NICK SCHNEIDER

Deep Fusion of Camera and LIDAR

KIT Scientific Publishing

Nick Schneider

Deep Fusion of Camera and LIDAR

Schriftenreihe
Institut für Mess- und Regelungstechnik,
Karlsruher Institut für Technologie (KIT)

Band 052

Eine Übersicht aller bisher in dieser Schriftenreihe erschienenen Bände finden Sie am Ende des Buchs.

Deep Fusion of Camera and LIDAR

by
Nick Schneider

KIT Scientific Publishing

Karlsruher Institut für Technologie
Institut für Mess- und Regelungstechnik

Deep Fusion of Camera and LIDAR

Zur Erlangung des akademischen Grades eines Doktors der Ingenieurwissenschaften von der KIT-Fakultät für Maschinenbau des Karlsruher Instituts für Technologie (KIT) genehmigte Dissertation

von Nick Schneider, M.Sc.

Tag der mündlichen Prüfung: 4. Oktober 2023
Referent: Prof. Dr.-Ing. Christoph Stiller
Korreferent: Prof. Dr. rer. nat. Markus Enzweiler

Impressum

Scientific Publishing

Karlsruher Institut für Technologie (KIT)
KIT Scientific Publishing
Straße am Forum 2
D-76131 Karlsruhe

www.ksp.kit.edu

Print on Demand 2024 – Gedruckt auf FSC-zertifiziertem Papier

ISSN 1613-4214
ISBN 978-3-7315-1361-2
DOI 10.5445/KSP/1000169933

Kurzfassung

Ein robustes und genaues Umgebungsmodell ermöglicht es autonomen Fahrzeugen sicher in der Welt zu navigieren. Ein vielversprechender Ansatz um ein solches Modell zu erzielen, ist die Fusion von multimodalen Sensoren wie Kamera und LIDAR. Speziell durch den Einsatz von neuronalen Netzen konnten hier in den letzten Jahren vielversprechende Ergebnisse erzielt werden.

Es bestehen jedoch weiterhin Herausforderungen in der Fusion multimodaler Daten. Zunächst wird eine genaue Kalibrierung benötigt, um die Daten in einem gemeinsamen Koordinatensystem zu beschreiben. Des Weiteren muss eine geeignete Repräsentation der Daten gefunden werden, die es ermöglicht, diese zu fusionieren. Dies ist herausfordernd, da LIDAR-Daten meist deutlich weniger dicht als die hochaufgelösten Kamerabilder sind. Zuletzt werden große Mengen an Daten benötigt, speziell beim Einsatz neuronaler Netze.

In dieser Dissertation werden drei Methoden und dazugehörige Experimente vorgestellt, die diese Herausforderungen adressieren. Das erste ist ein Verfahren zur Kalibrierung von LIDAR und Kamera, welches CNNs nutzt um alle gängigen Kalibrierschritte zu ersetzen. Experimente zeigen, dass CNNs tatsächlich in der Lage sind, eine Transformation zwischen LIDAR und Kamera effizient und ohne menschliche Interaktion zu schätzen. Zusätzlich werden zwei Methoden zur Vervollständigung von Tiefendaten vorgestellt. Diese erlauben, LIDAR und Kamera Daten in einer gemeinsamen Repräsentation zu fusionieren. Die erste Methode basiert auf einer modellbasierten Energieminimierung und ist auch ohne Trainingsdaten generisch einsetzbar. Die zweite Methode hingegen nutzt neuronale Netze zur Vervollständigung wodurch hohe Genauigkeiten erzielt werden. Abschließend wird ein Tiefen-Datensatz basierend auf KITTI vorgestellt. Der Datensatz besteht aus 93.000 RGB Bildern und dazugehörigen Tiefenkarten und dient zu Training und Evaluation von Algorithmen zur Tiefenschätzung.

