
ZED_BASIS	ZED_Seq_clean	100	0.001	100 | 2,170
KITTI_BASIS	KITTI12_15_normal	115	0.001	115 | 0,643

Tabelle 6.1: Trainingsparameter für die Basisnetzwerke.

6.2 Regentropfennetzwerke

Für das Training der Regentropfennetzwerke wurden der selbsterstellte ZED-Seq_rain_-real-Datensatz mit den echten Regentropfen und die augmentierte Variante des ZED-Sequenz-Datensatzes ZED-Seq_rain_aug verwendet. Das ZED_RAIN_AUG-Netzwerk wurde für 100 Epochen trainiert, von denen die letzte Epoche die beste war.

Das ZED_RAIN_REAL-Netzwerk wurde für insgesamt 700 Epochen trainiert, da der Trainingsdatensatz, bestehend aus Bildern von ZED-Seq_rain_real und einigen sauberen Bildern von ZED_Seq_clean ca. 1/7 der Größe des Trainingsdatensatzes vom ZED_-RAIN_AUG-Netzwerk ist. Durch das längere Training wird ungefähr dieselbe Menge an Bildern im Trainingsprozess betrachtet. Die besten Ergebnisse wurden bei Epoche 375 erreicht.

Des Weiteren wurde ein Netzwerk mit dem augmentierten KITTI-Datensätzen trainiert. Dies wurde für 115 Epochen trainiert und erzielte die besten Ergebnisse bei Epoche 113. Die Einzelheiten sind in Tabelle 6.2 zu betrachten.

Netzwerk-Name	**Datensatz**	**Epochen**	**Lernrate**	**Beste Epoche & Val-Loss**
ZED_RAIN_REAL	ZED-Seq_rain_real	700	0.0001	375 | 2,432
ZED_RAIN_AUG	ZED-Seq_rain_aug	100	0.0001	100 | 2,064
KITTI_RAIN_AUG	KITTI12_15_rain_aug	115	0.0001	113 | 0,685

Tabelle 6.2: Trainingsparameter für die Regentropfennetzwerke.

6.3 Nebelnetzwerke

Für das Training der Nebelnetzwerke wurden die mit Nebel augmentierten KITTI-Datensätze verwendet. Das Training stellte sich hierbei als sehr viel schwieriger heraus als bei den anderen Störeffekten. Bei starkem Nebel sind viele Bildausschnitte fast nur weiß bzw. grau. Da beim Trainingsprozess immer nur Ausschnitte der Bilder betrachtet werden, war es möglich, dass in diesen Ausschnitten oft nur wenig bis gar keine Strukturen zu erkennen waren. Dies ist gefährlich für das Training, da das Netz versuchen könnte, irgendetwas in der weiß-grauen Fläche zu erkennen, obwohl nichts zu erkennen ist. Das Netz könnte sich durch diese Situationen etwas anlernen, was nichts mit den eigentlichen Strukturen zu tun hat.
Auch ist die Gefahr groß, dass das Netz verlernt, mit normalen, nicht nebeligen Bildern zu funktionieren, da diese nicht so einen großen Anteil an weiß oder grau über dem ganzen Bild besitzen. Hierfür war es wichtig genug normale Daten zu den Trainingsdaten hinzuzufügen und keine zu große Lernrate zu verwenden.

Insgesamt wurde das KITTI_FOG-Netzwerk für 500 Epochen trainiert, von denen die letzte Epoche die besten Ergebnisse brachte. Mit einem gemischten Trainingsdatensatz aus dem bewölkten und nebeligen Szenario des DrivingStereo-Datensatzes wurde das DS_FOG-Netzwerk für 100 Epochen trainiert. Die letzte Epoche erwies sich dabei als die beste. Die genauen Trainingsparameter sind in Tabelle 6.3 dargestellt.

Netzwerk-Name	**Datensatz**	**Epochen**	**Lernrate**	**B. Epoche & Val-Loss**
KITTI_FOG	KITTI12_15_fog_aug	500	0.00001	500 | 1,047
DS_FOG	DS_foggy_cloudy	100	0.0001	100 | 0,611
VKITTI_FOG	VKITTI_fog_overcast	22	0.0001	22 | 0,813

Tabelle 6.3: Trainingsparameter für die Nebelnetzwerke.

Im Kapitel 5 wurde erläutert, dass vollständig synthetische Datensätze meist zu schlechter Generalisierung auf echten Daten führen. Da die Menge an verfügbaren Datensätzen mit Nebel nicht sehr groß war, wurde trotzdem ein Training auf dem VKITTI-Nebel-Datensatz durchgeführt. Dies hatte zudem den Vorteil, selbst einen Vergleich zwischen den realen Datensätzen mit augmentiertem Nebel und einem vollständig-künstlichem Nebel-Datensatz durchführen zu können. Für das Training wurden die Bilder des VKITTI_fog-Datensatzes mit Bildern des VKITTI_overcast-Datensatzes gemischt, welche bewölktes Wetter beinhalten. Aufgrund der Größe des Trainingsdatensatzes wurde sich auf

eine Trainingsdauer von 22 Epochen beschränkt. Die aus dem Training resultierende beste Epoche war die letzte Epoche.

6.4 Verdeckungsnetzwerke

Die Störungsnetzwerke für die Verdeckung wurden auf dem selbst erstellen ZED_occ_-real und den augmentierten Varianten ZED_occ_aug_2d und ZED_occ_aug_3d trainiert. Das ZED_OCC_REAL-Netzwerk wurde für 700 Epochen trainiert, von denen die Epoche 600 die beste war. Da der Trainingsdatensatz für ZED_OCC_AUG_2D ca. sechsmal so groß ist wie der von ZED_OCC_REAL, wurde es nur für 100 Epochen trainiert. Die daraus resultierende beste Epoche war die letzte.
Das KITTI_OCC_AUG_3D-Netzwerk wurde aus dem gleichen Grund für 330 Epochen trainiert. Die letzte Epoche stellte sich als die beste heraus.

Netzwerk-Name	**Datensatz**	**Epochen**	**Lernrate**	**B. Epoche & Val-Loss**
ZED_BASIS_PATCH	ZEQ-Seq mit Patches	60	0.0001	60 | 2,379
ZED_OCC_REAL	ZED-Seq_occ_real	700	0.0001	600 | 1,598
ZED_OCC_AUG_2D	ZED-Seq_occ_aug_2d	100	0.0001	100 | 2,680
KITTI_OCC_AUG_3D	ZED-Seq_occ_aug_3d	330	0.0001	330 | 1,502

Tabelle 6.4: Trainingsparameter für die Verdeckungsnetzwerke

Bei der Entwicklung der Augmentationsmethode für die 2D-Verdeckungen ergab sich zu Beginn eine Variante, bei welcher einfarbige Formen als Objekte genutzt wurden. Dies wurde zwar in späteren Versionen geändert, resultiere aber Idee, auch ein Netzwerk mit dieser Art von Verdeckung zu trainieren. Dies wurde nicht mit einem separaten Datensatz umgesetzt, sondern durch dynamische Augmentation der Bilder während des Trainings. Für jedes Bild existierte ein 50 % Wahrscheinlichkeit, dass ein bis drei schwarz Rechtecke (Patches) einer zufälligen Größe von 20 px bis 50 px erstellt werden. Mit dieser Methode wurde ein zusätzliches Netzwerk für 60 Epochen trainiert. Mit diesem Netzwerk kann untersucht werden, ob allein zufällige und asymmetrische Verdeckung mit simplen Formen bei der Disparitätsschätzung mit Verdeckung durch Objekte helfen kann. Die verwendeten Trainingsparameter und Bezeichner sind in Tabelle 6.4 dargestellt.

Im folgenden Kapitel wird die Evaluierung der trainierten Netze erläutert und die Ergebnisse besprochen.

7 Auswertung

In diesem Kapitel wird die Auswertung der trainierten Netzwerke beschrieben. Für verschiedene Testdaten werden die von den Netzen erzeugten Disparitätskarten anhand einiger Metriken bewertet und miteinander verglichen. Die Störungsnetzwerke werden dabei nur für den jeweiligen Störeffekt ausgewertet.

7.1 Metriken

Um die Leistung der Netzwerke bewerten zu können, werden die Ergebnisse anhand mehreren Metriken verglichen. Hierfür werden insgesamt 3 Metriken verwendet. Die Metriken lassen sich in die Kategorien *Groundtruth-* und *Bildqualitätsmetriken* einteilen.

7.1.1 Groundtruth-Metriken

Für alle Datensätze, für die ausreichend gute Groundtruth-Daten zur Auswertung der Ergebnisse zur Verfügung standen, wurden die standardmäßig verwendeten Metriken des Endpoint-Error (EPE) und des D1-Fehlers verwendet. Der EPE ist die absolute Differenz zwischen der geschätzten Disparität und dem Groundtruth-Wert. Für die gesamte Schätzung wird der Mittelwert des EPE aller Pixeln der geschätzten Disparitätskarte Y_i und den Groundtruth-Pixeln $\hat{Y}_i$ genommen, was dem mittleren absoluten Fehler (MAE) (Gl. 7.1) entspricht.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |\hat{Y}_i - Y_i| \tag{7.1}$$

Die D1-Metrik beschreibt den durchschnittlichen prozentualen Anteil der Ausreißer. Ein Pixel ist ein Ausreißer, wenn der EPE für das Pixel größer als 3 px und größer als 5 %

der maximalen Disparität ist (Gl. 7.2).

$$D1 := 3\ px > |D_{gt} - D_{est}| > |D_{gt} \cdot 0.05|\ px \tag{7.2}$$

Zusätzlich wird noch der prozentuale Anteil der Disparitäten betrachtet, deren EPE größer als 1 px ist.
Diese Metriken können jedoch nicht für die Evaluierung von Disparitätsschätzungen verwendet werden, für die keine ausreichend guten Groundtruth-Daten existieren, wie bei dem SeeingThroughFog-Datensatz (STF). Für die Evaluierung dieser Daten wird deshalb eine Metrik verwendet, welche die Bildqualität der mit der geschätzten Disparitätskarte erstellten Rekonstruktionen des rechten Bildes bewertet.

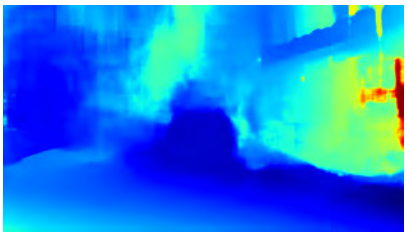
7.1.2 Bildqualitätsmetriken

Für ein linkes Bild eines Stereobildpaares werden die Pixel des linken Bildes anhand der zugehörigen Disparitäten der geschätzten Disparitätskarte verschoben. Das resultierende Bild ist eine approximierte Rekonstruktion des originalen rechten Bildes bzw. eine Projektion der linken Bildpunkte anhand der Disparitäten. Da der rechte Bildrand des rechten Bildes im linken Bild nicht zu sehen ist und die Disparitätskarte aus Sicht der linken Kamera erstellt wurde, kann dieser Bildbereich nicht rekonstruiert werden.
Daher wird bei dem Vergleich nur der Bildbereich betrachtet, welcher in beiden Bildern zu erkennen ist. Ein Beispiel einer solchen Rekonstruktion im Vergleich zum originalen rechten Bild ist in Abbildung 7.1 zu sehen. Die dort dargestellte Rekonstruktion basiert auf einer fehlerhaften Disparitätskarte (7.1c), wodurch das Ergebnis an einigen Stellen leicht bis stark verzerrt ist. Die Fehlerkarte in Abbildung 7.1d zeigt die Wurzel der mittleren quadratischen Abweichung (RMSE) der RGB-Pixel der Rekonstruktion verglichen mit dem Original. Der Fehler gibt dabei nur an, wie stark sich die Rekonstruktion vom Original unterscheidet und nicht wie weit die Disparitäten daneben liegen.

Diese Rekonstruktion kann mit dem echten rechten Bild verglichen werden, um die Korrektheit der transformierten Pixel und damit indirekt die dafür verwendeten Disparitäten zu bewerten. Diese Methode wird auch viel die Berechnung des Fehlers für unüberwachtes Lernen angewendet (s. Abschnitt 2.2.1) [Zhang et al., 2022]. Pixel-basierte Metriken wie beispielsweise MAE sind hierfür weniger geeignet, da das linke und rechte Bild oftmals Unterschiede in der Helligkeit oder Sättigung aufweisen. Die Rekonstruktion würde dadurch andere Werte besitzen als das Referenzbild, auch wenn der eigentliche Bildinhalt

(a) Originales rechtes Bild mit einer leichten grün/bläulichen Färbung.

(b) Rekonstruktion des rechten Bildes mit einem wärmeren Farbton.



(c) Die für die Rekonstruktion verwendete fehlerhafte Disparitätskarte.

(d) Fehlerkarte der Rekonstruktion. Die Farbskala ist oben links.

Abbildung 7.1: Beispiel einer Rekonstruktion eines des rechten Bildes auf Basis des linken Bildes.

korrekt dargestellt ist. Dies lässt sich im Vergleich von Abbildung 7.1a und 7.1b erkennen. Besser geeignet sind Metriken, die für den Vergleich besonders die vorhandenen Strukturen in den Bildern berücksichtigen.

Es existieren einige Metriken für die Bewertung von Bildqualität (engl. Image Quality Assessment (IQA)), welche die Qualität in Hinsicht des menschlichen visuellen Systems bewerten sollen. Im Gegensatz zu Pixel-basierten Metriken wie z. B. MAE, werden keine Vergleiche von einzelnen Pixeln durchgeführt, sondern Strukturen und Wertveränderungen verglichen. Die in dieser Arbeit verwendete Metrik ist der *Feature Similarity Index* (FSIM) [Zhang et al., 2011].

Bei der Metrik werden für den Vergleich zwei Kriterien verwendet: die Phasenkongruenz und der Betrag des Bildgradienten. Ein Filter wird mit verschiedenen Frequenzen zur Filterung der Bilder angewendet und resultiert in mehreren Filterantworten. Visuell erkennbare Merkmale in Bildern stimmen mit den Punkten überein, wo mehrere Filterantworten *kongruente* (übereinstimmende) Phasen besitzen. In diesem Fall wird das *log-Gabor* Filter verwendet. Phasenkongruenz ist invariant gegenüber Kontrastunterschieden, wodurch die gleichen Merkmale bei unterschiedlichem Kontrast gefunden werden können. Da Kontrast aber auch Einfluss auf die menschliche Wahrnehmung hat,

muss dieser für die Bewertung trotzdem miteinbezogen werden. Dafür wird der Bildgradient verwendet, welcher durch die Verwendung des Sobel-, Prewitt- oder Scharr-Operators erhalten werden kann. Der Betrag des Gradienten (Gl. 7.3, aus [Zhang et al., 2011]) besteht aus den partiellen Ableitungen entlang der Horizontalen $G_x(x)$ und der Vertikalen $G_y(x)$ eines Bildes $f(x)$.

$$G = \sqrt{G_x^2 + G_y^2} \tag{7.3}$$

Diese Art der Evaluierung der Korrektheit von Disparitätskarten ist unter den Umständen fehlender Groundtruth-Daten zwar eine gute Alternative, ist aber nicht vollkommen verlässlich. Repetitive Muster in den Bildern können zu Fehlern in der Rekonstruktion führen [Zhang et al., 2022]. In einer Szene, in der beispielsweise zwei Autos mit gleicher Farbe nebeneinander zu sehen sind, können in der Rekonstruktion Pixel zum rechten Auto zugeordnet werden, die eigentlich zum linken Auto gehören. Dies kann durch fehlerhafte Disparitäten verursacht werden. Diese Fehler können anhand der beschriebenen Evaluierungsmethode gegebenenfalls nicht als Fehler erkannt werden, da der Farbwert übereinstimmt. Auch spiegelt die Größe des Fehlers nicht die Größe der Differenz zwischen der geschätzten und der gewünschten Disparität wider. Dies muss bei der Betrachtung der im Folgenden darstellten Ergebnisse beachtet werden.

7.2 Allgemeines Vorgehensweise

Jedes trainierte Netzwerk wurde auf dem normalen Testdatensatz ohne Störeffekte und auf mindestens einem Datensatz mit Störeffekten ausgewertet. Dabei wurde für den ganzen Testdatensatz der Mittelwert der jeweiligen Metrik bestimmt. In den meisten Fällen existieren mehrere Datensätze mit Störeffekten beispielsweise einer mit echten und einer mit augmentierten Bildern. Durch die Evaluierung mit den Daten ohne Störeffekte soll getestet werden, ob das Netzwerk etwas gelernt hat, was sich nur auf die Daten mit Störeffekten anwenden lässt. Mit den Daten mit Störeffekten soll geprüft werden, wie gut ein Netzwerk mit den jeweiligen Störeffekten umgehen kann. Dabei wird im Besonderen auch auf den Unterschied der Ergebnisse bei Verwendung von echten und augmentierten Trainingsdaten geachtet.
Die Bewertung der Ergebnisse anhand der Metriken werden dabei in Tabellen dargestellt. Das jeweils beste Ergebnis für eine Metrik (einer Spalte) wird **fett** und das schlechteste

(a) Farbskala „Jet“. (b) Farbskala der Fehlerkarten.

Abbildung 7.2: Verwendete Farbskalen für die Disparitäts- und Fehlerkarten.

Ergebnis *kursiv* hervorgehoben. Zusätzlich werden für das beste, das schlechteste und weitere Netzwerke die geschätzten Disparitätskarten gezeigt. Für die Darstellung wird die Farbskala „Jet“ (Abb. 7.2a) verwendet. Für bessere Sichtbarkeit wird die maximale Disparität den Datensätzen angepasst. Für die Datensätze, die mit Groundtruth-Daten ausgewertet werden können, wird zudem die zugehörige Fehlerkarte präsentiert. Diese wird auf anhand der D1-Metrik erstellt und verwendet die in Abbildung 7.2b dargestellte Farbskala.
Bei den Datensätzen, für welche die Auswertung anhand der Rekonstruktion durchgeführt wurde, werden anstatt der Fehlerkarte diese Rekonstruktionen gezeigt.

Für die Evaluierung der Netze für die aufgenommenen Testszenarien werden die Aufnahmen der Stereolabs ZED 2i verwendet. Der Grund dafür ist die höhere Auflösung der Bilder und die bessere Qualität der Disparitätskarten im Vergleich zu Luxonis OAK.
Bei Betrachtung der Ergebnisse der Netzwerke für die aufgenommenen Test-Szenarien werden zudem die Ergebnisse der Stereokameras hinzugezogen. Zu beachten ist, dass die Disparitätsschätzung der Luxonis OAK für die Aufnahmen mit der Luxonis OAK sind. Da die Basis und das Sichtfeld geringer sind als bei der ZED 2i, sind die Aufnahmen der Luxonis OAK nicht vollständig vergleichbar. Die Schätzungen werden aus diesem Grund mit den Groundtruth-Disparitätskarten der Luxonis OAK ausgewertet.

7.3 Ergebnisse der Regentropfennetzwerke

Die Evaluierung der Regentropfennetzwerke teilt sich in die Betrachtung der Netzwerke, die mit den ZED-Datensätzen und denen, die mit den KITTI-Datensätzen trainiert wurden. Für die Auswertung der KITTI-Netzwerke wird der normale und mit Regentropfen augmentierte KITTI15-Datensatz verwendet. Außerdem werden die Aufnahmen des PixelAccurateDepth-Datensatzes (PAD) mit zwei Regenstärken als realistisches Szenario verwendet. Die Auswertung der ZED-Netzwerke wird anhand des aufgenommenen Szenarios mit Regentropfen (s. Abschnitt 5.2.2) und Testdaten der ZED-Sequenz mit Regentropfen ausgewertet.

7.3.1 KITTI-Netzwerke

Das trainierte KITTI_RAIN-Netzwerk und das Basisnetzwerk werden mit den normalen KITTI15-Daten ohne Störeffekte und dem mit Regen augmentierten Datensatz ausgewertet. Die Ergebnisse sind in Tabelle 7.1 dargestellt.

Auswertung für die KITTI15-Testdaten

Testdaten	Netzwerk-Name	EPE (px)	D1-all(%)	> 1 px (%)
KITTI15_NORMAL	KITTI_BASIS	*0,807*	*2,065*	*17,247*
	KITTI_RAIN	**0,723**	**1,781**	**15,272**
KITTI15_RAIN	KITTI_BASIS	*1,301*	*6,433*	*26,409*
	KITTI_RAIN	**0,823**	**2,685**	**18,073**

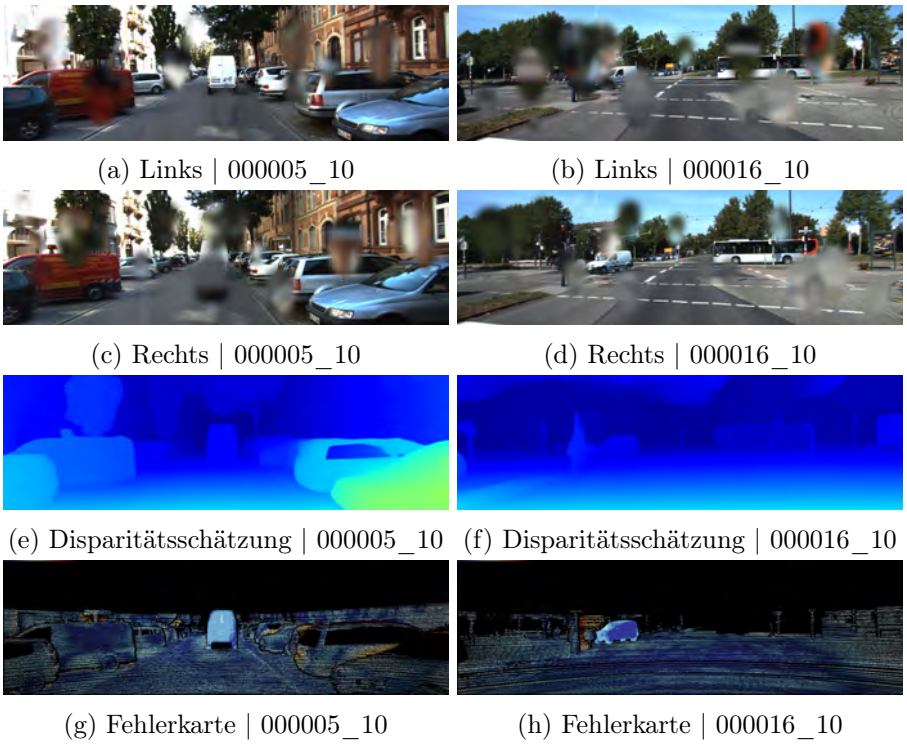
Tabelle 7.1: Ergebnisse der KITTI-Netzwerke für KITTI15_NORMAL und KITTI15_-RAIN.

Für die Testdaten des KITTI15_NORMAL-Datensatzes erbringt das KITTI_RAIN-Netzwerk im Vergleich zum Basisnetzwerk bessere Ergebnisse. Bei Begutachtung der Disparitäts- und Fehlerkarten zeigt sich, dass dies nicht durch etwa große Fehler des Basisnetzwerks, sondern allgemeine genauere Schätzungen des KITTI_RAIN-Netzwerks zustande kommt. Dies zeigt, dass das trainierte KITTI_RAIN-Netzwerk auch für Daten ohne Regentropfen korrekte Schätzungen erzeugen kann. Zudem ist es durchschnittlich besser als das Basisnetzwerk. Die für die Regentropfen-Daten benötigte Nutzung von mehr Kontextinformationen verbessert somit auch die Ergebnisse für normale Daten.

Die Ergebnisse der Netze für die KITTI15_RAIN-Testdaten zeigen eine größere Differenz der Genauigkeit zwischen dem Basis- und dem KITTI15_RAIN-Netzwerk. Das Basisnetzwerk hat Schwierigkeiten, Disparitäten für verdeckte Objekte wie zum Beispiel Autos vollständig zu erkennen. Auch führen die Regentropfen zu fehlerhaften „Flecken“ in der Disparitätskarte.
Im Vergleich zu den Ergebnissen für die normalen Daten hat sich das KITTI_RAIN-Netz im Allgemeinen nur leicht verschlechtert. Die Unterschiede des EPE, D1-Fehlers und des 1-Pixel-Fehlers bedeuten, dass die Schätzungen im Durchschnitt nicht viel ungenauer sind, sondern vereinzelte Stellen mit größeren Abweichungen hinzugekommen sind.