Abstract

Generating a robust and accurate environment model enables autonomous vehicles to move safely in the world. A promising approach to achieve such a model is to fuse multimodal sensors such as camera and laser scanner. Thanks to the advance of neural networks, in the last few years promising results were achieved.

However, there remain major challenges in fusing multi-modal data. First, a convenient yet accurate calibration is required to describe the measurements in a common coordinate system. Second, a well-suited representation to fuse the data needs to be found, which is especially challenging as LIDAR data is typically significantly sparser than high resolution camera images. Finally, a large amount of training and evaluation data is required, especially if LIDAR and camera data are processed with deep neural networks.

In this dissertation, those challenges are addressed by proposing three different methods, each accompanied by associated experiments. First, a method for LIDAR-to-camera calibration is presented which is the first to use a deep neural network to replace all common calibration steps. The conducted experiments show that CNNs are indeed able to efficiently estimate a transformation between two sensors without any human interaction. Furthermore, two different depth completion approaches are presented, which allow processing depth measurements in the same space as the RGB image. Both methods perform well on various domains. The first method is based on a model-based energy minimization and can be applied generically without the need of training data. The second approach is leveraging neural networks for the completion and is slightly more accurate. Finally, a large-scale depth dataset is introduced which consists of 93k RGB and depth images and can be used to train and evaluate camera and LIDAR-based algorithms. This dataset is targeted, but not restricted, to the use of deep neural networks and is based on the well-known KITTI dataset.

Vorwort

Die hier vorliegende Arbeit entstand während meiner Zeit bei der Mercedes-Benz AG unter Betreuung von Herrn Prof. Christoph Stiller am Institut für Mess- und Regelungstechnik. Zunächst möchte ich mich bei Herrn Prof. Stiller für die Betreuung der Arbeit bedanken. Ihre schnelle Auffassungsgabe hat einen stets hochwertigen fachlichen Austausch ermöglicht, wodurch diese Arbeit stetig an Qualität gewann.

Betrachte ich, wie viel Arbeit und Nerven meine Kollegen, Freunde, Familie und schließlich ich selbst in dieses Werk investiert haben, erfüllt es mich mit großer Freude und unendlich viel Dankbarkeit nun diese Worte schreiben zu dürfen.

Ein großer Dank gilt Uwe Franke für das Ermöglichen der Arbeit, die lockere und professionelle Atmosphäre und die Freiheit und Impulse in der Forschung. Ich danke dem ganzen Team *Image Understanding* für die Diskussionen sowohl während als auch nach der Arbeit. Hier sind viele enge Freundschaften entstanden, die mich durch meine Zeit als Doktorand getragen haben. Im Speziellen danke ich Marius Cordts für das stetige neu motivieren, die vielen Korrekturen und das oft so notwendige Getränk am Abend. Weiterhin danke ich den zahlreichen Lektoren: Markus Enzweiler, Felix Abrecht, Eike Rehder, Lukas Schneider, Jonas Uhrig, Niklas Hanselmann und Martin Lauer. Bei meinen Freunden bedanke ich mich für den moralischen Beistand. Euch war es, im Gegensatz zu mir, immer klar, dass ich diese Arbeit fertigstelle. Besonderer Dank gilt meiner Familie und meinen Eltern, die immer an mich geglaubt haben, ohne je etwas von mir zu verlangen.

Ganz besonders danke ich meiner Frau Lisa, meinem Sohn Finley und meiner Tochter Emma Faye, welche unglaublich viel Geduld, verpasste Urlaube, einsame Abende und einen gestressten Ehemann/Papa auf sich genommen haben. Ohne Euch wäre die "Dottoabait" weniger lustig und just unmöglich gewesen.

Bruchsal, im März 2024 *Nick Schneider*

Contents

Notation and Symbols

Acronyms

ACC	Adaptive Cruise Control
CMOS	Complementary Metal-Oxide-Semiconductor
CNN	Convolutional Neural Network
DOF	Degree of Freedom
ELU	Exponential Linear Unit
FMCW	Frequency-Modulated Continuous Wave
FOV	Field of View
GNSS	Global Navigation Satellite Systems
GPU	Graphics Processing Unit
iMAE	Inverse Mean Absolute Error
iRMSE	Inverse Root Mean Squared Error
IoU	Intersection-over-Union
IR	Infrared
KITTI	Karlsruhe Institute of Technology and Toyota Technological Institute at Chicago (dataset, [29])
LIDAR	Light Detection and Ranging
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MEMS	Micro-Electromechanical Systems
MRF	Markov Random Field
MVS	Multi-View Stereo
NiN	Network in Network [69]
OPA	Optical Phased Array

RELU	Rectified Linear Unit
RADAR	Radio Detection and Ranging
RMSE	Root Mean Squared Error
SfM	Structure-from-Motion
SGD	Stochastic Gradient Descent
SGDC	Semantically Guided Depth Completion
SGM	Semi-Global Matching
SICNN	Sparsity Invariant CNN
ToF	Time of Flight

General Notation

Scalars	Regular (Greek) lower case	x, y, δ, σ
Vectors	Bold (Greek) lower case	$\mathbf{x}, \mathbf{y}, \boldsymbol{\delta}, \boldsymbol{\sigma}$
Matrices	Bold (Greek) upper case	$\mathbf{A}, \mathbf{B}, \boldsymbol{\Delta}, \boldsymbol{\Sigma}$
Sets	Calligraphic upper case	$\mathcal{M}, \mathcal{I}$

Generic Mathematical Symbols

ε	A small constant
$*$	Convolution operator
$\odot$	Element-wise multiplication operator
$\hat{x}$	An estimate

Neural Networks

w	Weight
b	Bias
α	Learning rate
β_1	Adam Solver: Decay rate for the first moment
β_2	Adam Solver: Decay rate for the second moment

LIDAR Camera Calibration

$\bar{x}$	Point in homogeneous coordinates
t	Translation $\in \mathbb{R}^3$
R	Rotation matrix $\in \mathbb{R}^{3\times3}$
T	Transformation $\in SE(3)$
T_{decalib}	Decalibrated transformation matrix
$\hat{T}_{\text{decalib}}$	Estimate of T_{decalib}
σ	Dual-quaternion
p	Real part of dual-quaternion
q	Dual part of dual-quaternion

Semantically Guided Depth Completion

$E()$	Energy function
$E_{una}()$	Unery energy function
$E_{pair}()$	Pairwise energy function
$E_o()$	Outlier energy function
θ	Plane parameters
$\mathbf{o}$	Outlier flags
$\mathbf{c}$	Coplanarity flags
$\mathbf{z}$	Sparse depth measurements
I	Image pixels
$\mathscr{M}$	Pixels with depth measurement
$w_{i,j}$	Pairwise weights
$D_g()$	Geodesic distance
$c()$	Cost function
$c_E()$	Edge cost function
$c_S()$	Semantic cost function
$c_T()$	Cost function for path length
$n_i \in \mathscr{M}$	Nearest pixel with measurement
$\mathscr{V}$	Voronoi cells

Sparsity Invariant CNNs

$\boldsymbol{o}$	Mask of observed depth measurements

Large-Scale Depth Dataset

D	Depth map
M	Valid pixel mask
d	Disparity
o	KITTI outliers
δ_j	Inlier ratios

1 Introduction

This dissertation addresses open challenges in the field of autonomous driving by fusing LIDAR and camera data leveraging deep neural networks. This chapter will first introduce the goals and motivation of this research. To that end, we start with the history and current state of autonomous driving in Section 1.1. We continue with an overview of sensors in autonomous vehicles (Section 1.2) and discuss how to fuse sensor data in Section 1.3. More recently, multimodal data fusion methods leveraging deep neural networks have been proposed. An overview and discussion on them is presented in Section 1.4. Finally, the goals of this dissertation and its outline are provided in Section 1.5 and Section 1.6, respectively.