Dies lässt sich durch die Betrachtung der in Abbildung 7.3 dargestellten Disparitäts- und Fehlerkarten bestätigen. Auf der rechten Seite ist die Szene *000016_10* dargestellt, wo

im linken Bild die Ampel vollständig und auf der rechten Seite leicht durch Regentropfen verdeckt ist. In der geschätzten Disparitätskarte fehlt der obere Teil der Ampel komplett. Zudem ist die Disparität des Abschnittes zwischen der Ampel und dem Auto daneben im Vergleich zur Umgebung zu groß.



(a) Links | 000005_10 (b) Links | 000016_10

(c) Rechts | 000005_10 (d) Rechts | 000016_10

(e) Disparitätsschätzung | 000005_10 (f) Disparitätsschätzung | 000016_10

(g) Fehlerkarte | 000005_10 (h) Fehlerkarte | 000016_10

Abbildung 7.3: Ergebnis des KITTI_RAIN-Netzes für zwei Szenen des KITTI15-Datensatzes mit Regentropfen.

In der Fehlerkarte, die den D1-Fehler zeigt, ist diese Stelle ein besonders großer Fehler. Der obere Teil der Ampel ist leider nicht durch die Groundtruth-Disparitätskarte abgedeckt und kann somit nicht in der Bewertung miteinbezogen werden. Auch der Himmel wird nicht im Fehler berücksichtigt, für den das Netz eine zu große Disparität bestimmt hat. Die Disparität für den Himmel müsste grundsätzlich kleiner als 1 Pixel sein.
Die auf der linken Seite der Abbildung dargestellten Szene *000005_10*, ist der weiße Transporter im rechten Bild (Abb. 7.3d) stark von Tropfen verdeckt. In der geschätzten Disparitätskarte ist dieser aber vollständig sichtbar. Die zugehörige Fehlerkarte zeigt, dass die Disparitäten trotzdem noch von den Groundtruth-Werten abweichen.

Auswertung für die PAD-Testdaten

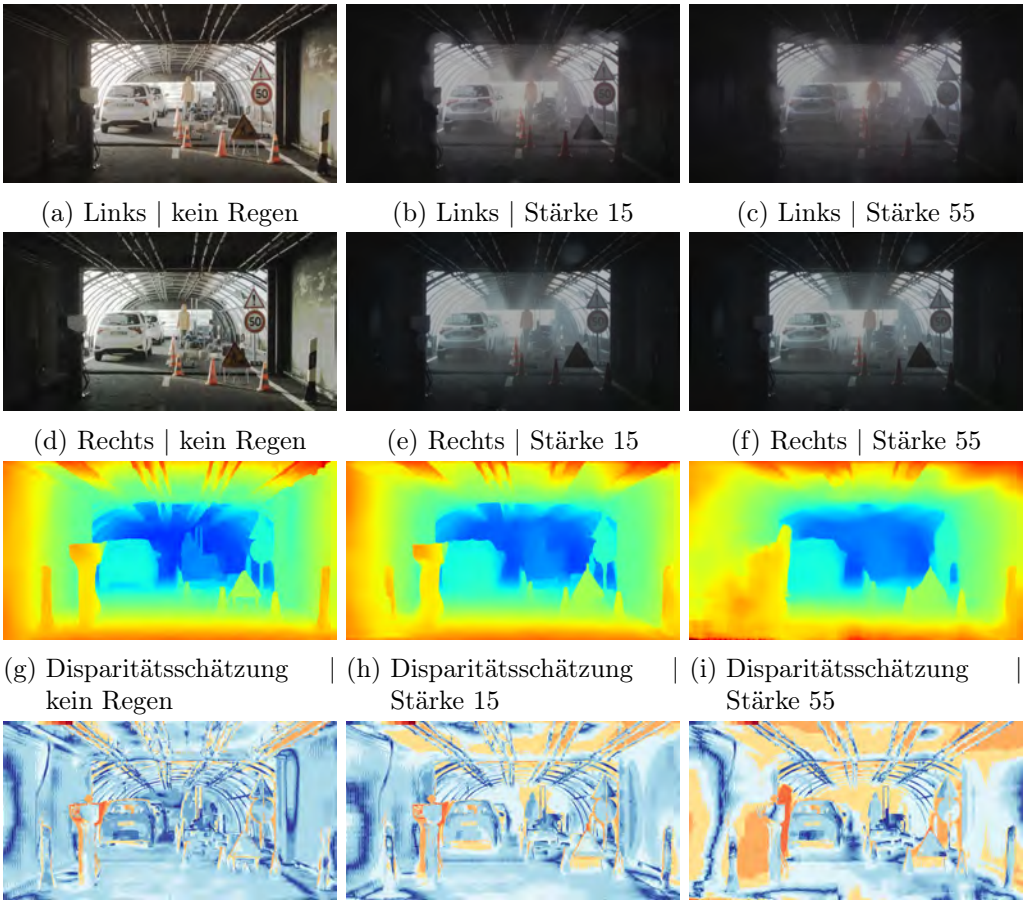
Zusätzlich zu dem augmentierten KITTI15-Datensatz wurde der PixelAccurateDepth-Datensatz für eine Auswertung mit Daten mit realistischem Regen verwendet. Dafür wurden die Aufnahmen der Szene 3 bei Tag mit Regen und ohne genutzt. Die Aufnahmen mit Regen umfassen je 10 Aufnahmen für eine leichte und eine starke Regenstärke. Die Ergebnisse dafür sind in Tabelle 7.2 zu sehen.

Die Ergebnisse für die Testdaten ohne Regen (PAD_CLEAN_S3) zeigen, dass das Basisnetzwerk leicht bessere Schätzungen liefert als das KITTI_RAIN-Netzwerk. Das KITTI_RAIN-Netzwerk hat besonders Schwierigkeiten mit der hellen Außenwand neben dem weißen Auto (s. Abb. 7.4a und b), welches den Großteil des Unterschiedes im Fehler ausmacht. Das Basisnetz hat mit dieser Stelle auch Schwierigkeiten, wie in Abbildung 7.4g zu sehen ist, doch ist die fehlerhafte Fläche kleiner.

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
PAD_CLEAN_S3	KITTI_BASIS	**1,440**	**7,765**	**53,554**
	KITTI_RAIN	*1,580*	*8,806*	*55,246*
PAD_RAIN_S3_15	KITTI_BASIS	**1,943**	**16,402**	**69,841**
	KITTI_RAIN	*2,348*	*21,348*	*73,430*
PAD_RAIN_S3_55	KITTI_BASIS	**2,927**	**30,417**	**75,662**
	KITTI_RAIN	*3,213*	*34,660*	*77,755*

Tabelle 7.2: Ergebnisse der KITTI-Netzwerke für PAD_CLEAN_S3 und PAD_RAIN_-S3.

Bei der Regenstärke von 15 mm/h/m^2 (PAD_RAIN_S3_15) erzielt das Basisnetzwerk die besseren Ergebnisse. In der gezeigten Aufnahme aus den Testdaten (7.4b und 7.4e) ist zu sehen, dass die Kamera leicht von Tropfen bedeckt ist. Diese verursachen keine direkte Verdeckung, verändern aber durch die Brechung des Lichts stellenweise die Helligkeit der Szene. Zudem verdunkelt der fallende Regen die Szene stark.
Die geschätzten Disparitätskarten des Basisnetzwerks sind im Vergleich zu der Szene ohne Regen weniger detailliert und Kanten von Objekten sind undeutlicher. Zudem sind die Disparitäten für die Wand im hinteren Bereich der Kammer und Bereiche der Decke fehlerhafter, wie in der Disparitätskarte in Abbildung 7.4h und der zugehörigen Fehlerkarte 7.4k zu erkennen ist.
Das KITTI_RAIN-Netz erzeugt leicht schlechter Ergebnisse bei den die bereits genannte Probleme mit der Außenwand durch den Regen noch verstärkt ausgeprägt sind.



(a) Links | kein Regen (b) Links | Stärke 15 (c) Links | Stärke 55

(d) Rechts | kein Regen (e) Rechts | Stärke 15 (f) Rechts | Stärke 55

(g) Disparitätsschätzung | kein Regen (h) Disparitätsschätzung | Stärke 15 (i) Disparitätsschätzung | Stärke 55

(j) Fehlerkarte | kein Regen (k) Fehlerkarte | Stärke 15 (l) Fehlerkarte | Stärke 55

Abbildung 7.4: Ergebnis des KITTI_Basisnetzes für Szene 3 des PAD-Datensatzes ohne Regen und mit Regen der Stärke 15 und 55 mm/h/m^2.

Für die Aufnahmen mit Regenstärke 55 mm/h/m^2 wird der beschriebene Effekt des Regens verstärkt. Mehr Tropfen sorgen für mehr Veränderung der Helligkeit und Sichtbarkeit einiger Objekte. Dies resultiert in großer Unsicherheit in den Disparitätskarten, wie beispielsweise in der Disparitätskarte 7.4i zu erkennen ist. Die Umrisse von Objekten wie von dem Verkehrsschild sind nicht mehr deutlich zu erkennen und die Stelle zwischen dem Leitpfosten und der Säule vorne links wird nicht als Teil der Wand erkannt. Auch die Decke und die hintere Wand der Kammer besitzen größere Abweichungen der Disparitäten.
Da das Szenario der Testdaten nicht nur Regentropfen vor der Kamera, sondern auch aktiven Regenfall beinhaltet, der die Helligkeit der Szene stark verringert, lässt sich durch das trainierte Netzwerk keine Verbesserung beobachten.

Gesamtauswertung der KITTI-Regenropfen-Netzwerke

Das KITTI_RAIN-Netzwerk erbringt für den KITTI-Datensatz mit Regentropfen deutlich bessere Ergebnisse als das Basisnetzwerk. Dies zeigt, dass das Netzwerk darauf trainiert werden konnte, die Tropfen zu ignorieren und die korrekten Disparitäten zu bestimmen. Die Ergebnisse für die PAD-Testdaten zeigen, dass dies aber nicht so einfach auf realistische Szenarien übertragbar ist. Hierbei ist jedoch das größte Problem, dass in diesen Fällen nicht nur Regentropfen die Disparitätsschätzung erschweren, sondern auch fallende Regentropfen. Dieser Regenfall führt zu Veränderung der Lichtverhältnisse und verschlechtert die Sichtbarkeit für Objekte mit größerer Entfernung.
Für die PAD-Testdaten hat das Basisnetz insgesamt die bessere Leistung gezeigt. In Szenarien mit leichtem Regenfall kann das Netz je nach Anwendungsfall ausreichende gute Disparitätskarten liefern. Für stärkeren Regen sind die Ergebnisse jedoch nicht verlässlich genug.
Sollte es Anwendungsfälle geben, bei denen ausschließlich Tropfen die Sicht auf die Szene versperren, könnte ein Netzwerk wie das KITTI_RAIN eingesetzt werden.

7.3.2 ZED-Netzwerke und Stereokameras

Für die Evaluierung der Regentropfennetzwerke, die auf den jeweiligen ZED-Datensätzen trainiert wurden, wurde das aufgenommene Szenario 1 (s. Abschnitt 5.2.2) mit Regentropfen (*ZED_SCENA_RAIN*), das Szenario ohne Störeffekte (*ZED_SCENA_CLEAN*) und der Testdatensatz der aufgenommenen ZED-Sequenzen mit Regentropfen (*ZED_SEQ_RAIN*) verwendet. Auch die Ergebnisse der Stereokameras für diese Szenarien werden betrachtet. Für *ZED_SEQ_RAIN* existieren keine Ergebnisse für die Luxonis OAK-D Pro, da dieser Datensatz nur mit der Stereolabs ZED 2i aufgenommen wurde.

Auswertung für das Szenario ohne Störeffekt

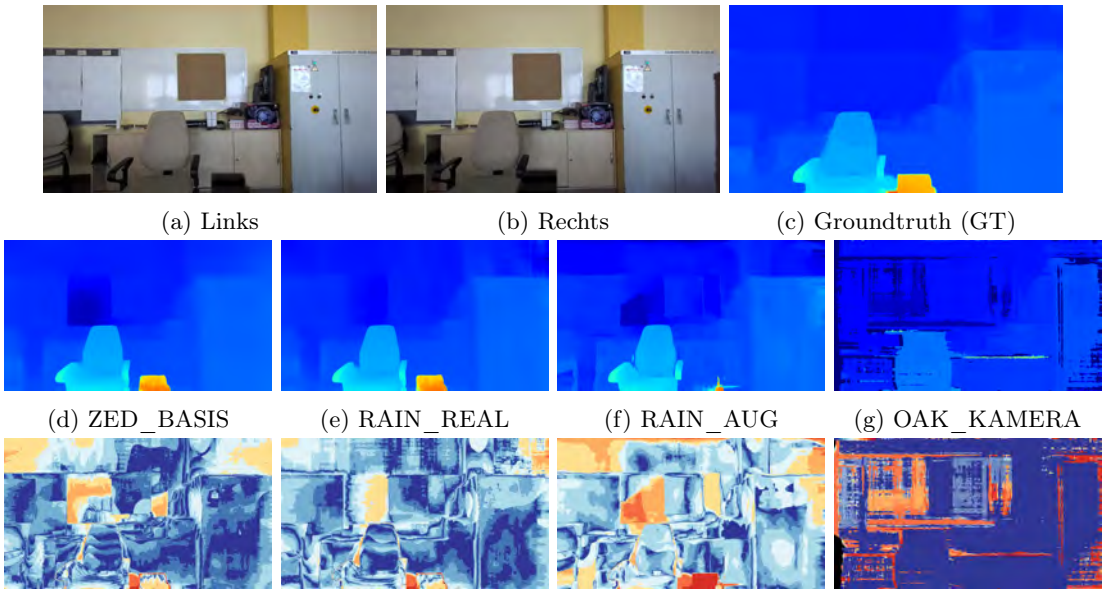
Die Ergebnisse für das Szenario ohne Störeffekte sind in Tabelle 7.3 dargestellt.

Für ZED_SCENA_CLEAN erreicht das ZED_RAIN_REAL-Netzwerk mit einem EPE von 0,939 % und D1-Fehler von 4,836 % die besten Ergebnisse. Das Basisnetzwerk liegt mit einem EPE von 1,256 % und einem D1-Fehler von 8,195 % an zweiter Stelle. In Abbildung 7.5 sind einige Ergebnisse für die Szene ohne Störeffekte dargestellt. Es werden zudem das linke und rechte Bild als auch die Groundtruth-Disparitätskarte gezeigt.

Testdaten	Netzwerk-Name	EPE (px)	D1-all(%)	> 1 px (%)
ZED_SCENA_CLEAN	ZED_BASIS	1,256	8,195	28,740
	ZED_RAIN_REAL	**0,939**	**4,836**	25,287
	ZED_RAIN_AUG	2,499	18,468	*57,776*
	ZED_KAMERA	1,720	7,229	**14,781**
	OAK_KAMERA	*6,213*	*30,204*	34,106

Tabelle 7.3: Ergebnisse der ZED-Regentropfennetzwerke für das ZED_CLEAN Szenario.

Bei Betrachtung der Fehlerkarten in Abbildung 7.5e zeigt sich, dass der geringere Fehler des ZED_RAIN_REAL-Netzwerks besonders durch die besseren Disparitätsschätzungen für die Spiegelung im Whiteboard zustande kommt. Dies lässt sich wahrscheinlich durch die Spiegelungen auf der Plexiglasscheibe erklären, die für die Aufnahme der Regentropfen-Bilder genutzt wurde. Durch das Training mit diesen Daten hat es möglicherweise gelernt, die in der Spiegelung zu erkennenden Informationen zu ignorieren und den Kontextinformationen mehr einzubeziehen.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) RAIN_REAL (f) RAIN_AUG (g) OAK_KAMERA

(h) ZED_BASIS (i) RAIN_REAL (j) RAIN_AUG (k) OAK_KAMERA

Abbildung 7.5: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_CLEAN.

Den Metriken nach bringt das ZED_RAIN_AUG-Netzwerk die schlechtesten Ergebnisse der drei Netzwerke. Die Disparitätskarte (Abb. 7.5f) und die zugehörige Fehlerkarte (7.5j) zeigen, dass abgesehen von der Spiegelung ein großer Teil des Fehlers durch das Fehlen der

Strebe der Plexiglas-Halterung verursacht wird. Da die Verdeckung durch Regentropfen einen ähnlichen Effekt hat wie die Verdeckung durch diese Strebe, bestimmt das Netzwerk die Disparitäten für den Schrank, der dahinter liegt. Dies entspricht jedoch nicht der Groundtruth-Disparitätskarte und wird deshalb als Fehler angesehen. Im Allgemeinen sind die Disparitätskarten von ZED_RAIN_AUG zwar detaillierter, besitzt aber auch mehr Abweichungen.

Die Disparitätskarte der OAK-Kamera (Abb. 7.5g) ist von vielen „Streifen“ durchzogen, die in beide Richtungen starke Abweichungen der Disparitäten besitzen. Diese werden hauptsächlich durch die Spiegelung im Whiteboard und durch die obere Kante des Schrankes darunter verursacht. Die Disparitätskarte der ZED-Kamera enthält die gleichen Fehler wie die des Basisnetzwerkes nur mit größeren Abweichungen. Besonders stark ist sie für die Spiegelung im Whiteboard.

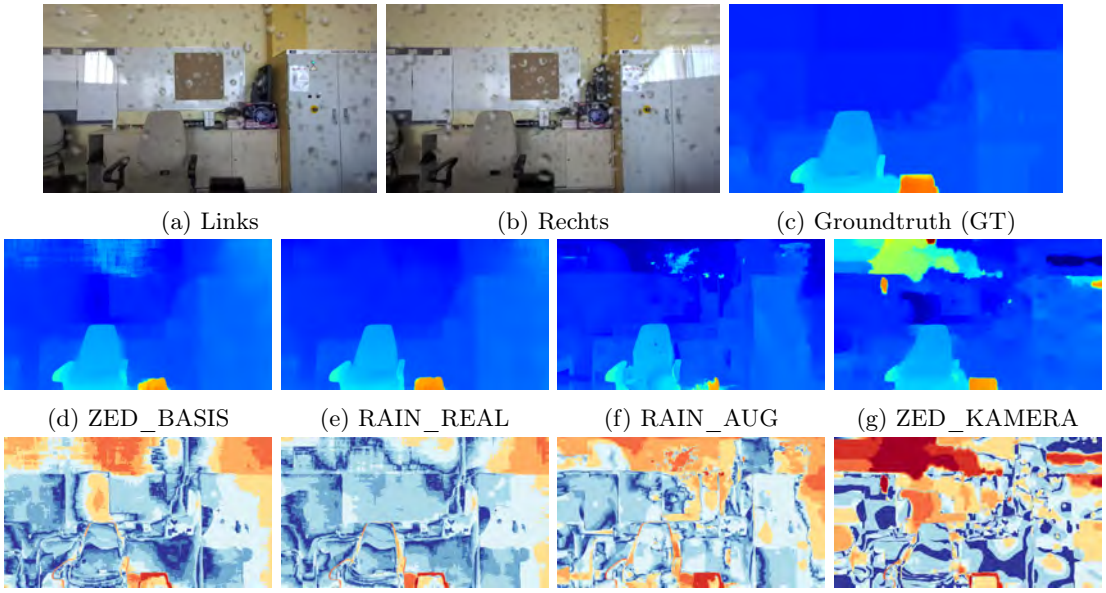
Auswertung für Szenario 1 - Regentropfen

Die Ergebnisse für das Regentropfen-Szenario in Tabelle 7.4 zeigen, dass das ZED_RAIN_REAL-Netzwerk dafür im Durchschnitt die besten Disparitätsschätzungen liefert. In Abbildung 7.6 sind die Ergebnisse einer Aufnahme des Regenszenarios mit der

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
ZED_SCENA_RAIN	ZED_BASIS	2,238	16,898	55,528
	ZED_RAIN_REAL	**1,734**	**10,471**	**49,052**
	ZED_RAIN_AUG	3,481	26,318	*70,682*
	ZED_KAMERA	8,060	27,487	49,793
	OAK_KAMERA	*16,623*	*62,150*	65,900

Tabelle 7.4: Ergebnisse der ZED-Regentropfennetzwerke für das ZED_RAIN Szenario.

Plexiglasscheibe in Steckplatz 5 dargestellt (eine weitere Szene in Abb. A.6). In der Disparitätskarte vom Basisnetzwerk für die Szene (Abb. 7.6d) und der Fehlerkarte (Abb. 7.6h) ist zu erkennen, dass die Regentropfen zu einer größeren fehlerhaften Stelle für die Wand über dem Whiteboard führt. Der Vergleich der Fehlerkarte mit der Fehlerkarte für die Szene ohne Störeffekte in Abbildung 7.5h zeigt, dass im Allgemeinen eine höhere Abweichung der Disparitätswerte besteht. Die fehlerhaften Stellen oben links und oben rechts sind wahrscheinlich Folgen der Spiegelungen in der Plexiglasscheibe und nicht direkt durch die Regentropfen verursacht.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) RAIN_REAL (f) RAIN_AUG (g) ZED_KAMERA

(h) ZED_BASIS (i) RAIN_REAL (j) RAIN_AUG (k) ZED_KAMERA

Abbildung 7.6: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_-RAIN Steckplatz 5.

Das ZED_RAIN_REAL-Netzwerk liefert für diese Szene die besten Ergebnisse. Die Disparitäts- und Fehlerkarte zeigen, dass keine großen fehlerhaften Stellen gibt, die von den Regentropfen verursacht sind. Im Vergleich zum Ergebnis für die Szene ohne Störeffekte ist die Genauigkeit im Durchschnitt gesunken.
Die Disparitätskarte vom ZED_RAIN_Aug-Netz weist einige Flecken mit zu großer oder zu kleiner Disparität auf, die wahrscheinlich aus falschen Korrespondenzen zwischen Regentropfen entstanden sind. Wie auch schon für die vorherige Szene ohne Regentropfen wurden die Disparitäten für die Strebe der Plexiglas-Halterung nicht vollständig bestimmt.
Die ZED-Kamera hat hierbei große Schwierigkeiten mit der Spiegelung oben links. Die Disparitätskarte enthält zudem viele kleine Flecken, wo die Tropfen zu Abweichungen der Disparitäten geführt haben. Die Disparitätskarten der OAK-Kamera enthalten noch mehr und stärkere Abweichungen, die das zuverlässige Erkennen der Szene verhindern.

Auswertung für die Testdaten der ZED-Sequenz mit Regentropfen

Die Ergebnisse für die Testdaten der ZED-Sequenz mit Regentropfen ist in Tabelle 7.5 dargestellt. In Abbildung 7.7 ist eine Szene der Testdaten mit einigen Disparitätsschätzungen und Fehlerkarten zu sehen.

Für die Testdaten vom ZED_SEQ_RAIN-Datensatz verbessern sich die Ergebnisse zu der vorherigen Test-Szene. Das ZED_RAIN_REAL-Netzwerk erbringt auch hier die besten Ergebnisse. Vereinzelt zeigen sich Fehler bei größeren gleichmäßigen Flächen wie einer weißen Wand oder große Tropfen, die durch das Licht einer Lampe stark erhellt wurden.