1.1 On The Road to Autonomous Driving

Mobility has always been an integral part of our society. Especially the invention of the automobile in the 19th century has had a drastic impact on the everyday life of humans. Since then cars and other vehicles have shaped the way of transportation. Today, a significant amount of time is spent in cars [57], making them an important space of living.

However, the increase in vehicle usage is constantly putting a burden on our traffic infrastructure, resulting in significant road congestion and full parking lots. Every day, road traffic accidents cause thousands of fatalities [82] and even more injuries. And with more vehicles on the road, the impact on the environment has increased as well. Large amounts of energy and resources for infrastructure, production, and fueling of vehicles are required.

Motivated by this development, car manufacturers, big technological companies and many startups compete in bringing a groundbreaking solution to the market—*autonomous driving*.

Autonomous driving is expected to improve the quality and availability of mobility in many aspects. It will allow to optimize the time spent in vehicles by enabling the passenger to work, leisure or sleep. It will enable elderly or impaired people to be picked up directly from home and be

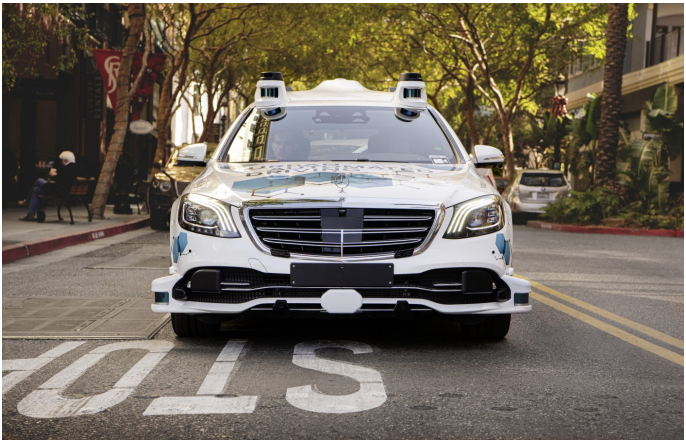


Figure 1.1: A vehicle equipped with sensors for autonomous driving. Many autonomous driving projects such as the one by the Mercedes-Benz AG and the Robert Bosch GmbH rely on a variety of different sensors such as camera, LIDAR, RADAR and ultrasonic. Source: Bosch Press [4]

dropped at their target. Furthermore, traffic accidents could be drastically reduced, resulting in fewer fatalities per year. However, autonomous vehicles are not only a technological improvement but can also revolutionize mobility itself. They enable complex shared mobility services, reducing the need of privately owned vehicles. *Shared Mobility* would increase the average number of passengers per car. The accompanying reduction of the total number of vehicles on the roads could significantly relieve both the traffic infrastructure and the environment.

In recent years, autonomous driving has been moving from the realm of science-fiction to reality, especially due to increased processing power and the advances in machine learning. Yet, many problems are still unsolved and autonomous driving remains one of the most challenging technological problems of the 21st century.

What makes the task of autonomous driving so complex? Political, legal, and sociocultural acceptance require an autonomous vehicle not to be only good, but almost perfect. Safe and accident free driving are of the highest priority. To ensure safe and reliable maneuvering in complex traffic, autonomous vehicles need to accurately perceive and understand their environment. A multitude of static and dynamic objects have to be detected

	Camera	LIDAR	RADAR	Ultrasonic
Price	$$	$$$$	$$$	$
Range	60-100m*	< 250 m	< 250 m	< 6 m
Distance Accuracy	+	++++	++	+
Height Accuracy	++	++++	-	-
Angular Resolution	++++	++	+	-
Semantic Understanding	++++	+++	++	+
Velocity Estimation	++	+	++++	+
Performance at Night	++	++++	++++	++++
Performance in Rain	+	++	+++	++

* The range of camera systems strongly depends on the number of pixels per degree and, if used in a stereo setup, from the stereo baseline. Typical ranges are between 60 m and 100 m.

Table 1.1: Comparison of different automotive sensors. The table should only be used for a general overview. Units such as '$' and '+' are in no means related to any real number and should not be used to compare sensors directly (e.g. a LIDAR sensor is not twice as expensive as a camera although it has twice as many '$' signs).

and interpreted in terms of their geometry, motion, semantics, and intent. A particular challenge arises from highly dynamical situations that put strong requirements on detection distance, accuracy, and latency. The performance of current perception systems do not meet those requirements and hence prevent today's self-driving cars from operating in higher-speed scenarios. As a way out, in this dissertation advanced algorithms are investigated which are coupled with precise long range and high-resolution sensors, that are presented in the following section.

1.2 The Sensors That Drive Autonomous Vehicles

In this section the most common sensors for autonomous driving are introduced. While there are a lot of different sensors used in autonomous vehicles, such as gyroscopes, Global Navigation Satellite Systems (GNSS), rain sensors and others, this section focuses on sensors for environment perception.

A subjective summary of advantages and disadvantages is pictured in Table 1.1. The table shows why many autonomous sensor setups rely on a variety of complementary sensors.

Ultrasonic sensors Providing functions for automated and assisted parking, ultrasonic sensors are already an essential part of today's vehicles. The main advantages are relatively small costs due to available mass production and a small form factor.

Ultrasonic sensors send acoustic wave impulses with a frequency above the human audible frequency range at roughly 20kHz. The impulses are then reflected by objects and received by the sensor. To produce and receive acoustic waves, piezoelectric transducers are used to convert electrical pulses into mechanical vibrations and vice versa. The distance and direction of objects can then be calculated by considering the speed of sound and the delta between emission and reception of the echo.

Typical automotive ultrasonic sensors report ranges between 15 cm and 5.5 m [5]. Accuracies are typically between 20 cm and 25 cm (tangential). The Doppler effect can have a significant impact on the detection. If the change in frequency is too large, the transducer (which typically operates at only one resonant frequency) will not be able to detect the returning echo. Hence, ultrasonic sensors are mainly used in low speed scenarios and are therefore used as a parking sensor or to ensure a safe take-off.

RADAR sensors Radio Detection and Ranging (RADAR) sensors have been employed in operational series vehicles since 1998 [77]. The initial application for automotive RADAR sensors was Adaptive Cruise Control (ACC), which was then followed by numerous other driving assistance systems. Hence, RADAR is a well studied field in the automotive area and a lot of experience and expertise has been built up in the last two decades.

The base principle is similar to the ultrasonic sensor. Known RADAR waveforms are transmitted and the echo of objects is then backscattered to the sensor. One main advantage of RADAR sensors is their capability to not only measure distance and direction of objects but also leverage the Doppler effect to estimate their velocity.

There exist different variants of RADAR sensors. They mainly differ in their horizontal field of view and detection range. Typically, a combination of long, mid and short range RADARs are used in an autonomous vehicle, to capture a full 360° surround-view.

Due to the wide beam spread of RADAR sensors, one of the main disadvantages is the missing ability to accurately predict the object's extent. Furthermore, in contrast to camera sensors, RADAR lacks to provide rich appearance

information and with that it is a lot harder to extract semantics from RADAR signals.

Cameras Cameras resemble best how we humans perceive the environment. They allow to capture the same dense semantic information we need to understand and navigate in a scene. This is especially helpful in environments which are designed for visual human perception such as traffic [117, p. 348].

Accordingly, camera systems have found a widespread use in today's vehicles for many tasks. Simple assistance functions help to extend the driver's view. More advanced driver assistant systems leverage cameras for lane detection, traffic sign recognition as well as pedestrian and vehicle classification [117, p. 350].

In order to understand the main advantages and disadvantages of camera systems, it is important to first analyze the main principle of how images are generated. The main requirement of image generation is that the observed scene needs to be sufficiently illuminated. The light is reflected by an object and then focused onto an image sensor by optical lens arrangements. The camera setups used in today's autonomous vehicles mainly differ in the image sensors, lenses as well as in the camera arrangement.