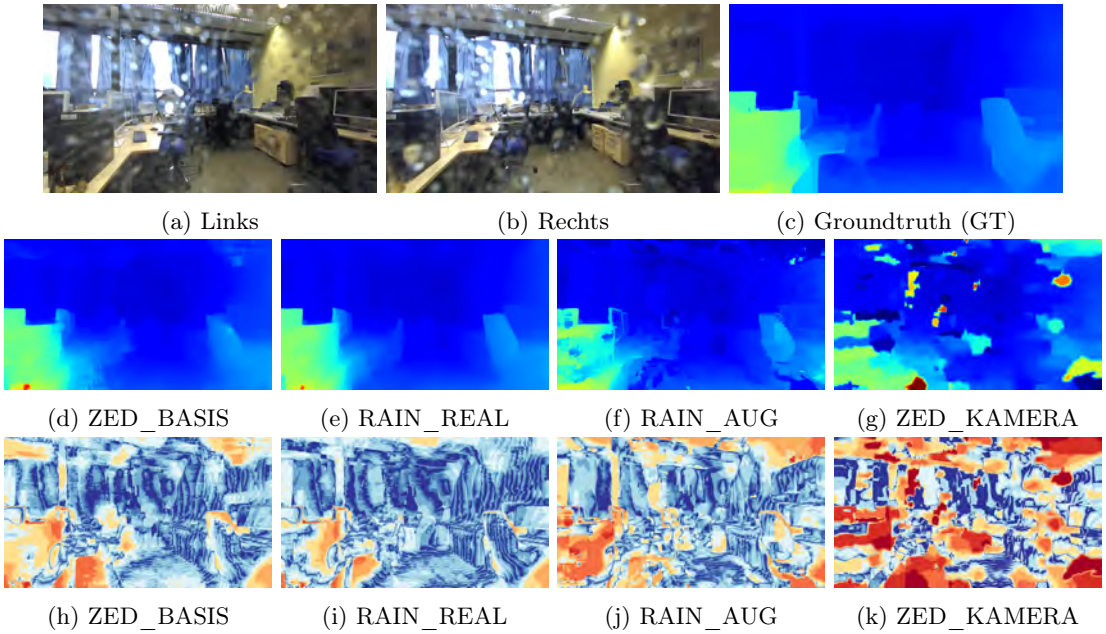
Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
ZED_SEQ_RAIN	ZED_BASIS	1,924	14,888	38,297
	ZED_RAIN_REAL	**1,693**	**12,880**	**32,987**
	ZED_RAIN_AUG	2,692	21,913	*47,713*
	ZED_KAMERA	*7,257*	*29,587*	45,122
	OAK_KAMERA	–	–	–

Tabelle 7.5: Ergebnisse der ZED-Regentropfennetzwerke für die ZED_SEQ_RAIN-Testdaten.

Das Basisnetzwerk erbringt im Vergleich schlechtere Ergebnisse mit teilweise größeren Abweichungen für dieselben Problemstellen. Trotzdem sind die Disparitätskarten im Allgemeinen noch ausreichend genau. Die Disparitätskarten des ZED_RAIN_AUG-Netzes enthalten dieselben fehlerhaften Stellen wie für ZED_SCENA_RAIN, mit grundsätzlich hohem Detailgrad, aber auch starken Abweichungen. Besonders Tropfen, die vom Licht einer Lampe stark betroffen sind, führen zu fehlerhaften Stellen mit großen Abweichungen. Bei der ZED-Kamera führen Szenen mit vielen Regentropfen an den Stellen zu vielen fehlerhaften Korrespondenzen. Starker Lichteinfall verstärkt diese Fehler noch mehr.

Gesamtauswertung der ZED-Regentropfennetzwerke

Die Ergebnisse für beide Testdatensätze zeigen, dass das Training mit realen Regentropfen zu deutlich besseren Ergebnissen führt als mit den künstlich erzeugten Tropfen. Das ZED_RAIN_REAL-Netz zeigt vielversprechende Leistung für Szenarien, bei denen ausschließlich Regentropfen ein Problem sind. Auch bei vielen Tropfen und ungünstigem Lichteinfall ist das Netz in der Lage, zuverlässige Disparitätsschätzungen zu erzeugen.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) RAIN_REAL (f) RAIN_AUG (g) ZED_KAMERA

(h) ZED_BASIS (i) RAIN_REAL (j) RAIN_AUG (k) ZED_KAMERA

Abbildung 7.7: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SEQ_RAIN Szene 26.

Im Vergleich zu den betrachteten Stereokameras zeigt sich, dass die Kameras für diese Szenarien nicht verwendbar sind.

7.4 Ergebnisse der Nebelnetzwerke

Die Auswertung der Nebelnetzwerke wird anhand von Testdaten von drei verschiedener Datensätze durchgeführt. Zuerst werden die Ergebnisse für die Testdaten des KITTI15-Datensatzes ohne und mit augmentiertem Nebel betrachtet. Für eine Evaluierung mit realistischem Nebel wird zudem der PixelAccurateDepth-Datensatz verwendet. Dieser bietet hochauflösende Groundtruth-Disparitätskarten, wodurch eine genaue Fehlerbestimmung möglich ist. Des Weiteren werden Testdaten des SeeingThroughFog-Datensatzes verwendet, welcher die besten und realistischsten Nebelbilder bietet. Da für die verfügbaren Groundtruth-Daten bei Nebel nicht nutzbar sind, wird zur Bewertung die FSIM-Metrik verwendet.

7.4.1 Auswertung für die KITTI-Testdaten

KITTI-Testdaten ohne Nebel

Für die Testdaten des KITTI15-Datensatzes ohne Nebel erzielte das Basisnetzwerk die besten Ergebnisse, wie in Tabelle 7.6 zu sehen ist. Das Netz liefert gute Disparitätskarten mit vielen Details und besonders für Straßen und Hauswände genaue Schätzungen. Leichte Abweichungen existieren für die Oberflächen von Fahrzeugen und größere besonders für die Konturen von Fahrzeugen bzw. dem Hintergrund drumherum.

Die Ergebnisse des KITTI_FOG-Netzes enthalten im Durchschnitt mehr Abweichungen als das Basisnetz, was sich aus dem D1- und 1-Pixel-Fehler ablesen lässt. Bei Betrachtung der Disparitäts- und Fehlerkarten fallen aber viele Fehler im oberen Bildbereich auf, für die keine Groundtruth-Daten existieren und somit nicht in die Bewertung mit einfließen. Besonders für größere Bereiche, wo der Himmel zu sehen ist, werden oft zu große Disparitäten bestimmt, die eigentlich kleiner als 1 px sein müssten. Auch sind Objekte, wie z. B. Straßenschilder oder Ampeln, bei denen im Hintergrund der Himmel zu sehen ist, oftmals oberhalb der Groundtruth-Daten nicht vollständig erkennbar.

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
KITTI_15_NORMAL	KITTI_BASIS	**0,807**	**2,060**	**17,247**
	KITTI_FOG	0,947	3,149	19,935
	VKITTI_FOG	2,201	9,668	32,692
	DS_FOG	*3,103*	*10,750*	*34,942*
KITTI_15_FOG	KITTI_BASIS	1,515	10,226	28,431
	KITTI_FOG	**1,009**	**4,722**	**23,361**
	VKITTI_FOG	*2,160*	*13,774*	*37,202*
	DS_FOG	1,806	11,762	35,923

Tabelle 7.6: Ergebnisse der Nebelnetzwerke für die KITTI15-Testdaten

Das VKITTI_FOG-Netz ist grundsätzlich in der Lage, die Disparitäten für die groben Strukturen der Szene zu bestimmen. Die erkennbaren Probleme, die den größeren Fehlern führen, sind besonders die Konturen von Fahrzeugen, die spiegelnden Oberflächen besitzen oder vor einfarbigen Hintergründen zu sehen sind. Auch größere über- oder unterbelichtete Flächen stellen eine große Herausforderung dar.

Das DS_FOG-Netzwerk hat ebenso große Schwierigkeiten mit extremen Helligkeitsunterschieden. Besonders auffällig ist dies bei Stellen, die im Schatten liegen und dadurch zu einer gleichfarbigen Fläche werden. Büschen, dunkle Autofenstern oder auch Stücke von

Stangen von Verkehrsschildern wird oft eine viel zu geringe Disparität zugewiesen. Für VKITTI_FOG und DS_FOG lässt sich diese Leistung mit den Unterschieden der Domänen der Datensätze erklären. Der künstliche VKITTI-Datensatz enthält nicht nur keine realistischen Texturen, sondern auch keine realistische Beleuchtung. Der DrivingStereo-Datensatz besteht zwar aus realen Aufnahmen, die sich aber von Farbsättigung und besonders der Helligkeit von den KITTI-Datensätzen unterscheidet. Es existieren beispielsweise keine dunklen Schatten, durch die Objekte sich kaum vom Hintergrund abheben. Diese Unterschiede führen dazu, dass die Netze nicht für solche Situationen trainiert wurden und deshalb für die Testdaten schlechte Ergebnisse erbringen.

KITTI-Testdaten mit Nebel

Für die Testdaten vom KITTI15-Datensatz mit künstlichem Nebel erreichte das KITTI_FOG-Netzwerk den Metriken nach die besten Ergebnisse. Im Durchschnitt enthalten die Disparitätskarten nur geringe Abweichungen. Bei dichterem Nebel bzw. größeren Ausschnitten, wo nur Nebel zu sehen ist, wie in der Szene in Abbildung 7.8 kommt es stellenweise zu größeren Abweichungen (weitere Beispiel in Abb. A.5).

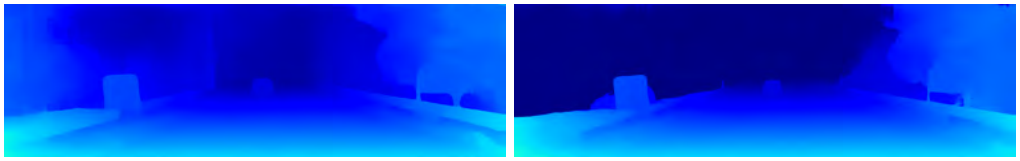
Bei Betrachtung der Disparitätskarte (7.8c) ist zu erkennen, dass auf Höhe der mittleren Leitplanken eine Fläche aus zu hohen Disparitäten bestimmt wurde. Die Disparitäten entsprechen den Werten, die dem Auto hinter dem Schild zugewiesen wurden, wodurch das Auto nicht vom Hintergrund zu unterscheiden ist. Gerade aus in der Bildmitte werden die Disparitäten immer geringer.

Das KITTI_Basisnetz hat bei diesen Daten große Schwierigkeiten, Objekte zu erkennen, die von Nebel umgeben sind. Zudem hat es wie das KITTI_FOG-Netz das Problem, für Bildausschnitte mit viel Nebel großflächig zu hohe Disparitäten zu bestimmen. Das DS_FOG-Netz hat dieselben Schwierigkeiten. Der größere durchschnittliche Fehler ist auf vereinzelte Fehler für Bildausschnitte mit extremen Schatten oder Überbelichtung zurückzuführen, wie schon bei der Auswertung für die KITTI_RAIN-Ergebnisse erläutert wurde.
Den Metriken nach erbringt VKITTI_FOG die schlechtesten Ergebnisse. Die Disparitäts- und Fehlerkarten zeigen, dass die Fehlerursache aber anders ist als bei den vorherigen Netzen. In Abbildung 7.8f und 7.8g sind die Disparitäts- und Fehlerkarte des VKITTI_FOG-Netzes für die bereits betrachtete Szene dargestellt. Im Vergleich zum KITTI_FOG-Netz hat es für fast den gesamten Bildbereich, wo nur Nebel zu sehen ist,

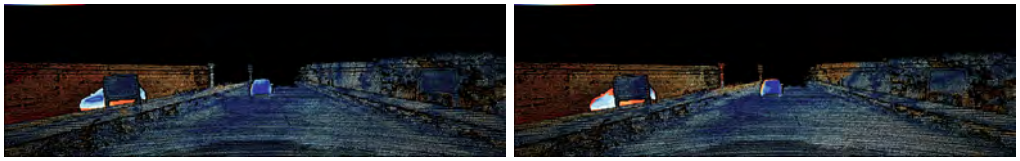
Disparitäten kleiner als 1 zugewiesen. Die vordere Straßenlaterne ist dafür gar nicht zu erkennen. Das Auto hinter dem Schild wird aber noch zum Teil erkannt.

(a) Links ohne Nebel | 000179_10 (b) Rechts ohne Nebel | 000179_10

(c) Rechts mit Nebel | 000179_10 (d) Rechts mit Nebel | 000179_10



(e) Disparitätsschätzung | KITTI_FOG (f) Disparitätsschätzung | VKITTI_FOG



(g) Fehlerkarte | KITTI_FOG (h) Fehlerkarte | VKITTI_FOG

Abbildung 7.8: Ergebnis des KITTI_FOG und KITTI_FOG-Netzes für die Szene 000179_10 des KITTI15-Datensatzes mit Nebel.

7.4.2 Auswertung für die PAD-Testdaten

Für die Evaluierung auf echten Nebeldaten wird der PixelAccurateDepth-Datensatz mit und ohne Nebel verwendet. Es existieren Aufnahmen mit unterschiedlicher Nebeldichte, die anhand der meteorologischen Sichtweite (SW) kategorisiert wurde (s. Abschnitt 5.1.5). Für die Auswertung wird die höchste Dichte mit einer Sichtweite von 20 m und einer leichteren Nebeldichte mit einer Sichtweite von 40 m verwendet. Hierbei werden nur Aufnahmen der Szene 3 mit Tageslicht verwendet. In Abbildung 7.10 ist die Szene ohne Nebel (a, d), mit Nebel bei Sichtweite 40 m (b, e) und mit Nebel bei einer Sichtweite von 40 m (c, f) dargestellt.

PAD-Testdaten ohne Nebel

Testdaten	Netzwerk-Name	EPE (px)	D1-all(%)	> 1 px (%)
PAD_CLEAN	KITTI_BASIS	**1,440**	**7,765**	53,554
	KITTI_FOG	1,626	11,547	57,296
	VKITTI_FOG	1,717	14,437	**53,497**
	DS_FOG	*3,232*	*23,432*	*60,567*

Tabelle 7.7: Ergebnisse der KITTI-Nebelnetzwerke für die PAD-Testdaten

Als Referenz wurde zuerst die Auswertung mit den Aufnahmen ohne Nebel durchgeführt. Die Ergebnisse hierfür sind in Tabelle 7.9 zu sehen. Das Basisnetzwerk erreicht hierbei die besten Ergebnisse. Die Disparitätskarte (Abb. 7.9g) besitzt einen hohen Detailgrad, der bei Betrachtung der Stangen an der Decke und den Objekten im hinteren Teil der Kammer feststellbar ist. Die zugehörige Fehlerkarte zeigt, dass die größten Abweichungen an Stellen mit schwacher und starker Beleuchtung sind. Die Säule vorne links hebt sich aufgrund der geringen Beleuchtung nur wenig vom Hintergrund ab, was dem Netzwerk eine genaue Bestimmung der Disparitäten erschwert. Das gleiche Problem zeigt sich für die Außenwand der Kammer links vom vorderen Auto. In diesem Fall führt die starke Helligkeit zu Spiegelungen an dem weißen Auto, wodurch die Kontur nicht eindeutig zu erkennen ist. Im Durchschnitt sind die geschätzten Disparitäten aber sehr nahe an den Groundtruth-Werten.

Das KITTI_FOG-Netzwerk hat dieselben Probleme wie das Basisnetzwerk, nur mit größeren Abweichungen. Die Ergebnisse des VKITTI_FOG-Netzwerks sind vom EPE-Fehler nicht viel schlechter als das KITIT_FOG-Netz, besitzt dafür aber größere Abweichungen, wie der D1-Fehler zeigt. Die Fehlerkarte der betrachteten Aufnahme in Abbildung 7.10j zeigt, dass dies hauptsächlich durch eine kleine Stelle neben dem Leitpfosten vorne rechts verursacht wird. Auch die Wand um die Säule vorne links zeigt mittelgroße Abweichungen der Disparitäten. Der weiße Teil der linken Seite des dreieckigen Gefahrenstellen-Verkehrsschildes wurde mit dem Hintergrund verwechselt. Dieser Stelle wurde die Disparitäten der Wand dahinter zugewiesen, was in der Disparitätskarte in Abbildung 7.10g zu erkennen ist. Gleichzeitig gibt es aber auch viele für die die Abweichung nur sehr gering sind (blaue Stellen).
Das DS_FOG-Netz zeigt dieselben Probleme, die schon für die KITTI-Datensätze erläutert wurden. Für eine große Fläche der vorderen linken Wand, die nur geringe beleuchtet wird, bestimmt das Netz im Vergleich zur Umgebung viel zu geringe Disparitäten. Auch

die Stellen zwischen den Stangen an der Decke besitzen größere Abweichungen der Disparitäten.

PAD-Testdaten mit Nebel - Sichtweite 40 m

Die Ergebnisse für die Szene mit Nebel mit der Sichtweite von 40 m sind in Tabelle 7.8 zu sehen. Durch den Nebel sind Details der weniger deutliche sichtbar und die Beleuchtung der Szenen verändert. Hierbei liefert das Basisnetzwerk wieder die besten Ergebnisse.

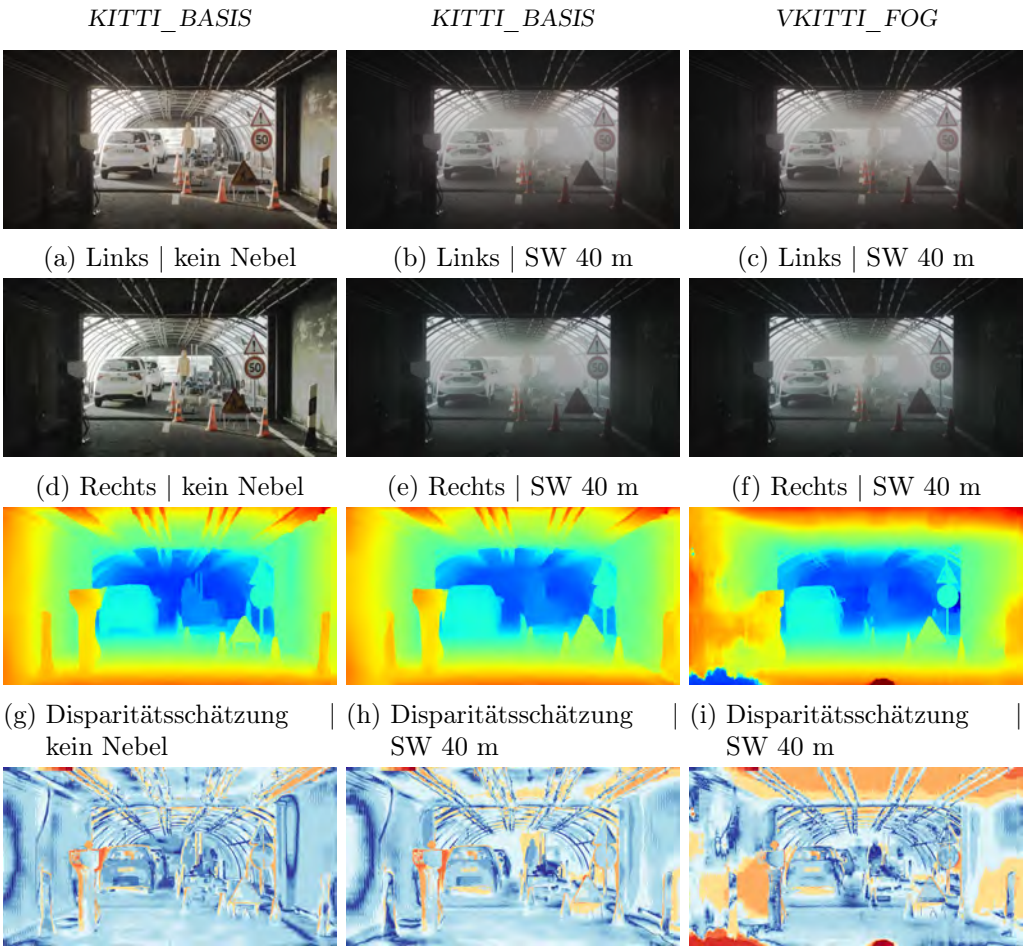
Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
PAD_FOG 40 m	KITTI_BASIS	**1,709**	**11,790**	**62,577**
	KITTI_FOG	2,036	18,294	63,298
	VKITTI_FOG	2,794	25,279	66,417
	DS_FOG	*4,199*	*31,714*	*67,458*

Tabelle 7.8: Ergebnisse der KITTI-Nebelnetzwerke für die PAD-Testdaten

Im Vergleich zur Szene ohne Nebel verschlechtert sich die Genauigkeit geschätzten Disparitätskarte. Die Schätzung für eine der Aufnahmen ist in Abbildung 7.9h dargestellt. Diese ist im Allgemeinen etwas unschärfer als zuvor, was besonders bei den Objekten im hinteren Bereich der Kammer zu erkennen ist. Die Fehlerkarte (Abb. 7.9k) zeigt, dass die Abweichungen für die hintere Wand und auch die Problemstelle links neben dem vorderen Auto größer geworden sind.

Die Ergebnisse des KITTI_FOG-Netzes zeigen eine stärkere Verschlechterung der Genauigkeit. Genau wie bei dem Basisnetz sind Konturen von Objekten in der erzeugte Disparitätskarte nicht mehr so klar zu erkennen. Besonders starke Abweichung befinden sich an der Decke, die durch den Nebel schlechter beleuchtet ist als zuvor. Interessanterweise hat das KITTI_FOG-Netz bei dieser Nebelstärke keine starken Abweichungen für die schwierige Stelle links neben dem vorderen Auto.

Das VKITTI_FOG-Netz hat große Probleme mit dem Boden und der Wand vorne links, wie in Abbildung 7.9i und 7.9l zu erkennen ist. Auch für die Decke sind die Disparitäten stark abweichen. Für den hinteren Bereich der Kammer sind die Disparitäten dagegen mit wenig Abweichung bestimmt worden. Besonders die Schaufensterpuppe ist klar vom Hintergrund getrennt erkannt worden und ist deutlich in der Disparitätskarte zu erkennen. Für das DS_FOG-Netz hat der Nebel zu einem verstärken Fehler für die Wand vorne links vor dem Leitpfosten geführt, die auch ohne Nebel ein Problem war. Für den hinteren Bereich ist die Genauigkeit der Schätzung ähnlich wie bei KITTI_FOG.



(a) Links | kein Nebel (b) Links | SW 40 m (c) Links | SW 40 m

(d) Rechts | kein Nebel (e) Rechts | SW 40 m (f) Rechts | SW 40 m

(g) Disparitätsschätzung | kein Nebel (h) Disparitätsschätzung | SW 40 m (i) Disparitätsschätzung | SW 40 m

(j) Fehlerkarte | kein Nebel (k) Fehlerkarte | SW 40 m (l) Fehlerkarte | SW 40 m

Abbildung 7.9: Ergebnisse des KITTI-Basisnetzes (1. und 2. Spalte) und VKITTI_FOG-Netzes (3. Spalte) für die Szene 3 des PAD-Datensatzes ohne Nebel und mit Sichtweite 40 m.

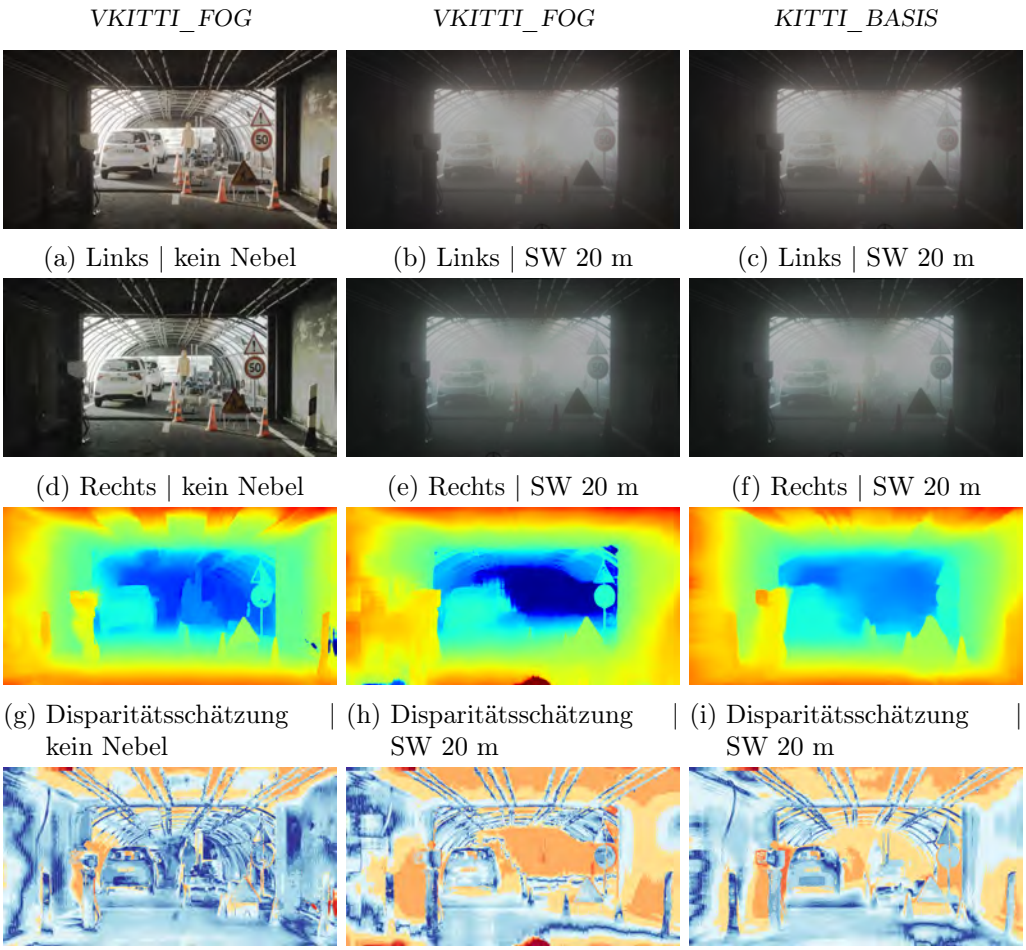
PAD-Testdaten mit Nebel - Sichtweite 20 m

Testdaten	Netzwerk-Name	EPE (px)	D1-all(%)	> 1 px (%)
PAD_FOG 20 m	KITTI_BASIS	**3,023**	**36,875**	**77,358**
	KITTI_FOG	3,661	44,170	*81,421*
	VKITTI_FOG	4,080	*49,504*	80,581
	DS_FOG	*5,462*	48,881	76,823

Tabelle 7.9: Ergebnisse der KITTI-Nebelnetzwerke für die PAD-Testdaten

Für die Aufnahmen der Szene mit dichterem Nebel mit einer Sichtweite von 20 m erbringt das Basisnetz die besten Ergebnisse. Der EPE-Fehler und besonders der D1-Fehler steigen im Vergleich zu den Ergebnissen bei einer Sichtweite von 40 m stark. Die Disparitäts- und Fehlerkarte in Abbildung 7.10i und 7.10l zeigen, dass dies durch stärkere Abweichungen im hinteren Bereich der Kammer verursacht wird. Die hintere Wand, das hintere Auto und die Schaufensterpuppe sind nicht mehr klar erkennbar. Auch dem Hintergrund um die Säule vorne links und um das Straßenschild rechts wird eine zu große Disparität zugewiesen. Dadurch erscheinen die Objekte größer, als sie es sind.

Das KITTI_FOG-Netz liefert ähnliche Ergebnisse wie das Basisnetz mit den gleichen Problemstellen, aber im Durchschnitt größeren Abweichungen. Das VKITTI_FOG-Netzwerk hat verstärkte Schwierigkeiten mit den wenig beleuchteten Stellen im vorderen Bereich der Kammer. Dadurch lassen sich in der Disparitätskarte die Objekte vor den Wänden nicht mehr vom Hintergrund unterscheiden, wie in Abbildung 7.10e zu sehen ist. Die Disparitätskarte zeigt zudem die Besonderheiten der vom VKITTI_FOG-Netz erzeugten Disparitätskarten. Im Gegensatz zu den anderen Netzwerken, die für den hinteren Bereich der Kammer eine höhere Disparität bestimmt haben, hat das VIKITI_FOG-Netz Disparitäten im Bereich 0 bis 2 zugewiesen. Hier wird der Nebel nicht als „vorgezogene“ Wand erkannt, wie es bei den anderen Netzen der Fall ist.
Das DS_FOG-Netzwerk hat wie zuvor große Schwierigkeiten mit der Wand vorne links. Die Fläche, für die stark abweichende Disparitäten bestimmt wurden, ist größer geworden und die Abweichung gestiegen. Die Genauigkeit für den hinteren Bereich der Kammer ist vergleichbar mit dem Basisnetzwerk.



(a) Links | kein Nebel (b) Links | SW 20 m (c) Links | SW 20 m

(d) Rechts | kein Nebel (e) Rechts | SW 20 m (f) Rechts | SW 20 m

(g) Disparitätsschätzung | kein Nebel (h) Disparitätsschätzung | SW 20 m (i) Disparitätsschätzung | SW 20 m

(j) Fehlerkarte | kein Nebel (k) Fehlerkarte | SW 20 m (l) Fehlerkarte | SW 20 m

Abbildung 7.10: Ergebnisse des VKITTI_FOG-Netzes (1. und 2. Spalte) und des KITTI-Basisnetzes (3. Spalte) für die Szene 3 des PAD-Datensatzes ohne Nebel und mit Sichtweite 20 m.

7.4.3 Auswertung für die STF-Testdaten

STF-Testdaten ohne Nebel

Nach der verwendeten FSIM-Metrik erreicht das DS_FOG-Netz für die Testdaten von STF_CLEAN mit durchschnittlich 86,541 % Ähnlichkeit die besten Ergebnisse. Ein wenig schlechter sind die Ergebnisse des VKITTI_FOG-Netzes mit 85,893 %. Mit ein wenig Abstand folgen dann das KITTI_FOG und KITTI_Basisnetz mit 84,139 % und 84,004 %.

Testdaten	**Netzwerk-Name**	**FSIM (%)**
STF_CLEAN	KITTI_BASIS	*84,004*
	KITTI_FOG	84,139
	VKITTI_FOG	85,893
	DS_FOG	**86,541**
STF_FOG	KITTI_BASIS	96,230
	KITTI_FOG	*96,088*
	VKITTI_FOG	**96,370**
	DS_FOG	96,140

Tabelle 7.10: Ergebnisse der KITTI-Nebelnetzwerke für die STF-Testdaten

Die erzeugten Disparitätskarten variieren je nach Szene stark in ihrer Qualität. Für „engere“ Szenen wie z. B. vor einer Hauswand oder in einer engeren Gasse sind die Disparitätskarten sehr klar und ohne große Abweichungen. Offenere Szenen wie beispielsweise auf breiten Straßen oder der Autobahn, besitzen größere Abweichungen und Fehlschätzungen (s. Abbildung A.7).

Ein Beispiel ist die Szene der 1. Spalte in Abbildung 7.11g. Der FSIM-Wert für die Rekonstruktion der Szene (7.11j) ist 69,377 %. Bei Betrachtung der Disparitätskarte (7.11g) mit der die Rekonstruktion erstellt wurde, sind größere Stellen mit hohen Disparitäten sichtbar, die nicht der Disparität der Umgebung entsprechen. Dies führt in der Rekonstruktion zu den „hellbraunen“ Stellen, wo die Disparität zu keiner gültigen Position im anderen Bild geführt hat. Dort konnte kein Pixelwert des linken Bildes übernommen werden. Zusätzlich zu diesen Stellen sind beispielsweise Häuserwände teilweise stark verzerrt. An diesen Stellen ist die Disparität etwas zu gering oder zu groß, sodass ein Pixel aus der Nachbarschaft des richtigen Pixels genommen wird. Dieser Pixel kann zufällig dieselbe Farbe besitzen wie das korrekte Pixel oder er besitzt eine andere Farbe wie im Falle eines Fensters anstatt der Wand daneben.

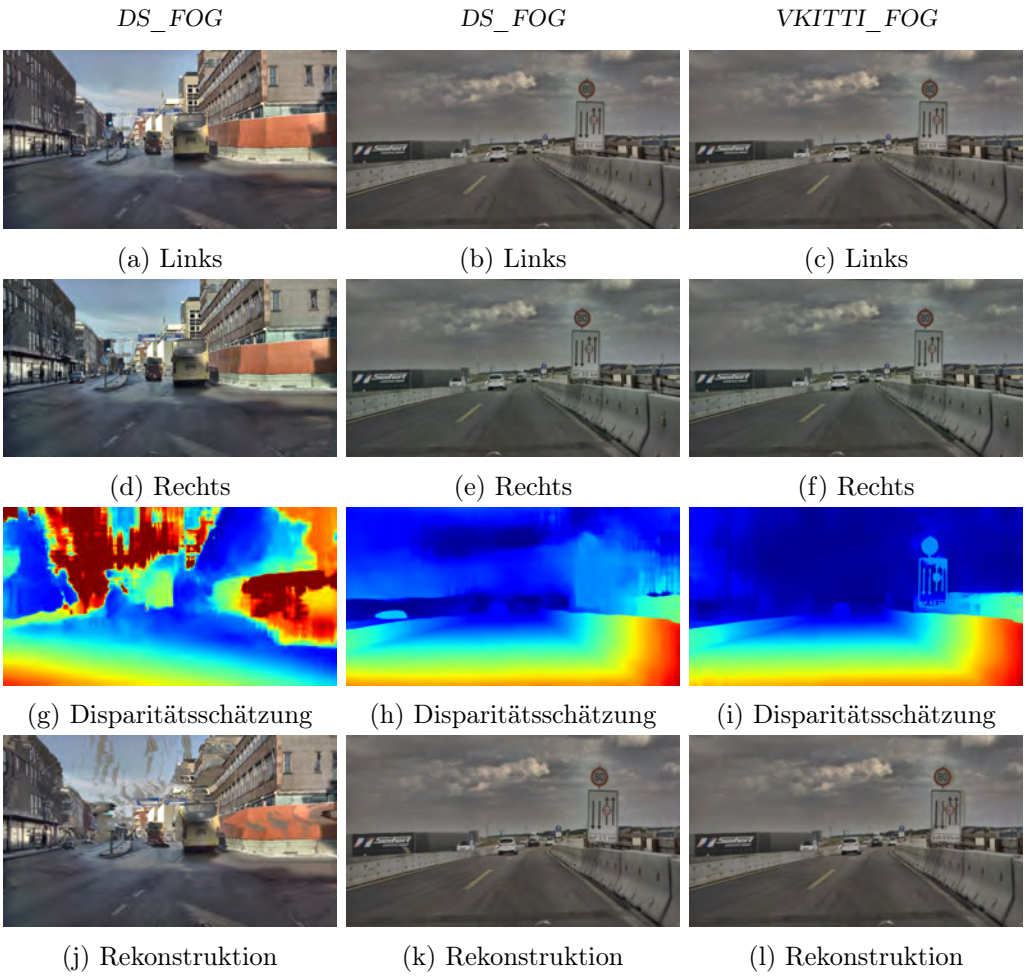


Abbildung 7.11: Ergebnisse des DS_FOG-Netzes (1. und 2. Spalte) und des VKITTI_-FOG (3. Spalte) für Szenen des STF-Datensatzes ohne Nebel.

In der zweiten Spalte der Abbildung 7.11 wird das Ergebnis des DS_FOG-Netzes für eine weitere Szene dargestellt. In der 3. Spalte ist für dieselbe Szene das Ergebnis des VKITTI_FOG-Netzwerkes zu sehen. Bei Betrachtung der Rekonstruktionen lassen sich kaum Fehler feststellen. Der größte Fehler ist bei der Rekonstruktion von VIKITI_FOG (Abb. 7.11l) für das Verkehrsschild auf der rechten Seite zu erkennen. Die FSIM-Werte 95,230 % für DS_FOG und 95,442 % unterstützen diese Beobachtung.
Die zugehörigen Disparitätskarten sind jedoch sehr unterschiedlich. Die untere Bildhälfte der Disparitätskarte von DS_FOG (Abb. 7.11h) ist ohne größere Fehler. Nur die kleine hellblaue Stelle vorne links lässt sich nicht wirklichem keinem Objekt zuordnen. In dieser Bildhälfte sind ausschließlich die Straße, die Fahrzeuge darauf und Mauern zu erkennen.

Die obere Bildhälfte enthält hauptsächlich den Himmel und an einer Stelle das Verkehrsschild. Der Himmel wird als eine Fläche mit relativ gleichmäßigen Disparitäten erkannt. Eine Ausnahme stellt die hellere Wolke in der Mitte dar. Das diese Disparitäten für den Himmel falsch sind, lässt allein durch den Vergleich mit den Disparitäten der Fahrzeuge weiter hinten in der Szene feststellen. Da diese näher zur Kamera sind als der Himmel, dürfen die Disparitäten für den Himmel nicht größere als die für die Fahrzeuge sein. Die Disparitäten sollten ähnlich zu den sein, die für die Wolke in der Mitte bestimmt wurden.

Die Disparitätskarte von VKITTI_FOG (Abb. 7.11i) ist für die obere Bildhälfte im Vergleich deutlich besser. Für die Bildmitte hat das Netz Disparitäten im Bereich 0 bis 2 bestimmt. Auch rechtes vom Verkehrsschild sind die Disparitäten zum Großteil sehr niedriger. Nur am linken Bildrand ist eine größere Fläche mit falschen Disparitäten, die dem vordersten Auto entsprechen. Des Weiteren hat das Netz Schwierigkeiten für die weiße Oberfläche des Verkehrsschildes korrekte Disparitäten zu bestimmen. Abgesehen von den Symbolen und dem Rand sind die Disparitäten viel zu gering.

STF-Testdaten mit Nebel

Für die STF-Testdaten mit Nebel erbringt der Metrik nach das VKITTI_FOG-Netzwerk mit 96,370 % im Durchschnitt die besten Ergebnisse. Das zweitbeste Netzwerk ist mit 96,230 % das Basisnetzwerk, gefolgt vom DS_FOG-Netz mit 96,140 % und zum Schluss das KITTI_FOG-Netz mit 96,088 %. Diese Werte liegen innerhalb von 0,282 %, weshalb man auf eine hohe Ähnlichkeit zwischen Ergebnisse der einzelnen Netzwerke schließen könnte. Bei Betrachtung der erzeugten Disparitätskarten ist jedoch zu erkennen, dass die hohe Wertung nicht der Qualität der Disparitätskarten entspricht.

In Abbildung 7.12 sind Ergebnisse von VKITTI_FOG und KITTI_BASIS dargestellt. In der 1. Spalte ist eine Szene mit einer Baustelle zu sehen, bei der in der oberen Bildhälfte auf der linken Seite mehrere Pfeiler und auf der rechten Seite einige Bäume zu sehen sind. Dadurch ist die Fläche, wo ausschließlich Himmel oder dichter Nebel zu erkennen ist, nicht sehr groß. Die erzeugte Disparitätskarte (Abb. 7.12g) ist sehr detailliert und zeigt sogar die Disparitäten für dünnere Stäbe und Leitungen. Für den Himmel in der Bildmitte wurden geringe Disparitäten bestimmt. Bei den Stellen zwischen den Leitungen, die zwischen den Pfeilern gespannt sind, hat das Netz im Vergleich zum Himmel darüber zu hohe Disparitäten bestimmt.

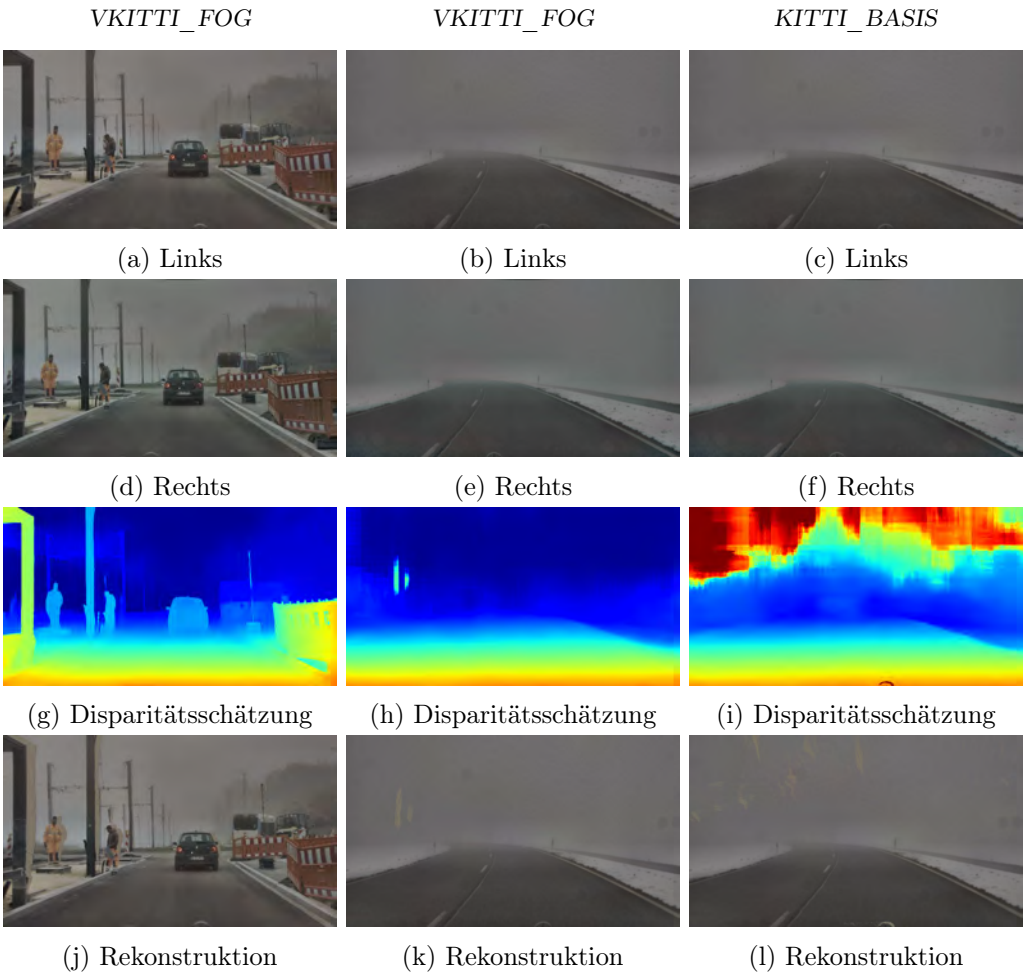


Abbildung 7.12: Ergebnisse des VKITTI_FOG (1. und 2. Spalte) und des KITTI_Basisnetzes (3. Spalte) für Szenen des STF-Datensatzes mit Nebel.

Die meisten sichtbaren Fehler in der Rekonstruktion (Abb. 7.12j) sind durch die Verdeckungen aus den verschiedenen Perspektiven der Kameras verursacht. In der zweiten Spalte der Abbildung 7.12 ist eine weitere Szene mit der Schätzung und Rekonstruktion von VKITTI_FOG zu sehen. In der dritten Spalte ist für dieselbe Szene die Schätzung vom Basisnetz dargestellt. Diese Szenen zeigt eine leere Straße mit dichtem Nebel, welcher die komplette obere Bildhälfte ausfüllt.

In der Rekonstruktion von VKITTI_FOG (Abb. 7.12k) sind bis auf die zwei Streifen auf der linken Seite keine Fehler zu erkennen. Die Disparitätskarte (Abb. 7.12l) zeigt für die untere Bildhälfte einen gleichmäßigen Verlauf der Disparitäten ohne große sichtbare Abweichungen. In der Bildmitte, wo die Straße im Nebel verschwindet, findet ein un-

gleichmäßiger Wechsel zwischen der Disparität der Straße und der niedrigen Disparität für die obere Bildhälfte statt. Im oberen Bildbereich, wo nur Nebel zu sehen ist, hat das Netz Disparitäten im Bereich 0,9 bis 1,5 bestimmt. Nur auf der linken Seite sind zwei größere Streifen mit größeren Abweichungen zum Hintergrund zu erkennen.

Die Rekonstruktion des Basisnetzwerks (Abb. 7.12k) zeigt für die obere Bildhälfte mehrere Stellen unterschiedlicher Größe, für die die bestimmte Pixelposition ungültig war. Bei Betrachtung der zugehörigen Disparitätskarte (Abb. 7.12i) ist ein großer Teil der oberen Bildhälfte, in der nur Nebel zu sehen ist, als Fläche mit sehr hohen Disparitäten bestimmt worden. Zur Bildmitte hin werden die Disparitäten geringer und zutreffender. Die Disparität für die Straße ist augenscheinlich korrekt bestimmt worden.

Die Ergebnisse für die anderen Testdaten entsprechen den dargestellten Ergebnissen. Die Ergebnisse des KITTI_FOG und DS_FOG-Netz sind denen des Basisnetzwerkes sehr ähnlich. Für Bildausschnitte mit Nebel enthalten die Disparitäten von KITTI_-FOG oft noch größere Flächen mit falschen Disparitäten. Das DS_FOG-Netz bestimmt im Durchschnitt noch größere Disparitäten für diese Ausschnitte.

7.4.4 Gesamtauswertung der Nebelnetzwerke

Über alle Testdaten zusammen erreicht das Basisnetzwerk den Metriken zu Folge die besten Ergebnisse. Am meisten Wert sollte jedoch auf die Ergebnisse für die PAD- und STF-Testdaten gelegt werden, da diese Szenarien mit realistischem Nebel repräsentieren. Für die PAD-Testdaten erzielte das Basisnetz die besten Ergebnisse, jedoch nicht ausreichend gute. Das größte Problem, dass der Nebel als eine Fläche mit viel zu großer Disparität erkannt wird, ist in allen Ergebnissen und vor allem bei den STF-Testdaten präsent.
Die Ursache dieses Problems ist die Homogenität des Nebels. Der Nebel besitzt für größere Abschnitte dieselbe Farbe oder zumindest einen gleichmäßigen Farbverlauf. Dies führt dazu, dass die Netze viele falsche Korrespondenzen finden, die meistens eine sehr große Disparität besitzen.

Im Gegensatz zum Basisnetz und den anderen Netzen ist das VKITTI_FOG-Netz viel besser darin, in Bildbereichen mit Nebel und ähnlichem wie z. B. dem Himmel bei STF_-CLEAR, keine Korrespondenzen mit zu hoher Disparität zu finden. Dies liegt an den Trainingsdaten des Netzes. Für die Bilder des VKITTI-Datensatzes existieren vollständige Disparitätskarten. Dabei werden auch Groundtruth-Disparitäten für die Bereiche

gegeben, in denen nur Himmel zu sehen ist. Diese Disparitäten sind immer kleiner 1, da der Himmel so weit entfernt ist, dass die Korrespondenzen kleiner als ein Pixel sein müssten, um dem größten Tiefenwert zu entsprechen. Das VKITTI_FOG-Netz hat somit beim Training gelernt, für homogene Abschnitte wie den Himmel oder dichten Nebel, sehr geringe Disparitäten zu vergeben.
Die anderen Netze konnte dies nicht lernen, da die KITTI-Datensätze und der DrivingStereo-Datensatz in den Groundtruth-Disparitätskarten für den oberen Bildbereich keine Disparitäten bereitstellen. Aus demselben Grund ist es nicht möglich, diesen Unterschied in der Bewertung für diese Testdaten zu berücksichtigen.

Des Weiteren hat sich gezeigt, dass die Bildqualität der Rekonstruktion die Qualität und Korrektheit der dafür verwendeten Disparitätskarte nicht ausreichend gut widerspiegelt. Dies gilt im Besonderen für Bilder mit Nebel. Wie anhand der dargestellten Rekonstruktionen und Disparitätskarten in Abbildung 7.11 und 7.12 zu erkennen ist, können für solche Bilder unterschiedliche Disparitäten zu einer augenscheinlich gleichen Rekonstruktion führen. Eine Bildqualitätsmetrik, wie die FSIM-Metrik ist nicht in der Lage, falsche Pixel zu erkennen, den richtigen RGB-Wert, aber die falsche Pixelposition besitzen.