Image sensors transform photons to electrical charge. Today, mainly Complementary Metal-Oxide-Semiconductor (CMOS) sensors are used in vehicles [117, p. 358]. Their resolution is a trade-off between cost, quality, and pixels per degree which are needed for the desired application.

The selection of *optics* strongly depends on the application. Fisheye lenses with a Field of View (FOV) of more than 180° are used for short-range surround view applications such as parking. Front and side facing mono and stereo camera systems typically have a much smaller FOV, often ranging from 25° to 70°. They are mainly used for mid- and long range object detection and tracking.

A main advantage of cameras is that they can be easily placed on all sides of the vehicle in various configurations. An important configuration choice is whether to use two cameras in a stereo setup. While a single camera is not capable of retrieving the depth of a scene, stereo cameras can provide that depth through passive triangulation. In a stereo camera system, the scene is captured from two different perspectives, similar to the two eyes of the human vision. Typically, the cameras' lenses and image sensors are

the same and the cameras are aligned so that the axes are parallel. Given a known extrinsic and intrinsic calibration between the cameras, the depth of the scene can be estimated by finding correlations in the left and right image. To this end, the so-called *disparity* is used, which is the difference of the coordinates of two correlated pixels. In a fronto-parallel stereo camera system, the distance z_P of a point $P \in \mathbb{R}^3$ can be computed from a given baseline b, a focal length f in pixels and a disparity measurement d in pixels [117, p. 365] as

$$z_P = \frac{b\,f}{d}. \tag{1.1}$$

The impact of a disparity error Δd on the distance can be derived from linear error propagation as

$$\Delta z = \frac{\delta z}{\delta d}\Delta d = -\frac{\Delta d\, {z_P}^2}{b\,f}. \tag{1.2}$$

We can see that given a constant disparity error, the absolute distance error increases quadratically with the distance.

As an example, the stereo camera setup used to record the Cityscapes Dataset [16] has a focal length of $f \approx 2200\,p$ and a baseline of $b \approx 21$ cm. Assuming a disparity error of $\Delta d = 0.5\,p$ at an object distance of $z_P = 50$ m the distance error results in $\Delta z \approx$ -2.7 m.

Given its strengths, visual recognition has become an essential part of autonomous driving. Images contain dense information about the semantics of a scene and are therefore used to detect and classify all different kinds of objects. Their high resolution allows for exact angular measurements, which complements LIDAR and RADAR sensors. High resolution combined with color recognition enables computer vision algorithms to extract semantic information of a scene beyond any other sensor. Some abilities are unique to cameras, such as classifying traffic light states and different lane types.

However, visual perception also has its disadvantages. Due to the measurement principle, cameras are limited at night and under adverse weather conditions. And, although stereo cameras allow estimating depth, the accuracy decreases with the distance of the object, making them inferior to RADAR and LIDAR sensors for far away objects.

Given the strength of cameras to extract rich semantic information and measure exact angles but also its weakness to accurately estimate the dis-

tance of objects, it could be of great advantage to combine a camera system with an accurate depth sensor, such as a LIDAR sensor.

LIDAR sensors Like RADAR and ultrasonic, LIDAR is an active sensor. Instead of radio or audio waves, laser pulses are used to reconstruct a 3D model of the environment.

Like no other sensor, LIDAR scanners have defined the appearance of self-driving cars in the last decade. Made famous by the DARPA Challenge 2007, spinning laser scanners have been deployed on most roof racks of early self-driving car projects. The most prominent one was Velodyne's [1] HDL-64E, which was also infamous for its large size and price of 75k USD [38, 49].

Since the DARPA Challenge 2007, LIDAR technology has made significant progress in size, range, resolution and cost, making it interesting to the automotive mass-market.