Aus diesen Gründen ist die Bewertung der Ergebnisse des VKITTI_FOG-Netzes durch die Metriken nicht besser oder teilweise schlechter als die Ergebnisse der anderen Netze. Bei einer endgültigen Bewertung muss deshalb auch die Betrachtung der Disparitätskarten selber mit einbezogen werden. Dieser Unterschied macht die Ergebnisse des VKITTI_FOG-Netzes im Vergleich zum Basisnetz zu den besseren. Einsetzbar ist das Netz jedoch noch nicht, da noch zu große Unsicherheiten bestehen. Zudem ist ein großes Problem des Netzes, dass es an einigen Stellen Schwierigkeiten mit den Unterschieden der Domänen der Trainings- und Testdaten hat. Besonders geringe Beleuchtung oder starke Überbelichtung verursachen oft große Abweichungen. Dies ist ein Nachteil der synthetischen Daten, die im Training verwendet wurden.

7.5 Ergebnisse der Verdeckungsnetzwerke

Die Auswertung der Verdeckungsnetzwerke wurde mit den aufgenommenen Testszenarien 2 bis 6 (s. Abschnitt 5.2) durchgeführt. Für die Bewertung werden die Groundtruth-Metriken EPE, D1 und der 1-Pixel-Fehler verwendet. Es werden dabei die modifizier-

ten Groundtruth-Disparitätskarten verwendet (s. Abschnitt 5.2.3). Für eine ausgewählte Szene des Szenarios werden die geschätzten Disparitätskarten und die zugehörigen Fehlerkarten einiger Netze bzw. der Kameras präsentiert.

7.5.1 Auswertung für das Szenario ohne Störeffekt

Die Ergebnisse der Auswertung der Netzwerke für das Szenario ohne Störeffekte sind in Tabelle 7.11 dargestellt. In Abbildung 7.13 ist das Szenario mit Groundtruth und einer Auswahl von Disparitätsschätzung und Fehlerkarten dargestellt. Das ZED_OCC_-REAL-Netzwerk erreicht mit einem EPE von 1,228 und einem D1-Fehler von 5,339 % im Durchschnitt die besten Ergebnisse. In der erzeugten Disparitätskarte in Abbildung 7.13e sind klare Objektkonturen und keine starken Abweichungen zu erkennen. Die zugehörige Fehlerkarte (7.13i) zeigt, dass für die gleichmäßigen Flächen der Wand leichte bis mittlere Abweichungen vorhanden sind. Für die Halterung der Plexiglasscheibe wurde die Disparität nur zum Teil bestimmt. Der größte Teil wurde ignoriert und die Disparität für den dahinter liegenden Schrank bestimmt. Der Teil, der nicht ignoriert wurde, macht den größten Teil des Fehlers aus. Für die spiegelnde Stelle des Whiteboards hat das Netz die gleichen Disparitäten bestimmt wie der Stelle daneben.

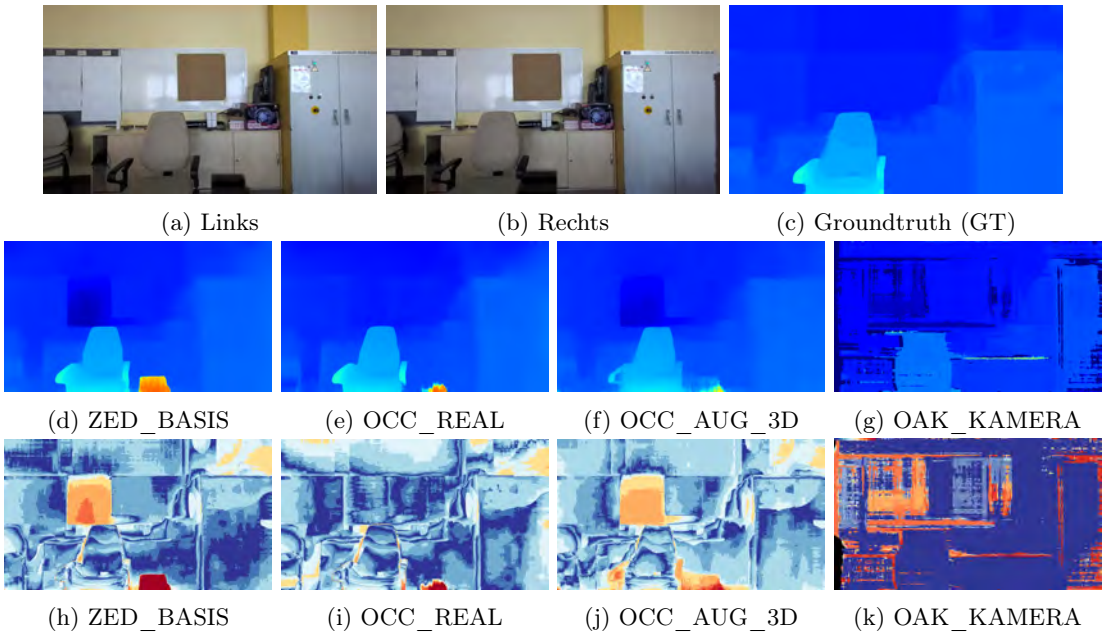
Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**$>$ 1 px (%)**
ZED_SCENA_CLEAN	ZED_BASIS	1,256	8,195	28,740
	ZED_BASIS_PATCH	1,315	6,974	26,397
	ZED_OCC_REAL	**1,228**	**5,339**	31,681
	ZED_OCC_AUG_2D	1,806	10,322	26,292
	ZED_OCC_AUG_3D	1,637	10,148	*42,707*
	ZED_KAMERA	1,720	7,229	**14,781**
	OAK_KAMERA	*6,213*	*30,204*	34,106

Tabelle 7.11: Ergebnisse der Verdeckungsnetzwerke für das Szenario ZED_SCENA_-CLEAN.

Das Basisnetzwerk erzielte im Vergleich leicht schlechtere Ergebnisse. Wie in der Disparitäts- (Abb. 7.13d) und Fehlerkarte (Abb. 7.13h) zu sehen ist, hat das Basisnetz die Halterung nicht ignoriert. Zudem führte die spiegelnde Stelle des Whiteboards zu starken Abweichungen der Disparitäten. Die Ergebnisse des ZED_BASIS_PATCH-Netzes sind vergleichbar mit den des Basisnetzwerkes. Für die freie Fläche oben rechts ist die Abweichung aber geringer, wodurch die bessere Wertung zustande kommt.

Das Ergebnis von ZED_OCC_3D-Netz zeigt eine allgemeine hohe Abweichung. Von allen gültigen Disparitäten besitzen 42,707 % mehr als 1 px Abweichung. Dafür hat das Netz weniger Schwierigkeiten mit der Halterung, die in der Disparitätskarte (7.13f) nur noch zu kleinen Teilen zu erkennen ist. Die Disparitätskarte des ZED_OCC_AUG_-2D-Netzes besitzt im Allgemeinen eine geringere Abweichung für die meisten Flächen in der Szene. Nur für die spiegelnden Stellen und die Halterung sind die Disparitäten stark abweichend.

Die Disparitätsschätzung der ZED-Kamera enthält größere Abweichungen für die spiegelnde Stelle, die Halterung und die Wand in der linken oberen Bildecke. Die OAK-Kamera liefert die schlechtesten Ergebnisse. Die Disparitätskarte (7.13g) zeigt viele Streifen, für die Disparität nicht bestimmt wurde. Zudem hat das Netz für die obere Kante des Schrankes sehr hohe Disparitäten bestimmt, die stark von den eigentlichen Werten abweichen. Da die Groundtruth-Disparitätskarte zur Auswertung der Schätzung auf dieser Schätzung basiert ist, stimmt ansonsten ein großer Teil der Disparitäten überein. Dies ist an den gleichmäßigen dunkelblauen Flächen in der Fehlerkarte (7.13g) zu erkennen.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_REAL (f) OCC_AUG_3D (g) OAK_KAMERA

(h) ZED_BASIS (i) OCC_REAL (j) OCC_AUG_3D (k) OAK_KAMERA

Abbildung 7.13: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_CLEAN.

7.5.2 Auswertung für Szenario 2 - Einzelne Verdeckung

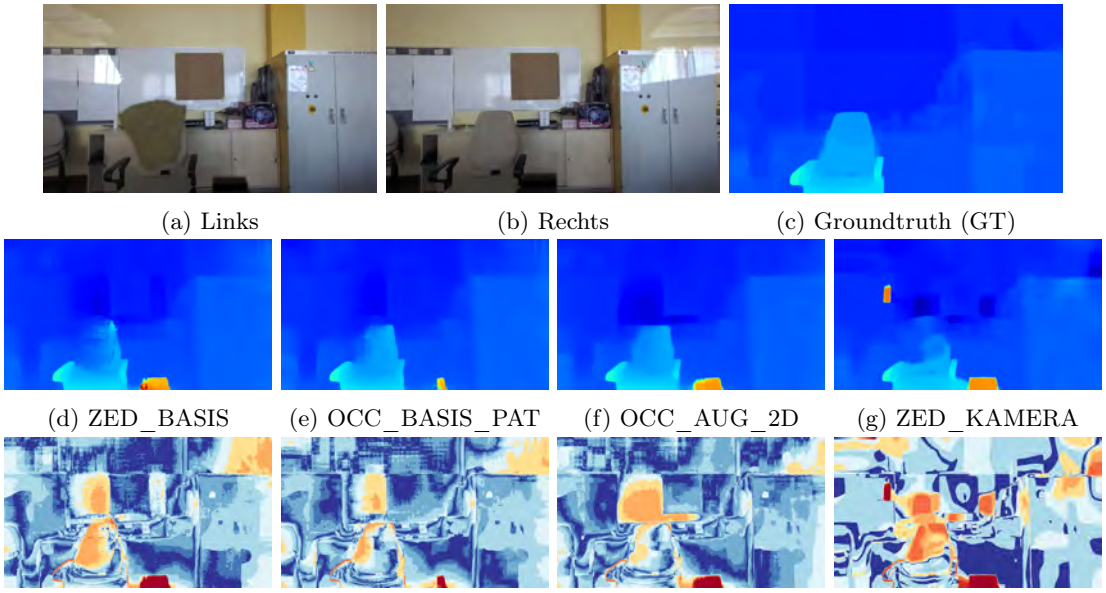
Das Szenario 2 umfasst Aufnahmen von Verdeckungen im linken Bild mit variierender Größe. Die Ergebnisse sind in Tabelle 7.12 dargestellt. In Abbildung 7.14 ist die Szene für Steckplatz 5 mit einigen Ergebnissen zu sehen.
Die beste Leistung für dieses Szenario erbringt das ZED_BASIS_PATCH-Netzwerk. Für die Szene mit Steckplatz 1 (s. Abb. A.10) bei der das untere linke Viertel des Bildes verdeckt ist, lässt sich der Stuhl nur schemenhaft erkennen. Die Disparitäten sind dabei stark abweichend. Je weiter die Plexiglasscheibe entfernt und die Verdeckung kleiner wird, desto besser werden die Ergebnisse. Die Disparitätskarte für Szene 5 (Abb. 7.14e) und die zugehörige Fehlerkarte (Abb. 7.14i) zeigen das nur die linke Seite der Stuhllehne stärker abweichende Disparitäten aufweist. Die Plexiglas-Halterung wurde fast vollständig ignoriert.

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
ZED_SCENA_TAPE	ZED_BASIS	2,188	16,263	49,349
	ZED_BASIS_PATCH	**1,722**	11,849	47,642
	ZED_OCC_REAL	1,763	**10,703**	48,714
	ZED_OCC_AUG_2D	2,485	15,045	43,332
	ZED_OCC_AUG_3D	2,113	14,835	*57,953*
	ZED_KAMERA	5,198	19,522	**42,074**
	OAK_KAMERA	*11,195*	*42,862*	50,162

Tabelle 7.12: Ergebnisse der Verdeckungsnetzwerke für das Szenario ZED_SCENA_TAPE.

Die Leistung von ZED_OCC_REAL ist vergleichbar. Nur für die Halterung sind die Ergebnisse ein wenig schlechter. Das Basisnetzwerk hat dieselben Probleme wie die vorherigen Netze, aber mit wesentlich stärkeren Abweichungen. Zudem ist die Stelle mit den Abweichungen auf der Stuhllehne größer als bei den anderen Netzen. Für die Halterung hat Netz die Disparitäten fast vollständig bestimmt.

Das ZED_OCC_AUG_2D-Netz zeigt die besten Ergebnisse in Bezug auf die Vollständigkeit der Stuhllehne, wie in Abbildung 7.14f zu sehen ist. Die Abweichungen für die Spiegelung im Whiteboard und der Halterungen erhöhen den durchschnittlichen Fehler aber wieder. Das ZED_OCC_AUG_3D-Netz dagegen ignoriert die Halterung teilweise vollständig. Dafür sind die Disparitäten für die Stuhllehne meist zu großen Teil stark abweichend.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_BASIS_PAT (f) OCC_AUG_2D (g) ZED_KAMERA

(h) ZED_BASIS (i) OCC_BASIS_PAT (j) OCC_AUG_2D (k) ZED_KAMERA

Abbildung 7.14: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_TAPE Steckplatz 5.

Für die Szene 1 ist die ZED-Kamera nicht in der Lage, zumindest grob die Form des Stuhles zu erkennen und die Disparitäten annähernd zu bestimmten. Für die Szenen mit kleinerer Verdeckung ist die Form des Klebebandstreifens in der Disparitätskarte (Abb. 7.14g) immer deutlich zu erkennen. Für diese Stelle und die Spiegelung bestehen größere Abweichungen.
Die OAK-Kamera schafft es noch weniger für die verdeckten Stellen sinnvolle Disparitäten zu bestimmen. Die Disparitätskarten enthalten starke Abweichungen in beide Richtungen.

7.5.3 Auswertung für Szenario 3 - Großflächige Verdeckung

Szenario 3 umfasst großflächige Verdeckung durch eine Fläche mit unterschiedlich großen Ausschnitten im linken Bild. Je weiter entfernt die Plexiglasscheibe gesetzt ist, desto kleiner ist die gesamt verdeckte Fläche. Dafür wird mehr von der betroffenen Stelle verdeckt. In der Szene für Steckplatz 1 (s. Abbildung A.11) wird ein Großteil der rechten Bildhälfte bis auf einen Ausschnitt verdeckt. In der letzten Szene betrifft die Verdeckung nur Teile

Testdaten	Netzwerk-Name	EPE (px)	D1-all(%)	> 1 px (%)
ZED_SCENA_PAPER	ZED_BASIS	4,310	32,945	72,777
	ZED_BASIS_PATCH	**1,722**	11,849	**47,642**
	ZED_OCC_REAL	1,763	**10,703**	48,714
	ZED_OCC_AUG_2D	2,964	21,611	60,108
	ZED_OCC_AUG_3D	2,113	14,835	57,953
	ZED_KAMERA	19,922	51,261	70,889
	OAK_KAMERA	*21,694*	*76,942*	*80,730*

Tabelle 7.13: Ergebnisse der Verdeckungsnetzwerke für das Szenario ZED_SCENA_-PAPER.

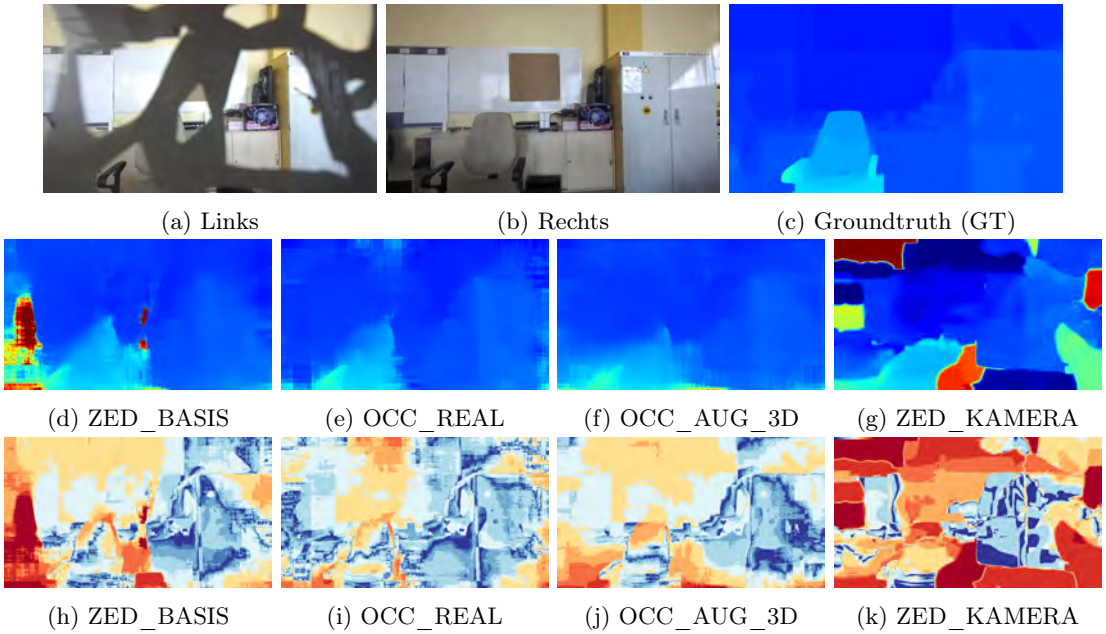
der Bildmitte. Die Ergebnisse der Netzwerke sind in Tabelle 7.13 zu sehen. In Abbildung 7.15 wird die Szene für Steckplatz 2 mit einigen Disparitätsschätzung dargestellt. In der dargestellten Szene ist ein großer Teil des Stuhls, des braunen Rechtecks auf dem Whiteboard und des Schrankes auf der rechten Seite verdeckt.

Für dieses Szenario liefert das ZED_OCC_REAL und das ZED_BASIS_PATCH-Netzwerk sehr ähnliche Ergebnisse. Das ZED_BASIS_PATCH-Netz hat den niedrigen EPE und ZED_OCC_REAL dafür einen geringeren D1-Fehler. Die Fehlerkarten für die Disparitätsschätzung zeigen, dass ZED_OCC_REAL-Netzwerk insgesamt mehr, aber dafür kleinere Abweichungen erzeugt hat.

Für die dargestellte Szene schafft es das ZED_OCC_REAL-Netz für den Schrank Disparitäten mit mittelgroßen Abweichungen zu bestimmen. Für die verdeckte Stelle des Whiteboards und den oberen Teil der Stuhllehne sind die Differenzen zum Groundtruth größer, wie in der Fehlerkarte (Abb. 7.15i) zu erkennen ist. Der Rest des Stuhls ist nur grob und undeutlich erkennbar. Die Ergebnisse des ZED_BASIS_PATCH-Netzwerks sind ähnlich zu den von ZED_OCC_REAL. Für die bereits genannten Problemstellen sind die Abweichung im Vergleich aber größer.

Die Disparitätskarten des ZED_OCC_AUG_3D-Netzes enthalten im Vergleich zu den anderen beiden Netzen durchschnittlich größere Abweichung. Zudem ist die Unsicherheit für den Stuhl größer, wodurch dieser nicht deutlich erkennbar ist (s. Abb. 7.15f). Das Netz hat insgesamt größere Schwierigkeiten, den verdeckten Stellen sinnvolle Disparitäten zu füllen.

Im Vergleich dazu besitzen die Disparitätskarten des ZED_OCC_AUG_2D-Netzes deutlichere Konturen für den Stuhl und den Schrank auf der linken Seite. Grund für die durchschnittlich schlechtere Bewertung sind einige fehlerhafte Stellen mit viel zu hoher



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_REAL (f) OCC_AUG_3D (g) ZED_KAMERA

(h) ZED_BASIS (i) OCC_REAL (j) OCC_AUG_3D (k) ZED_KAMERA

Abbildung 7.15: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_PAPER Steckplatz 2.

Disparität. Diese werden durch größere Falten an den Rändern der Folie verursacht. Größere Differenzen für die Spiegelung im Whiteboard verschlechtern die Wertung zusätzlich. Das Basisnetzwerk hat dieselben Schwierigkeiten wie ZED_OCC_AUG_2D. Die verdeckenden Teile des Papiers führen zu einigen Stellen mit sehr hohen Disparitäten, wie in Abbildung 7.15d und 7.15h zu erkennen ist. Für die kleineren Verdeckungen in der Bildmitte schafft es das Netz stellenweise die Disparitäten des Hintergrundes zu bestimmen.

Die Disparitätskarten der ZED-Kamera enthalten viele mit sehr starken Abweichungen. Die Flächen entsprechen der durch die Verdeckung verdeckten Stellen. Für Verdeckung der Bildmitte, wie z. B. in der dargestellten Szene, bestimmt die Kamera meist sehr geringe Disparitäten. Wie in der Disparitätskarte (Abb. 7.15g) der Szene 2 zu erkennen ist, werden dem verdeckten Teil des Stuhles die Disparitäten des Whiteboards dahinter zugewiesen. Die fehlerhaften Stellen an den Rändern bestehen aus zu großen Disparitäten.

Die OAK-Kamera ist nicht in der Lage, für die Aufnahmen nutzbare Disparitätskarten zu erzeugen. Der Großteil der Disparitäten ist ungültig und der Rest stark abweichend. Es lassen sich keine Objekte der Szene klar erkennen.

7.5.4 Auswertung für Szenario 4 - Stereo-Verdeckung

Das Szenario 4 umfasst zwei Szenen, in denen ein Blatt als Verdeckung in beiden Bildern zu sehen ist. In der ersten Szene ist das Blatt ca. 15 cm weit von der Kamera entfernt und verdeckt im rechten Bild die Lehne des Bürostuhls. In der zweiten Szene (s. Abbildung A.12) befindet sich das Blatt ca. 8 cm entfernt vor der Plexiglas-Halterung im linken Bild. Für dieses Szenario ist die Leistung des ZED_OCC_REAL-Netzes außerordentlich gut. Für die in Abbildung 7.16 dargestellte erste Szene schafft das Netz es, das Blatt fast vollständig zu ignorieren. In der Disparitäts- und Fehlerkarte (Abb. 7.16e, 7.16i) ist am rechten Schrank nur eine kleine Stelle mit größerer Abweichung zu erkennen. Des Weiteren ist die Halterung nur zu kleinen Teilen sichtbar. In der Disparitätskarte für die zweite Szene lässt sich keine Abweichung in Bezug auf das Blatt oder die Halterung feststellen.

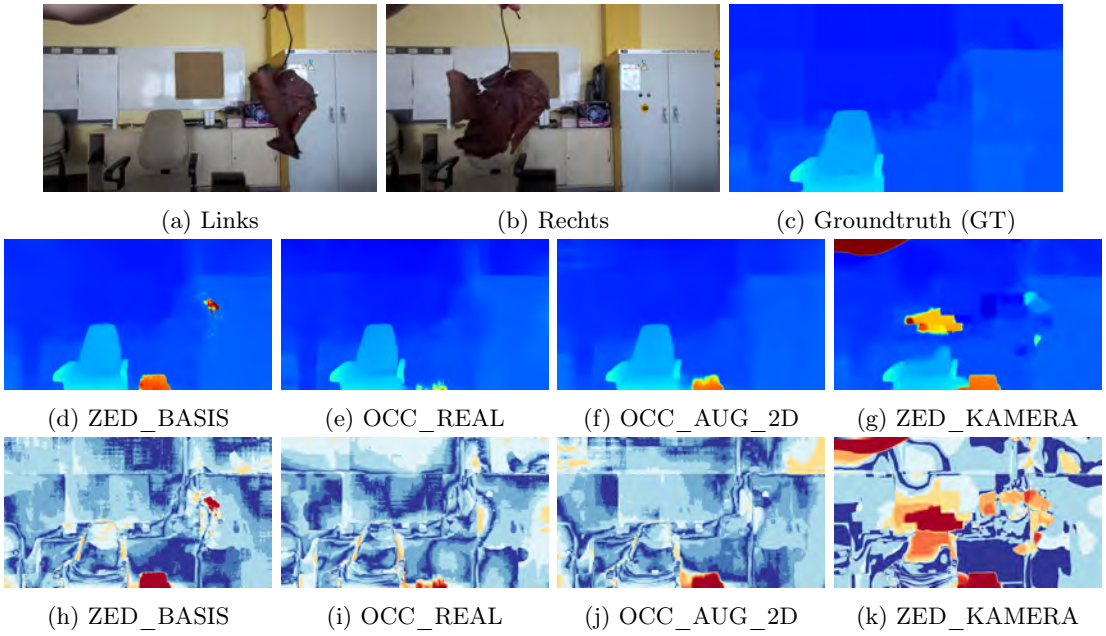
Das ZED_BASIS_PATCH-Netzwerk zeigt ähnlich gute Ergebnisse. Für die erste Szene bestimmt es die Disparität für eine kleine Stelle des Armes links oben im Bild. In der zweiten Szene sorgt die Spiegelung für größere Abweichungen und damit für einen größeren Fehler. Das Blatt wird aber ähnlich gut herausgefiltert. Das ZED_OCC_AUG_-2D-Netz erbringt in Bezug auf das Blatt die besten Ergebnisse. Bei Betrachtung der Disparitätskarte und zugehöriger Fehlerkarte (Abb. 7.16f, 7.16i) lassen sich nur minimale

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**$>$ 1 px (%)**
ZED_SCENA_LEAF	ZED_BASIS	1,893	9,701	35,415
	ZED_BASIS_PATCH	1,218	6,646	**31,330**
	ZED_OCC_REAL	**1,057**	**3,588**	38,831
	ZED_OCC_AUG_2D	1,540	8,608	34,124
	ZED_OCC_AUG_3D	1,775	10,497	*53,877*
	ZED_KAMERA	8,468	20,586	37,437
	OAK_KAMERA	*13,851*	*48,808*	53,297

Tabelle 7.14: Ergebnisse der Verdeckungsnetzwerke für das Szenario ZED_SCENA_-PAPER.

Abweichungen feststellen. Die schlechtere Gesamtwertung wird von Abweichungen für die Spiegelung und die Halterung verursacht.

Das ZED_OCC_AUG_3D-Netz dagegen schafft es nicht, das Blatt für die Bestimmung der Disparitäten zu ignorieren. In der ersten Szene wird zudem die Stuhllehne nicht eindeutig erkannt. Die Disparitätskarte der zweiten Szene enthält stärke Abweichungen für die linke Armlehne des Stuhls und die Stelle links daneben. Der Fehler für die Plexiglas-Halterung ist dafür geringer als beim 2D-Netz.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_REAL (f) OCC_AUG_2D (g) ZED_KAMERA

(h) ZED_BASIS (i) OCC_REAL (j) OCC_AUG_2D (k) ZED_KAMERA

Abbildung 7.16: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_LEAF Szene 1.

Das Basisnetz zeigt eine überraschend gute Leistung in Bezug auf die Störung durch das Blatt. In der Disparitätskarte für die erste Szene (Abb. 7.16d) ist die Stuhllehne klar erkennbar und besitzt nur geringe Abweichungen. Auf der rechten Seite ist eine kleinere Stelle mit zu hohen Disparitäten zu sehen, die durch das Blatt verursacht wurden. Für Szene 2 existieren für die Stelle neben dem Stuhl größere Abweichungen. Zudem ist der Umriss des Blattes vor der Halterung zu erkennen.

Die ZED-Kamera hat große Schwierigkeiten, die Disparitäten für die verdeckten Stellen beider Bilder zu bestimmen. Für die erste Szene führt die Verdeckung im rechten Bild

zu einer größeren Stelle mit hohen Disparitäten, wie in der Disparitätskarte (Abb. 7.16g) zu erkennen ist. Die verdeckte Stelle im linken Bild führt zu vereinzelten stärkeren Abweichungen. In der zweiten Szene führt das Blatt zu starker Abweichung für beinah die gesamte Fläche der verdeckten Stellen beider Bilder.

Die OAK-Kamera ist nicht in der Lage, korrekte Disparitäten für die verdeckten Stellen zu bestimmen. Für die verdeckte Stuhllehne in der ersten Szene wurden stellenweise sehr große Disparitäten bestimmt und ansonsten nur ungültige. Für die zweite Szene wurden für nahezu den gesamten Bereich um die Halterung nur ungültige Disparitäten bestimmt. Selbes gilt für den Bereich neben der linken Stuhllehne.

7.5.5 Auswertung für Szenario 5 - Transparente Verdeckung

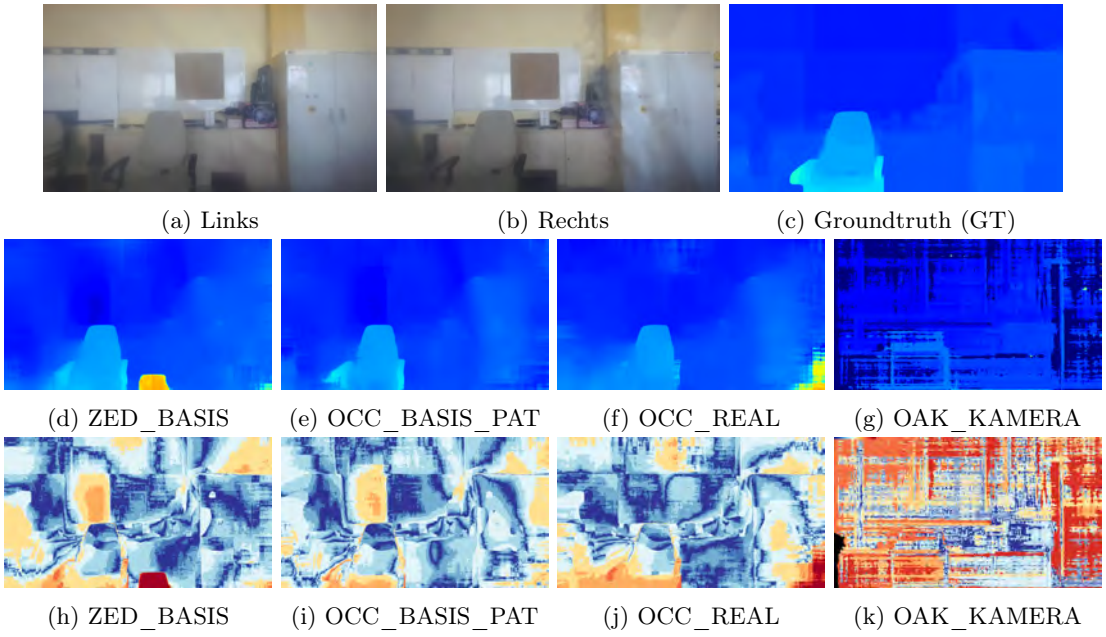
Das Szenario 5 umfasst Aufnahmen mit der Verdeckung durch eine transparente Folie. Es werden dabei keine Bereiche vollständig verdeckt, sondern durch die Folie hauptsächlich unscharf dargestellt. Vereinzelte Falten in der Oberfläche verursachen weitere Unreinheiten oder beeinflussen die Farbe des Hintergrundes. Je weiter die Scheibe von der Kamera entfernt wird, desto größer werden die nicht verdeckten Bereiche an dem jeweiligen äußeren Bildrand. In Tabelle 7.15 sind die Ergebnisse für das Szenario aufgeführt. In Abbildung 7.17 ist die Szene für den Steckplatz 1 und eine Auswahl von Disparitätsschätzung dargestellt.

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**$>$ 1 px (%)**
ZED_SCENA_TRANSP	ZED_BASIS	2,131	19,229	52,719
	ZED_BASIS_PATCH	**1,751**	**14,292**	49,748
	ZED_OCC_REAL	1,976	16,824	56,183
	ZED_OCC_AUG_2D	2,095	14,850	**45,372**
	ZED_OCC_AUG_3D	1,910	16,593	61,115
	ZED_KAMERA	8,097	40,007	64,733
	OAK_KAMERA	*9,117*	*49,610*	*67,637*

Tabelle 7.15: Ergebnisse der Verdeckungsnetzwerke für das Szenario ZED_SCENA_-TRANSP.

Das ZED_BASIS_PATCH-Netz erreicht für dieses Szenario die besten Ergebnisse. Im Allgemeinen führt die Störung der Folie im Vergleich zu ZED_SCENA_CLEAN im Durchschnitt zu einer leicht größeren Abweichung. Für Stellen, wo größere Falten in der Oberfläche der Folie sind, können diese Abweichungen noch stärker ausfallen. In 50 %

der Szenen wird die Halterung und die Spiegelung vollständig ignoriert.
Das ZED_OCC_REAL- und das ZED_OCC_AUG_3D-Netz haben Schwierigkeiten, den Bürostuhl und dabei besonders die linke Armlehne deutlich zu erkennen. Das ZED_OCC_REAL-Netz schafft es für jede Szene, die Halterung zu ignorieren, wie z. B. in Abbildung 7.17f. Die Falten der Folie links unten im rechten Bild führen dafür zu einer größeren Stelle mit zu hohen Disparitäten. Für das ZED_OCC_AUG_3D sind die Falten unten rechts kein Problem. Dafür ist der Bereich mit Abweichungen um den Stuhl herum größer.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_BASIS_PAT (f) OCC_REAL (g) OAK_KAMERA

(h) ZED_BASIS (i) OCC_BASIS_PAT (j) OCC_REAL (k) OAK_KAMERA

Abbildung 7.17: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_TRANSP Steckplatz 1.

Die im Vergleich schlechteren Ergebnisse von ZED_OCC_AUG_2D werden durch die Fehler für die Spiegelung und die Halterung verursacht. In Bezug auf die Abweichungen für den Bürostuhl sind die Disparitätskarten von ZED_OCC_AUG_2D besser als bei den anderen Netzen. Die vom Basisnetzwerk erzeugten Disparitätskarten besitzen im Allgemein für die Szene, abgesehen von einigen Problemstellen nur leichte Abweichungen. Für die Spiegelung und besonders die Halterung wurden jedoch stark abweichende Disparitäten bestimmt, wie in Abbildung 7.17h zu sehen ist. Auch für den Stuhl und den Bereich drumherum existieren größere Abweichungen.

Für die ZED-Kamera führt die Störung der Folie zu Stellen mit starken Abweichungen für die Wand am oberen Bildrand und den Schrank auf der linken Seite. Der fehlerhafte Bereich um den Stuhl herum ist je nach Szene unterschiedlich groß. Die Abweichung der Disparitäten für die Rückenlehne ist dafür meistens nur gering.
In den Disparitätskarten der OAK-Kamera ist zu erkennen, dass die Störung zu verstärken Vorkommen von ungültigen Disparitäten führt. In der Disparitätskarte für Szene 1 (Abb. 7.17g) ist die Kontur des Stuhls grob zu erkennen. Für die Fläche wurden aber zu geringe und ungültige Disparitäten bestimmt. Die Ergebnisse für die anderen Szenen sind ähnlich.

7.5.6 Auswertung für Szenario 6 - Eis/Frost

Für das Eis-Szenario erreicht das Basisnetzwerk im Vergleich die besten Ergebnisse (s. Tabelle 7.16). Ein EPE von 3,454 und ein D1-Fehler von 31,853 % ist jedoch kein ausreichend gutes Ergebnis. In Abbildung 7.18 sind die Aufnahme für Steckplatz 5 und einige der geschätzten Disparitätskarten dargestellt. In dieser Szene verdeckt das Eis im linken Bild ein Großteil der Bildmitte und im rechten Bild nur die Bildmitte auf der linken Seite.

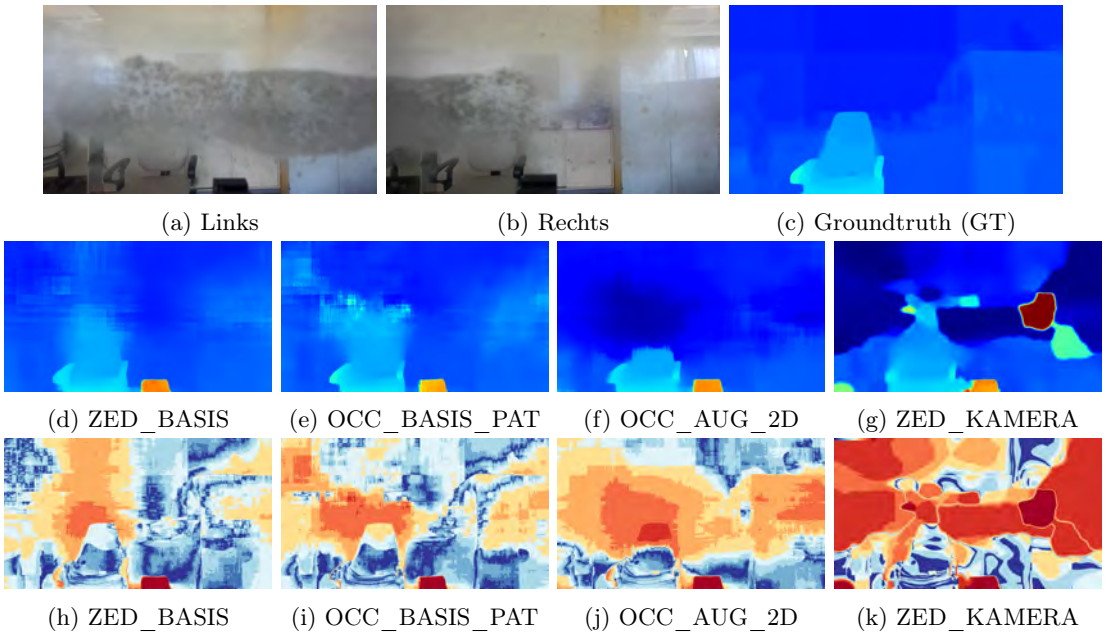
Die Disparitätsschätzung (7.18d) und die Fehlerkarte (7.18h) des Basisnetzwerkes zeigen große Unsicherheiten für den Bereich hinter dem Bürostuhl. Das Netz hat dabei Schwierigkeiten, das Eis und den Stuhl zu unterscheiden. Da im rechten Bild die Plexiglasscheibe für die rechte Bildhälfte nur *beschlagen* und nicht ganz verdeckt ist, konnte das Netz die Disparitäten der Szene dort besser bestimmen.

Das ZED_BASIS_PATCH-Netz erzeugte ähnliche Disparitätskarten wie das Basisnetz, aber mit größeren Unsicherheiten für die genannten Stellen. In der Disparitätskarte

Testdaten	**Netzwerk-Name**	**EPE (px)**	**D1-all(%)**	**> 1 px (%)**
ZED_SCENA_ICE	ZED_BASIS	**3,454**	**31,853**	64,107
	ZED_BASIS_PATCH	3,619	32,800	**64,000**
	ZED_OCC_REAL	7,792	51,683	78,715
	ZED_OCC_AUG_2D	4,856	46,005	71,302
	ZED_OCC_AUG_3D	5,505	45,205	76,214
	ZED_KAMERA	16,794	56,986	73,855
	OAK_KAMERA	*17,346*	*71,381*	*80,552*

Tabelle 7.16: Ergebnisse der Verdeckungsnetzwerke für das Szenario ZED_SCENA_-ICE.

(7.18d) ist zu erkennen, dass die fehlerhafte Stelle bei der Stuhllehne größer ist als bei dem Basisnetz. Zusätzlich verursachte die Spiegelung in der oberen linken Ecke einen größeren Bereich von Fehlschätzungen. Die Disparitäts- und Fehlerkarte (Abb. 7.18f, 7.18j)



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_BASIS_PAT (f) OCC_AUG_2D (g) ZED_KAMERA

(h) ZED_BASIS (i) OCC_BASIS_PAT (j) OCC_AUG_2D (k) ZED_KAMERA

Abbildung 7.18: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_ICE Steckplatz 5.

von ZED_OCC_AUG_2D zeigen dieselben Probleme der vorherigen Netze nur mit stärkeren Abweichungen. Eine Besonderheit ist, dass für die Stelle um die Stuhllehne keine zu hohen Disparitäten, sondern zu kleine bestimmt wurden.

Die Ergebnisse von ZED_OCC_AUG_3D sind von der Wertung der Metriken besser als die des 2D-Netzes. Dies liegt daran, dass das 3D-Netz die Disparität für die Plexiglas-Halterung nicht immer vollständig bestimmt hat. Dadurch ist der Fehler kleiner. Die Abweichungen für die durch das Eis betroffenen Stellen sind größer. Selbes gilt für das ZED_OCC_REAL-Netz, dessen Disparitätskarten noch größere Abweichungen ausweisen.

Die ZED-Kamera hat für die mit Eis bedeckten Stellen meist sehr geringe Disparitäten bestimmt. Vereinzelten Stelle wurden aber auch sehr hohe Disparitäten zugewiesen, wie in der Disparitätskarte in Abbildung 7.18g zu sehen ist. Die Lehne des Bürostuhls ist im Allgemeinen noch gut zu erkennen. Für die OAK-Kamera hat das Eis zu einem

stark verstärkten Vorkommen von ungültigen Stellen geführt. In den Disparitätskarten lassen sich nur schwer die Objekte der Szene ausmachen. Zudem ist die durchschnittliche Abweichung sehr hoch.

7.5.7 Gesamtauswertung - Verdeckungsnetzwerke

Im Allgemeinen erbringt das ZED_OCC_REAL-Netzwerk den Metriken zu Folge für alle Szenarien die besten Ergebnisse. Dies liegt unter anderem an den durchschnittlich geringeren Fehlern für die Spiegelung im Whiteboard, welche nicht im Fokus dieser Auswertung steht. Betrachtet man nur die von der Verdeckung betroffenen Stellen, zeigen das ZED_BASIS_PATCH und das ZED_OCC_AUG_2D-Netzwerk ähnlich gute und teilweise bessere Ergebnisse. Dies ist besonders im Hinblick auf Szenario 2 (einzelne Verdeckung) und 4 (Stereo-Verdeckung) zu erkennen, wo der Einfluss der Verdeckung teils sehr minimal ist. Alle drei Netze schaffen es zudem auch in einigen Aufnahmen die stetige Verdeckung durch die Plexiglas-Halterung zu ignorieren und die Disparitäten des dahinterliegenden Schrankes zu bestimmen. Besonders das ZED_OCC_AUG_2D-Netzwerk hat dabei jedoch mehr Probleme, wenn eine andere Verdeckung z. B. das Blatt in der Nähe sichtbar ist.

Die Verdeckungen in Szenario 6 (Eis) und Szenario 3 (großflächige Verdeckung), bei denen ein großer Teil des Bildes verdeckt bzw. schlechter Sichtbar machen, stellen eine größere Herausforderung für die Netze dar. Das für Szenario 6, das Basisnetz die besten Ergebnisse erreicht hat, zeigt, dass die Trainingsdaten für dieses Szenario nicht repräsentativ waren. In Szenario 3 scheint es, dass kein Training zu ausreichend guten Ergebnissen führen konnte oder die Verdeckung zu grundsätzlich zu groß ist. Im Allgemeinen ist jedoch eine Verbesserung der Leistung der trainierten Netzwerke im Vergleich zu dem Basisnetzwerk festzustellen.

Für eine optimale Auswertung, die die Leistung in Bezug auf die verdeckten Bildbereiche betrachtet, müssten Objekt-Masken verwendeten werden. Ein Netz, das lange auf dem Datensatz trainiert hat, könnte den Fehler für die Umgebung minimiert, aber nicht für die Stellen, welche durch das Objekt verdeckt sind. So könnte es dazu kommen, dass der Fehler insgesamt geringer ist als vor dem Training, die Disparitäten an den verdeckten Stellen aber nicht korrekt sind. Mit Objekt-Masken ist es möglich, den Fehler nur für die von der Störung beeinflussten Stellen zu berechnen.

8 Fazit und Ausblick

Im Folgenden wird ein Fazit für die erarbeiteten Ergebnisse und erhaltenen Erkenntnisse gezogen. Des Weiteren wird ein Überblick über alternative Ansätze oder Erweiterungen gegeben, die in zukünftigen Arbeiten untersucht werden könnten.

8.1 Fazit

In dieser Arbeit wurde der Einsatz von neuronalen Netzwerken zur Lösung des Stereokorrespondenzproblems unter dem Einfluss von umweltbedingten Störungen untersucht. Der Fokus wurde dabei auf die Störung durch Regentropfen, Nebel und allgemeine Verdeckung durch Objekte gelegt. Es wurde eine allgemeine Netzwerkarchitektur für Stereokorrespondenz als Grundlage gewählt, die mit unterschiedlichen Daten trainiert wurde. Um die Menge an Trainingsdaten zu erweitern, wurden verschiedene Methoden zur Augmentation von existierenden Daten verwendet, um künstliche Störeffekte hinzuzufügen. Für die Auswertung und den Vergleich mit der Leistung von Stereokameras wurden Bilder von Testszenarien für verschiedene Störeffekte aufgenommen.

Aus der Auswertung der trainierten Netze ging hervor, dass das Training von Netzwerken mit Trainingsdaten für die Störeffekte im Allgemeinen zu einer verbesserten Leistung für die Bestimmung von Korrespondenzen mit den betrachteten Störeffekten führt. Besonders deutlich wurde dies im Hinblick auf die Störung durch Regentropfen. Das Netzwerk, dessen Trainingsdaten Aufnahmen echter Regentropfen umfassten, war in der Lage, Disparitätsschätzungen ohne große Fehler zu erzeugen. Für Anwendungsfälle, bei denen ausschließlich Tropfen auf der Linse oder der Scheibe davor die Störung ausmachen, ist dies eine anwendbare Lösung.

Die Auswertung der Netzwerke für Verdeckungen hat ebenso vielversprechende Erfolge aufgezeigt. Das Training mit Daten mit echten und künstlichen Verdeckungen hat jeweils

zu verbesserten Ergebnissen für kleine und mittelgroße Verdeckungen geführt. Für großflächige Verdeckung oder die Verdeckung durch Eis konnten dagegen keine ausreichend guten Ergebnisse erreicht werden. Der Einfluss der Verdeckung konnte zwar reduziert werden, jedoch konnte in einigen Fälle die Disparitäten der verdeckten Bereiche nicht bestimmt werden. Ob bessere Trainingsdaten oder eine andere Netzwerkarchitektur zu ausreichend guter Leistung führen würde, müsste weiter untersucht werden.

Für die Störung durch Nebel waren gemischte Erfolge zu beobachten. Das Training mit den synthetischen Nebelbildern des VirtualKITTI-Datensatzes hat unerwarteterweise zu den Ergebnissen mit dem meisten Potenzial geführt. Die Eigenschaften des trainierten Netzwerkes, für nebelige Flächen niedrige und keine abnormal großen Disparitäten zu bestimmen, lässt es insgesamt die besseren Ergebnisse erbringen. Ausreichend ist die Leistung jedoch nicht, da noch viele Informationen durch den Nebel verloren gehen. Ob ein längeres Training zu noch besseren Ergebnissen führt, müsste weiter untersucht werden. Ein Netzwerk nur mit synthetischen Daten trainieren zu können, dass für reale Situationen zuverlässige Schätzungen liefert, wäre ein großer Erfolg. Die fehlende Verfügbarkeit von Datensätzen mit echten Aufnahmen, deren Groundtruth-Daten nutzbar sind, wäre dadurch weniger problematisch.

Es hat sich zudem gezeigt, dass die verwendeten Methoden zur Evaluierung der Netzwerke die Leistung nicht optimal bewerten. Eine Bildqualitätsmetrik in Kombination mit der Rekonstruktion basierend einer geschätzten Disparitätskarte für Nebelbilder eignet sich nicht für die Bewertung dieser Disparitätskarte. Des Weiteren eignet sich der Vergleich der gesamten geschätzten Disparitätskarte mit der Groundtruth-Disparitätskarte nicht, um die Leistung in Hinsicht eines kleineren lokalen Störeffektes zu bewerten. Andere ungewollte Störungen können dabei das Ergebnis verfälschen.

Im Bezug auf die Nutzung von augmentierten Daten haben die verwendeten Augmentationsmethoden zu unterschiedlich guten Ergebnissen geführt. Die Trainingsdaten der entwickelten Augmentationsmethode für 2D-Verdeckungen führte zu guten Ergebnissen, die mit den echten Daten vergleichbar waren. Im Vergleich dazu resultierte das Training mit den Daten der 3D-Variante in teilweise schlechterer Leistung. Die Daten mit künstlichen Regentropfen der verwendeten Methode waren nicht realistisch genug, um das Netz für echte Tropfen vorzubereiten. Die entwickelte Augmentationsmethode für künstlichen Nebel hat in Bezug auf echte Nebelbilder sogar zu einer Verschlechterung der Leistung des Netzwerkes geführt.

Die Ergebnisse dieser Arbeit zeigen, dass es möglich ist, neuronale Netzwerke für die Stereokorrespondenz unter umweltbedingten Störungen zu trainieren und verbesserte Leistung zu erreichen. Ausschlaggebend dafür sind repräsentative Trainingsdaten, die den Einfluss der Störeffekte auf die Bildinformationen widerspiegeln. Eine Steigerung der Leistung könnte durch weiteres Training, die Verwendung anderer Datensätze oder anderer Netzwerkarchitekturen erreicht werden und lässt Raum für weitere Untersuchungen.

8.2 Ausblick

Das für diese Arbeit ausgewählte Netzwerk wurde nicht speziell für den Umgang mit einem der betrachteten Störeffekte entwickelt. Für nebelige Bedingungen wäre es jedoch interessant, die Netzwerkarchitekturen der Arbeiten von Song et al. [2020] und Yao und Yu [2022] zu untersuchen. Die Arbeit von Yao und Yu [2022] mit dem Konzept des *fog volume* liefert vielversprechende Ergebnisse für künstliche Nebelbilder. Des Weiteren könnte die dabei einsetzbare unüberwachte Lernstrategie das Problem der fehlenden oder unzureichenden Groundtruth-Daten bei realen Aufnahmen in Nebel lösen. Datensätze wie der *SeeingThroughFog*-Datensatz könnten somit auch für das Training der Netze verwendet werden.
Im Hinblick auf Regentropfen und die Verdeckung durch Objekte existierten keine bekannten alternativen Ansätze. Hierbei könnte aber auch die Verwendung einer unüberwachten Lernstrategie untersucht werden, indem sie mit der in dieser Arbeit genutzten Netzwerkarchitektur kombiniert wird. Das Netzwerk könnte dafür eine Disparitätsschätzung anhand der verunreinige Bilder erstellen, die dann für die Erstellung der Rekonstruktion mit den „sauberen“ Bildern verwendet wird, womit dann der Fehler bestimmt wird. Das Netz könnte somit auf das Rekonstruieren der Szene ohne die Störeffekte trainiert werden. Als Ausgangspunkt kann die Arbeit von Zhang et al. [2022] dienen, die einen *soft-warping loss* entwickelt haben, der die Probleme des normalen *reconstruction loss* beheben soll, die in Abschnitt 7.1.2 beschrieben wurden.

In dieser Arbeit wurde deutlich, dass Unterschiede der Domänen verschiedener Datensätze großen Einfluss auf die Generalisierungsfähigkeit eines trainierten Netzes haben. Netzwerkarchitekturen wie das *Domain-invaraint Stereo Matching Network* von Zhang et al. [2020] versuchen den Einfluss von domänenspezifischen Eigenschaften für die Suche der Korrespondenzen zu reduzieren. Durch domänenübergreifende Normalisierung wird

die Verteilung von Merkmalen unterschiedlicher Datensätze, wodurch das Netzwerk weniger anfällig gegenüber Unterschieden von Kontrast, Rauschen, Mustern und ähnlichem ist. Nur auf künstlichen Daten trainiert, erreicht das Netzwerk für echte Datensätze zum Teil bessere Ergebnisse als andere Netzwerke, die auf den echten Daten trainiert wurden.

Nicht zuletzt kann diese Arbeit als Ausgangspunkt für fortgeführte Untersuchung der betrachteten oder weiterer nicht behandelter Störeffekten genutzt werden.

9 Namensnennung

Die folgenden 3D-Modelle, welche im Kontext dieser Arbeit verwendet wurden, sind unter der *CC BY 4.0*[1] lizenziert.

- *Crumpled paper* https://sketchfab.com/3d-models/crumpled-paper-f19ea7ce149c4b25a527abecec1da524 von **Darxkl05** (https://sketchfab.com/Darxkl05)
- *Leaf* https://sketchfab.com/3d-models/leaf-1ddbbe6f0ff84226a9d4d75a64b7fbeb von **sage.freeman** (https://sketchfab.com/sagefreeman)
- *Dry Leaf .::RAWscan::.* https://sketchfab.com/3d-models/leaf-1ddbbe6f0ff84226a9d von **Andrea Spognetta** (https://sketchfab.com/spogna)
- *Leaf* https://sketchfab.com/3d-models/leaf-42f3db9df1f14492852b11574daeeb3e von **dravid1852** (https://sketchfab.com/dravid1852)

[1] https://creativecommons.org/licenses/by/4.0/legalcode.de

Literaturverzeichnis

C. C. Aggarwal. *An Introduction to Neural Networks*, pages 1–52. Springer International Publishing, Cham, 2018. ISBN 978-3-319-94463-0. doi: 10.1007/978-3-319-94463-0_1. URL https://doi.org/10.1007/978-3-319-94463-0_1.

M. Awad und R. Khanna. *Deep Neural Networks*, pages 6–8, 127–147. Apress, Berkeley, CA, 2015. ISBN 978-1-4302-5990-9. doi: 10.1007/978-1-4302-5990-9_7. URL https://doi.org/10.1007/978-1-4302-5990-9_7.

J. Beyerer, F. Puente León, und C. Frese. *Bildaufnahmeverfahren*, pages 281–456. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016. ISBN 978-3-662-47786-1. doi: 10.1007/978-3-662-47786-1_7. URL https://doi.org/10.1007/978-3-662-47786-1_7.

M. Bijelic, T. Gruber, F. Mannan, F. Kraus, W. Ritter, K. Dietmayer, und F. Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11679–11689, 2020a. doi: 10.1109/CVPR42600.2020.01170.

M. Bijelic, T. Gruber, F. Mannan, F. Kraus, W. Ritter, K. Dietmayer, und F. Heide. Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020b.

Y. Boykov, O. Veksler, und R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001. doi: 10.1109/34.969114.

M. Brown, D. Burschka, und G. Hager. Advances in computational stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(8):993–1008, 2003. doi: 10.1109/TPAMI.2003.1217603.

Y. Cabon, N. Murray, und M. Humenberger. Virtual kitti 2, 2020.

C.-T. Chang. Role. https://github.com/ricky40403/ROLE, 2019. letzter Zugriff: 20.8.2023.

J.-R. Chang und Y.-S. Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, und Y. Wei. Deformable convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

U. Franke und S. Gehrig. *Stereosehen*, pages 378–382, 395–420. Springer Fachmedien Wiesbaden, Wiesbaden, 2015. ISBN 978-3-658-05734-3. doi: 10.1007/978-3-658-05734-3_22. URL https://doi.org/10.1007/978-3-658-05734-3_22.

A. Gaidon, Q. Wang, Y. Cabon, und E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4340–4349, 2016.

A. Geiger, P. Lenz, und R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.

H. Gotzig und G. O. Geduld. *LIDAR-Sensorik*, pages 317–334. Springer Fachmedien Wiesbaden, Wiesbaden, 2015. ISBN 978-3-658-05734-3. doi: 10.1007/978-3-658-05734-3_18. URL https://doi.org/10.1007/978-3-658-05734-3_18.

X. Guo, K. Yang, W. Yang, X. Wang, und H. Li. Group-wise correlation stereo network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019a.

X. Guo, K. Yang, W. Yang, X. Wang, und H. Li. Group-wise correlation stereo network, 2019b. URL https://arxiv.org/abs/1903.04025.

M. S. Hamid, N. A. Manap, R. A. Hamzah, und A. F. Kadmin. Stereo matching algorithm based on deep learning: A survey. *Journal of King Saud University - Computer and Information Sciences*, 34(5):1663–1673, 2022. ISSN 1319-1578. doi: https://doi.org/10.1016/j.jksuci.2020.08.011. URL https://www.sciencedirect.com/science/article/pii/S1319157820304493.

R. A. Hamzah und H. Ibrahim. Literature survey on stereo vision disparity map algorithms. *Journal of Sensors*, 2016:8742920, Dec 2015. ISSN 1687-725X. doi: 10.1155/2016/8742920. URL https://doi.org/10.1155/2016/8742920.

K. He, X. Zhang, S. Ren, und J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.

H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):328–341, 2008. doi: 10.1109/TPAMI.2007.1166.

G.-S. Hong und B.-G. Kim. A local stereo matching algorithm based on weighted guided image filtering for improving the generation of depth range images. *Displays*, 49:80–87, 2017. ISSN 0141-9382. doi: https://doi.org/10.1016/j.displa.2017.07.006. URL https://www.sciencedirect.com/science/article/pii/S0141938217300276.

A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, und A. Bry. End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Y. Kokubo, S. Asada, H. Maruyama, M. Koide, K. Yamamoto, und Y. Suetsugu. Removing raindrops from a single image using synthetic data. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 2081–2088, 2021. doi: 10.1109/ICPR48806.2021.9412888.

V. Kolmogorov. *Graph Based Algorithms for Scene Reconstruction from Two or More Views*. PhD thesis, USA, 2004. AAI3114475.

M. Kutila, P. Pyykönen, H. Holzhüter, M. Colomb, und P. Duthon. Automotive lidar performance verification in fog and rain. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1695–1701, 2018. doi: 10.1109/ITSC.2018.8569624.

A. Li, Z. Yuan, Y. Ling, W. Chi, S. Zhang, und C. Zhang. Unsupervised occlusion-aware stereo matching with directed disparity smoothing. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7457–7468, 2022. doi: 10.1109/TITS.2021.3070403.

D. Liu und R. Klette. Fog effect for photography using stereo vision. *The Visual Computer*, 32(1):99–109, Jan 2016. ISSN 1432-2315. doi: 10.1007/s00371-014-1058-7. URL https://doi.org/10.1007/s00371-014-1058-7.

N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, und T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene

flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016a. URL http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16. arXiv:1512.02134.

N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, und T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4040–4048, 2016b. doi: 10.1109/CVPR.2016.438.

M. Menze und A. Geiger. Object scene flow for autonomous vehicles. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

M. Menze, C. Heipke, und A. Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015.

M. Menze, C. Heipke, und A. Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 2018.

A. Newell, K. Yang, und J. Deng. Stacked hourglass networks for human pose estimation. In B. Leibe, J. Matas, N. Sebe, und M. Welling, editors, *Computer Vision – ECCV 2016*, pages 483–499, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46484-8.

J. Pang, W. Sun, J. S. Ren, C. Yang, und Q. Yan. !g for stereo matching. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

M. Poggi, F. Tosi, K. Batsos, P. Mordohai, und S. Mattoccia. On the synergies between machine learning and binocular stereo for depth estimation from images: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5314–5334, 2021.

J. Qiu, J. Liu, und Y. Shen. Computer vision technology based on deep learning. In *2021 IEEE 2nd International Conference on Information Technology, Big Data and Artificial Intelligence (ICIBA)*, volume 2, pages 1126–1130, 2021. doi: 10.1109/ICIBA52610.2021.9687873.

D. Scharstein, R. Szeliski, und R. Zabih. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. In *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, pages 131–140, 2001. doi: 10.1109/SMBV.2001.988771.

Z. Shi, N. Fan, D.-Y. Yeung, und Q. Chen. Stereo waterdrop removal with row-wise dilated attention. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3829–3836, 2021. doi: 10.1109/IROS51168.2021.9636216.

T. Song, Y. Kim, C. Oh, H. Jang, N. Ha, und K. Sohn. Simultaneous deep stereo matching and dehazing with feature attention. *International Journal of Computer Vision*, 128(4):799–817, Apr 2020. ISSN 1573-1405. doi: 10.1007/s11263-020-01294-2. URL https://doi.org/10.1007/s11263-020-01294-2.

J. Žbontar und Y. LeCun. Stereo matching by training a convolutional neural network to compare image patches. *J. Mach. Learn. Res.*, 17(1):2287–2318, jan 2016. ISSN 1532-4435.

Y. Wang, G. Wang, C. Chen, und Z. Pan. Multi-scale dilated convolution of convolutional neural network for image denoising. *Multimedia Tools and Applications*, 78(14):19945–19960, Jul 2019. ISSN 1573-7721. doi: 10.1007/s11042-019-7377-y. URL https://doi.org/10.1007/s11042-019-7377-y.

X. Xiang, M. Zhang, G. Li, Y. He, und Z. Pan. Real-time stereo matching based on fast belief propagation. *Machine Vision and Applications*, 23(6):1219–1227, Nov 2012. ISSN 1432-1769. doi: 10.1007/s00138-011-0405-1. URL https://doi.org/10.1007/s00138-011-0405-1.

G. Xu, J. Cheng, P. Guo, und X. Yang. Attention concatenation volume for accurate and efficient stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12981–12990, June 2022.

H. Xu und J. Zhang. Aanet: Adaptive aggregation network for efficient stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

B. Yan, C. Ma, B. Bare, W. Tan, und S. Hoi. Disparity-aware domain adaptation in stereo image restoration. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13176–13184, 2020. doi: 10.1109/CVPR42600.2020.01319.

G. Yang, X. Song, C. Huang, Z. Deng, J. Shi, und B. Zhou. Drivingstereo: A large-scale dataset for stereo matching in autonomous driving scenarios. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

C. Yao und L. Yu. Foggystereo: Stereo matching with fog volume representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13043–13052, June 2022.

K. Yoneda, N. Suganuma, R. Yanase, und M. Aldibaja. Automated driving recognition technologies for adverse weather conditions. *IATSS Research*, 43(4):253–262, 2019. ISSN 0386-1112. doi: https://doi.org/10.1016/j.iatssr.2019.11.005. URL https://www.sciencedirect.com/science/article/pii/S0386111219301463.

S. Zang, M. Ding, D. Smith, P. Tyler, T. Rakotoarivelo, und M. A. Kaafar. The impact of adverse weather conditions on autonomous vehicles: How rain, snow, fog, and hail affect the performance of a self-driving car. *IEEE Vehicular Technology Magazine*, 14 (2):103–111, 2019. doi: 10.1109/MVT.2019.2892497.

B. Zhang. *Machine Learning and Visual Perception*. De Gruyter, Berlin, Boston, 2020. ISBN 9783110595567. doi: doi:10.1515/9783110595567. URL https://doi.org/10.1515/9783110595567.

F. Zhang, V. Prisacariu, R. Yang, und P. H. Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

F. Zhang, X. Qi, R. Yang, V. Prisacariu, B. Wah, und P. Torr. Domain-invariant stereo matching networks. In A. Vedaldi, H. Bischof, T. Brox, und J.-M. Frahm, editors, *Computer Vision – ECCV 2020*, pages 420–439, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58536-5.

H. Zhang, L.-P. Chau, und D. Wang. Soft warping based unsupervised domain adaptation for stereo matching. *IEEE Transactions on Multimedia*, 24:3835–3846, 2022. doi: 10.1109/TMM.2021.3108900.

K. Zhang, D. Li, W. Luo, und W. Ren. Dual attention-in-attention model for joint rain streak and raindrop removal. *IEEE Transactions on Image Processing*, 30:7608–7619, 2021. doi: 10.1109/TIP.2021.3108019.

L. Zhang, L. Zhang, X. Mou, und D. Zhang. Fsim: A feature similarity index for image quality assessment. *IEEE Transactions on Image Processing*, 20(8):2378–2386, 2011. doi: 10.1109/TIP.2011.2109730.

Y. Zhang, A. Carballo, H. Yang, und K. Takeda. Perception and sensing for autonomous vehicles under adverse weather conditions: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing*, 196:146–177, 2023. ISSN 0924-2716. doi: https://doi.org/10.1016/j.isprsjprs.2022.12.021. URL https://www.sciencedirect.com/science/article/pii/S0924271622003367.

K. Zhou, X. Meng, B. Cheng, und C. Yáñez Márquez. Review of stereo matching algorithms based on deep learning. *Intell. Neuroscience*, 2020, jan 2020. ISSN 1687-5265. doi: 10.1155/2020/8562323. URL https://doi.org/10.1155/2020/8562323.

J. Žbontar und Y. LeCun. Computing the stereo matching cost with a convolutional neural network. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1592–1599, 2015. doi: 10.1109/CVPR.2015.7298767.

A Anhang

Abbildung A.1: Bildpaare der aufgenommenen ZED-Sequenz mit Regentropfen.

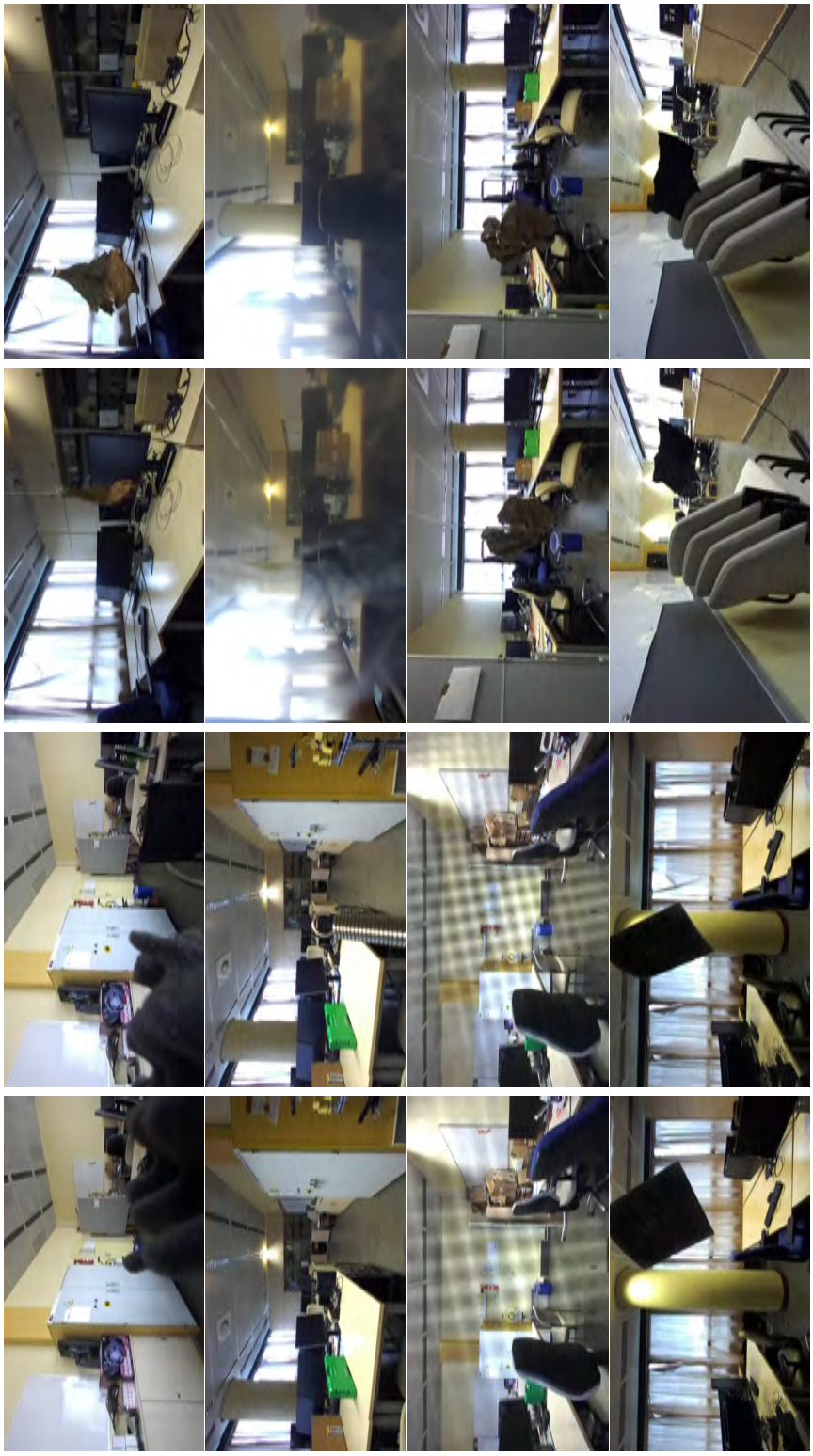
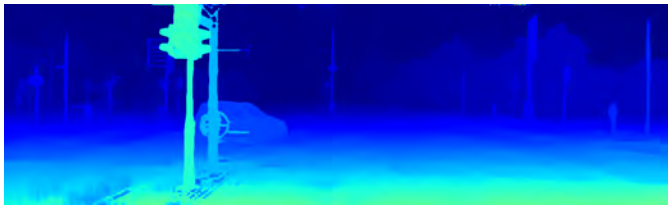


Abbildung A.2: Bildpaare der aufgenommenen ZED-Sequenz mit Verdeckungen.

(a) RGB Bild (links) | 000021_10 (Training)

(b) Disparitätskarte des Netzwerkes | 000021_10 (Training) | (normalisiert)

(c) RGB Bild (links) | 000043_10(Training)



(d) Disparitätskarte des Netzwerkes | 000043_10 (Training) | (normalisiert)

(e) RGB Bild (links) | 0000169_10 (Training)

(f) Disparitätskarte des Netzwerkes | 0000169_10 (Training) | (normalisiert)

Abbildung A.3: Ergebnisse des trainieren Grundnetzwerk auf dem KITTI 2015 Datensatz

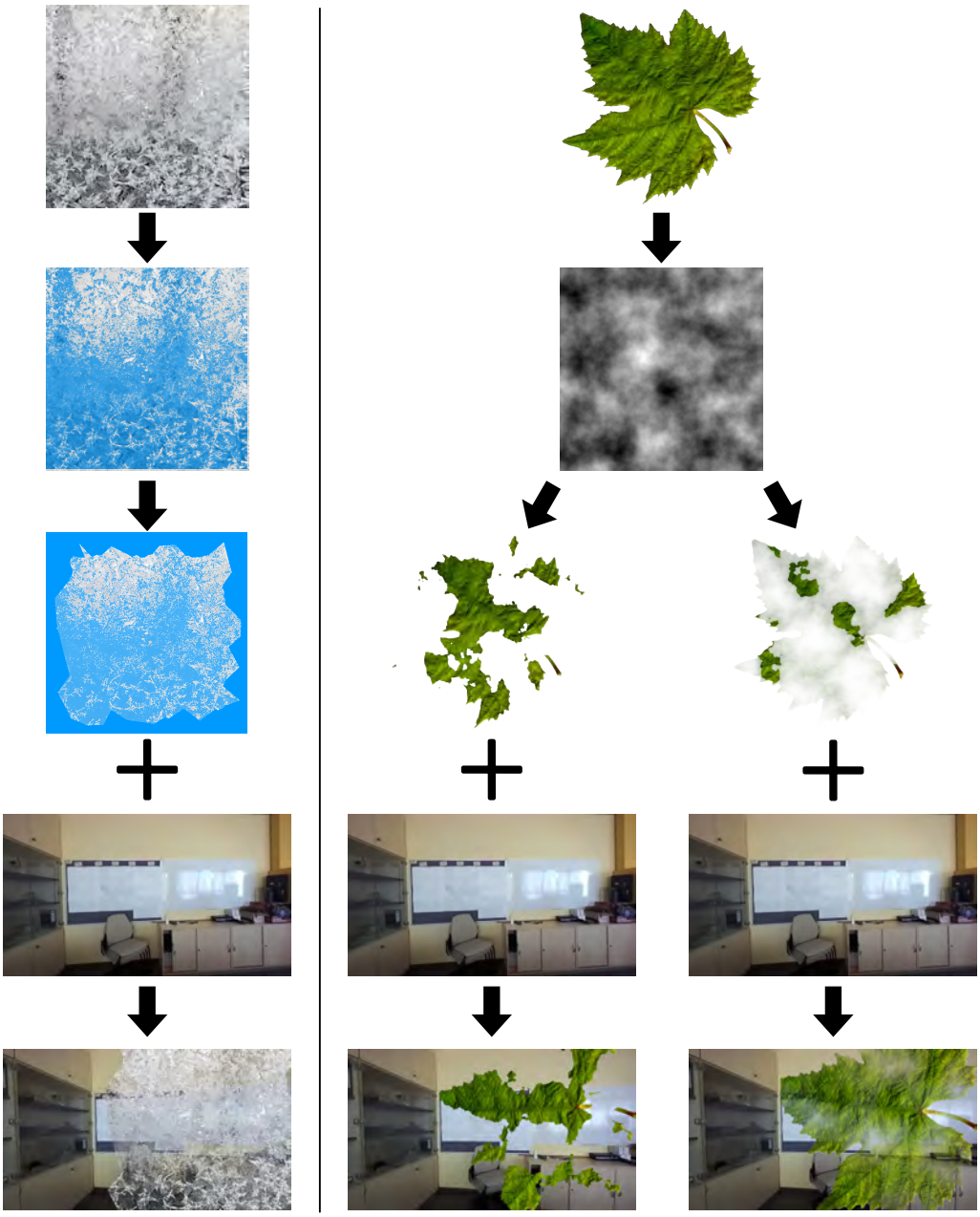
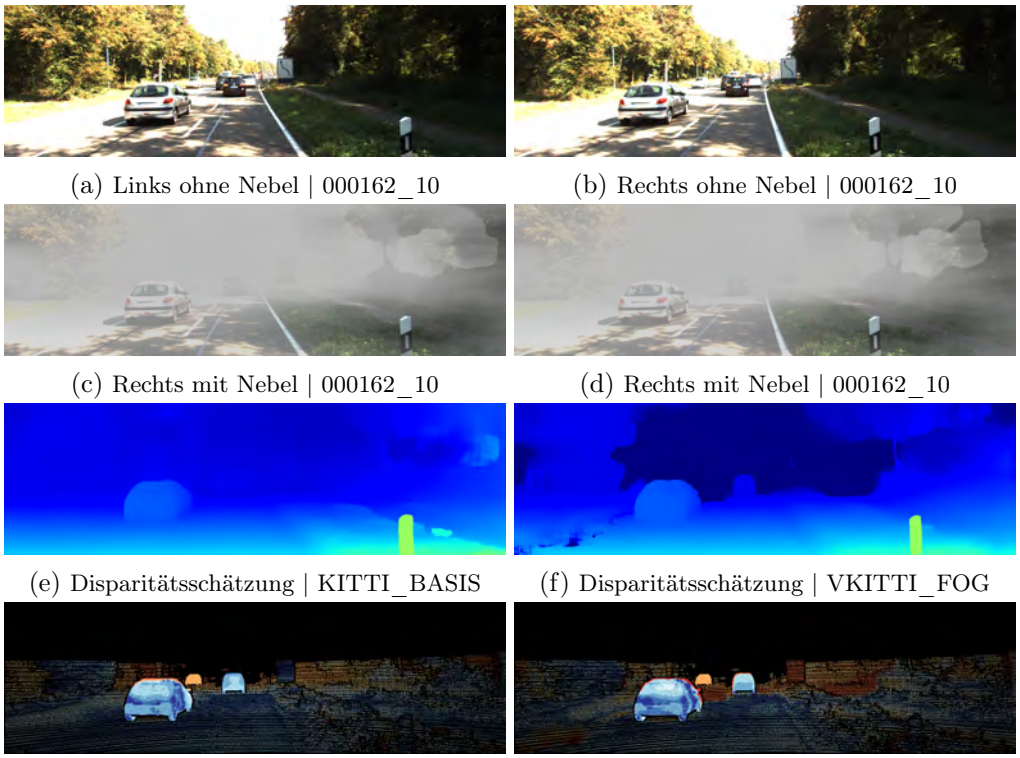


Abbildung A.4: Ablauf der 2D Verdeckungs-Augmentation. Links: Verdeckung mit Eis-Textur. Rechts: Verdeckung mit anderen Formen.



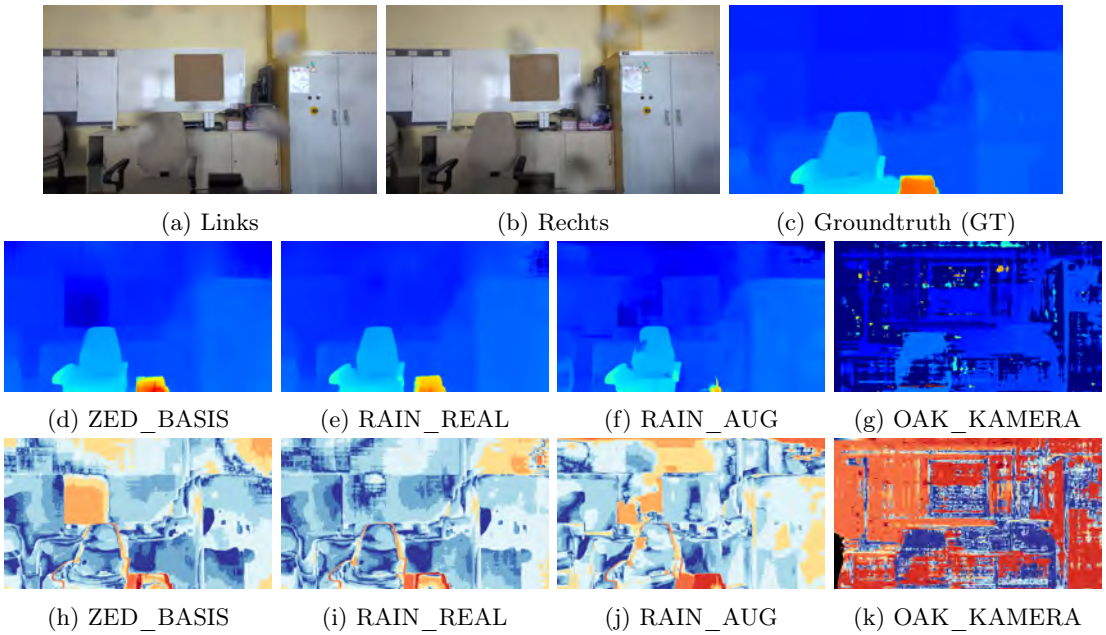
(a) Links ohne Nebel | 000162_10 (b) Rechts ohne Nebel | 000162_10

(c) Rechts mit Nebel | 000162_10 (d) Rechts mit Nebel | 000162_10

(e) Disparitätsschätzung | KITTI_BASIS (f) Disparitätsschätzung | VKITTI_FOG

(g) Fehlerkarte | KITTI_BASIS (h) Fehlerkarte | VKITTI_FOG

Abbildung A.5: Ergebnis des KITTI_BASIS und VKITTI_FOG-Netzes für die Szene 000162_-10 des KITTI15-Datensatzes mit Nebel.

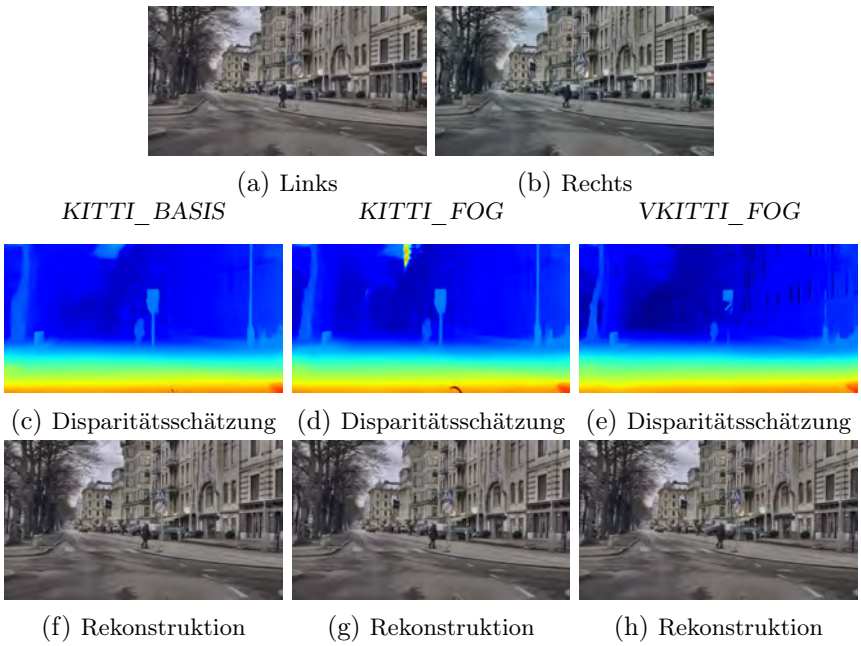


(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) RAIN_REAL (f) RAIN_AUG (g) OAK_KAMERA

(h) ZED_BASIS (i) RAIN_REAL (j) RAIN_AUG (k) OAK_KAMERA

Abbildung A.6: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_RAIN Steckplatz 1.



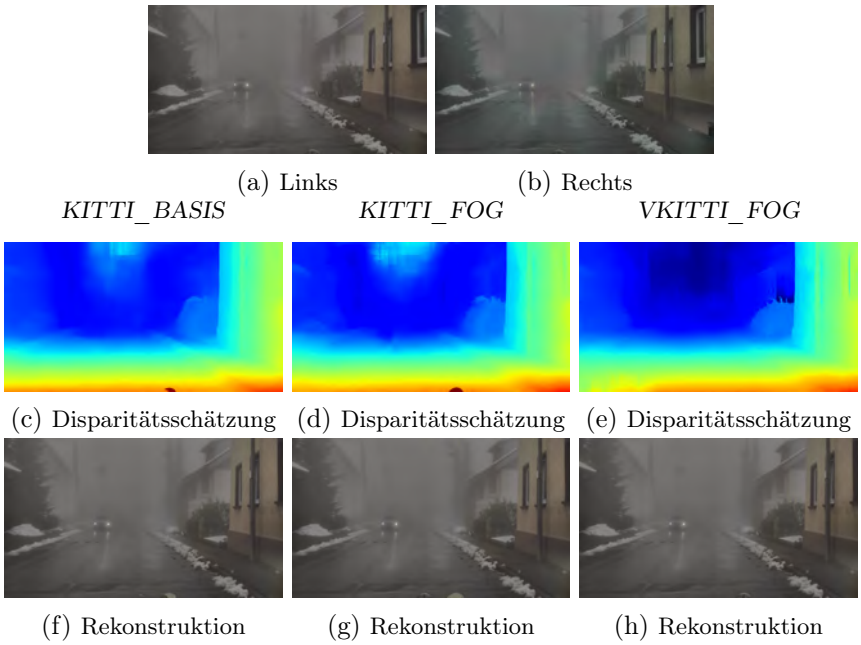
(a) Links (b) Rechts

KITTI_BASIS *KITTI_FOG* *VKITTI_FOG*

(c) Disparitätsschätzung (d) Disparitätsschätzung (e) Disparitätsschätzung

(f) Rekonstruktion (g) Rekonstruktion (h) Rekonstruktion

Abbildung A.7: Ergebnisse von KITTI_BASIS (1. Spalte), KITTI_FOG (2. Spalte), VKITTI_-FOG (3. Spalte) für die Szene 2018-12-22_14-52-12_02200 des STF-Datensatzes.



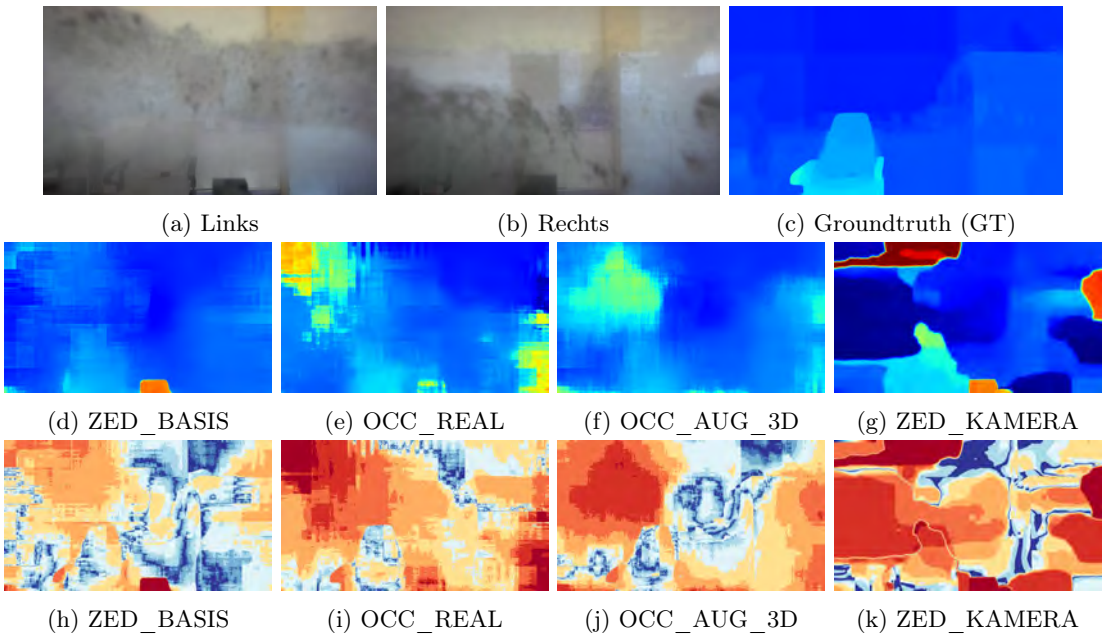
(a) Links (b) Rechts

KITTI_BASIS *KITTI_FOG* *VKITTI_FOG*

(c) Disparitätsschätzung (d) Disparitätsschätzung (e) Disparitätsschätzung

(f) Rekonstruktion (g) Rekonstruktion (h) Rekonstruktion

Abbildung A.8: Ergebnisse von KITTI_BASIS (1. Spalte), KITTI_FOG (2. Spalte), VKITTI_-FOG (3. Spalte) für die Szene 2018-10-29_15-15-15_0151 des STF-Datensatzes.

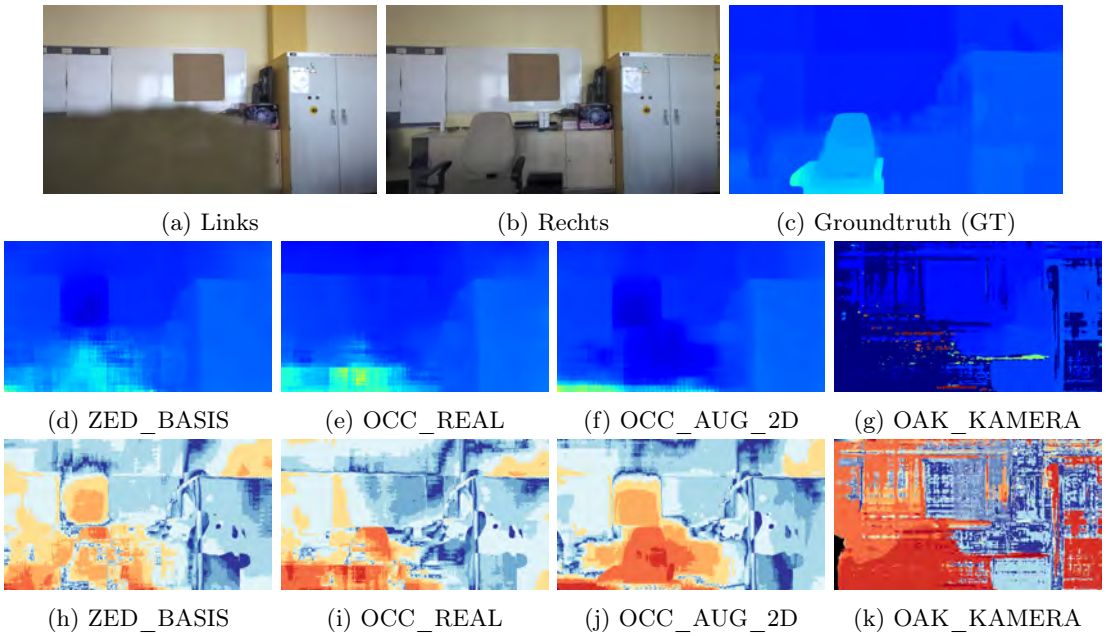


(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_REAL (f) OCC_AUG_3D (g) ZED_KAMERA

(h) ZED_BASIS (i) OCC_REAL (j) OCC_AUG_3D (k) ZED_KAMERA

Abbildung A.9: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_ICE Steckplatz 3.

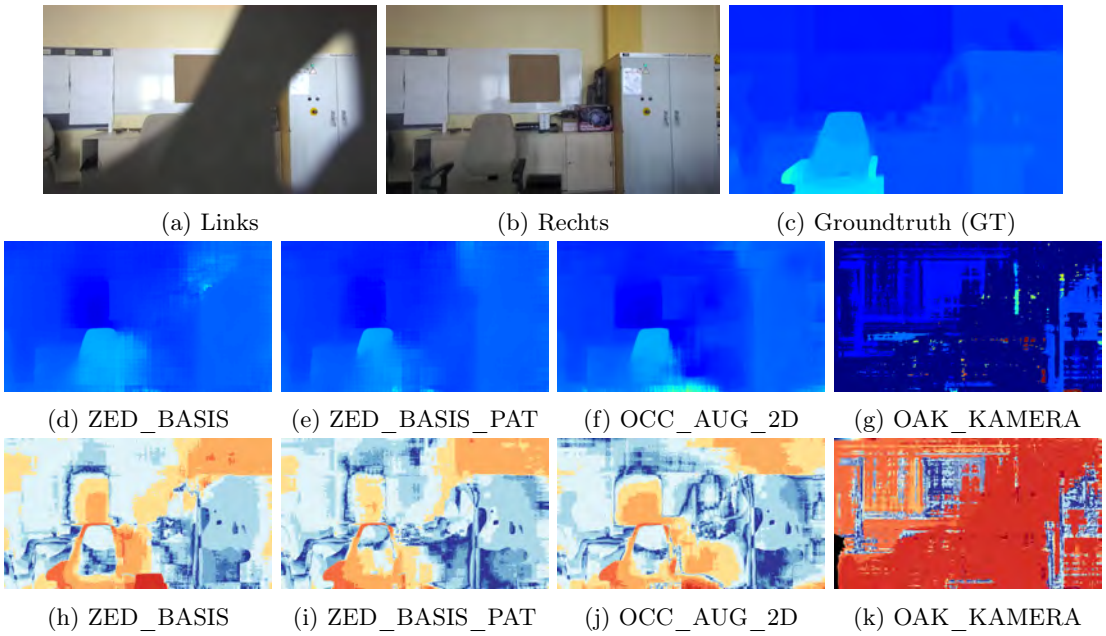


(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_REAL (f) OCC_AUG_2D (g) OAK_KAMERA

(h) ZED_BASIS (i) OCC_REAL (j) OCC_AUG_2D (k) OAK_KAMERA

Abbildung A.10: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_TAPE Steckplatz 1.

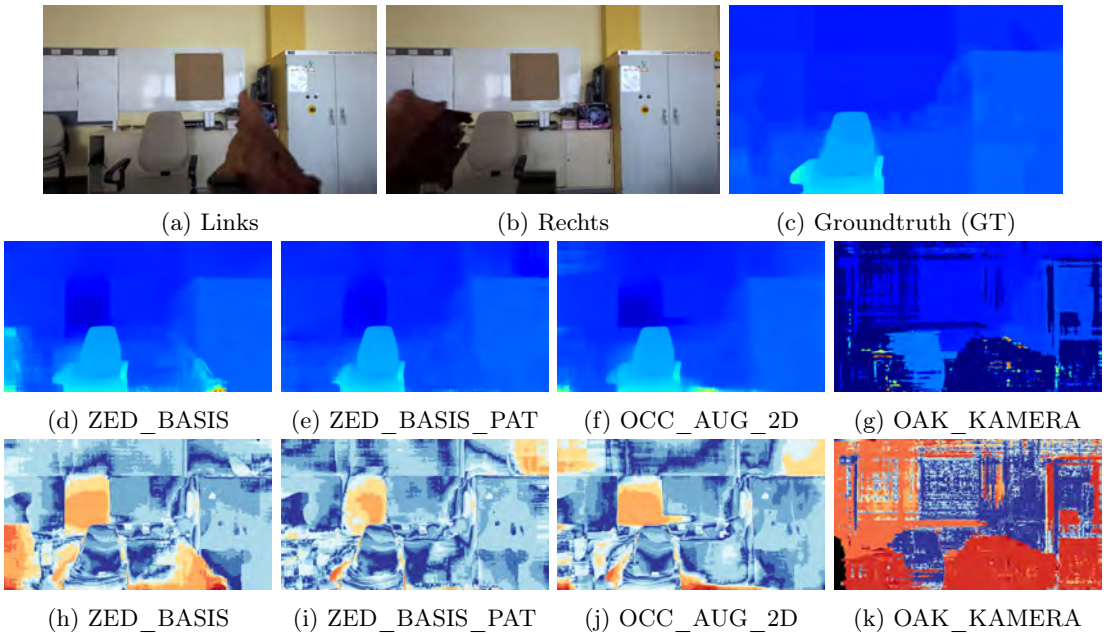


(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) ZED_BASIS_PAT (f) OCC_AUG_2D (g) OAK_KAMERA

(h) ZED_BASIS (i) ZED_BASIS_PAT (j) OCC_AUG_2D (k) OAK_KAMERA

Abbildung A.11: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_PAPER Steckplatz 1.

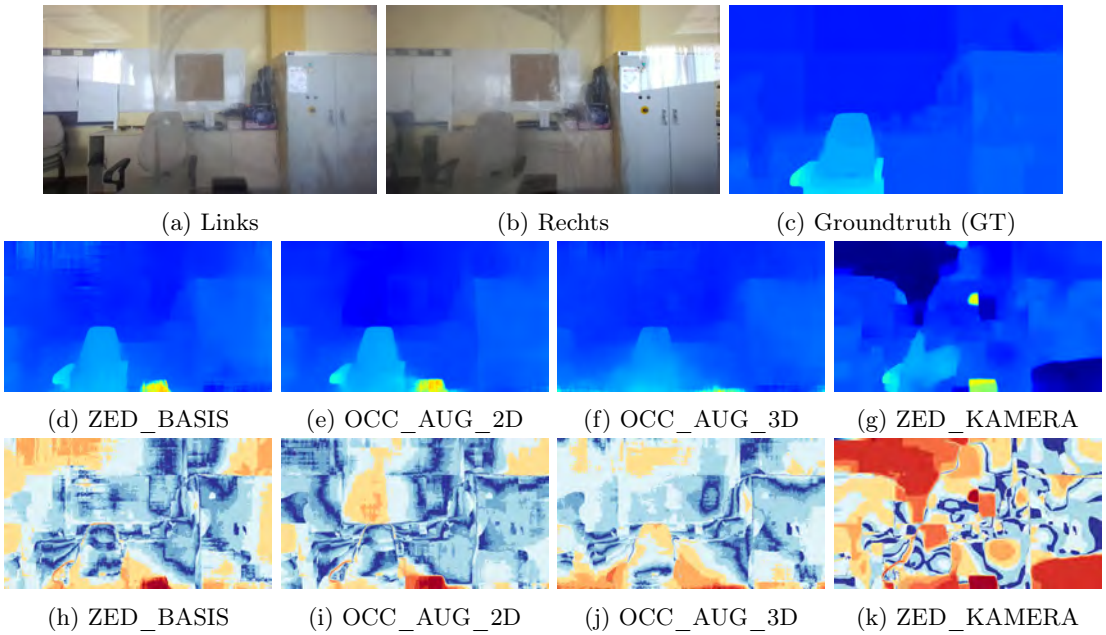


(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) ZED_BASIS_PAT (f) OCC_AUG_2D (g) OAK_KAMERA

(h) ZED_BASIS (i) ZED_BASIS_PAT (j) OCC_AUG_2D (k) OAK_KAMERA

Abbildung A.12: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_LEAF Szene 2.



(a) Links (b) Rechts (c) Groundtruth (GT)

(d) ZED_BASIS (e) OCC_AUG_2D (f) OCC_AUG_3D (g) ZED_KAMERA

(h) ZED_BASIS (i) OCC_AUG_2D (j) OCC_AUG_3D (k) ZED_KAMERA

Abbildung A.13: Linkes (a) und rechtes (b) Bild, Groundtruth (c), Disparitätsschätzungen der Netzwerke (d,e,f,g) und Fehlerkarten (h,i,j,k) für ZED_SCENA_TRANSP Steckplatz 5.

Erklärung zur selbstständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original