The basic principle is similar in most today's LIDAR systems. A short laser pulse with an Infrared (IR) wavelength is emitted and reflected by a target. The backscatter is then sensed by the receiver. The time of flight is measured to determine the depth of the scene. Most LIDAR systems differ in the used wavelength and the way the laser beam is directed.

Wavelengths used in automotive sensors depend on many aspects. Most importantly, the human eye transmits certain wavelengths stronger to the retina than others [41]. For a long time, wavelengths around 900 nm were used. However, the transmitted energy had to be limited due to eye safety. More recently, higher wavelengths around 1550 nm are investigated, allowing to emit higher power than 900 nm LIDAR sensors. This results in higher ranges, but at the same time they suffer more scattering from atmospheric moisture [41].

Beam steering can be classified in two main principles: Mechanical and solid-state. The best known mechanical system is a rotating assembly of transmitter and receiver units as used in the Velodyne HDL-64E. A more advanced solution is a MEMS LIDAR which uses an array of micro-electromechanical mirrors. MEMS are often referred to as solid-state but still have small moving parts. Two solutions, which do not require any moving parts, are OPA LIDAR and Flash LIDAR systems.

[1] `https://www.velodyne.com`

Due to recent advances in technology, today's LIDAR sensors have an increased range, up to 250 m and an accuracy of 2 cm or better. Compared to stereo camera setups, the accuracy is independent of range and the optical structure of the scene. Furthermore, the vertical and horizontal resolution has increased, although still not comparable with the resolution of cameras.

Disadvantages, which are common to all current LIDAR sensors, are a low resolution, errors due to crosstalk and a limited performance under adverse weather conditions. Furthermore, the intensity features of LIDAR sensors still lack behind the semantic understanding of RGB cameras. And, although there are some LIDAR companies, like Aeva[2], which offer FMCW LIDAR technologies (also known as Doppler LIDAR), the majority of LIDAR sensors do not provide any per point speed estimate.

1.3 Sensor Data Fusion Strategies

In the previous section, different sensors for autonomous driving were presented. We elaborated that each sensor has unique strengths and weaknesses. Therefore, a fusion of different sensors is desired. Especially the combination of camera and LIDAR could yield an accurate depth sensor with a high angular resolution and a rich semantic understanding.

Sensor fusion approaches can be divided in three sub-categories: object-level, raw data and feature-level fusion [117, p. 448]. In the following the different approaches are briefly described.

Object-level fusion Today, most available advanced driver assistant systems rely heavily on an object-level fusion. To this end, each sensor first generates an object independent of other sensor data. The objects are then associated and updated in the fusion step. This can reduce the complexity on fusion side and limits the bandwidth for communication. Especially due to the advances in deep learning in recent years, many approaches allow for a direct 3D object representation (e.g. in form of a 3D bounding box) and tracking from raw data. However, object-level fusion comes with a significant loss of information as raw per-sensor information is already strongly condensed. This can be critical for detecting objects, which are hard to recognize in either of the sensors such as road hazards or far away objects.

[2] `http://www.aeva.ai/`

Raw data fusion In contrast to an object-level fusion, raw data fusion approaches process the data without previously altering it. In theory, this yields minimal loss of information resulting in a better performance in both, reduction of false positives and detection rate. Especially hard to detect objects can leverage from such an early fusion. However, raw sensor fusion requires a very high bandwidth and puts a significant computational burden on the fusion module as many data points have to be transmitted and fused (millions in the case of image pixels).

Feature-level fusion To provide a compromise to the two previously presented fusion approaches, feature-level fusion aims to reduce the amount of data from one sensor while still providing rich enough information for the task at hand. This is typically done in a centralized architecture in which each sensor provides features, which are then combined to an object in the fusion step. Depending on the features, the amount of information to be transferred from sensor to the fusion module can be noisy and quite large. This results in high requirements in terms of bandwidth and computational power. Hence, typically regularization and filtering is used to reduce the amount of errors and overall features. However, this often discards small or far away objects with only few measurements.

Choosing an appropriate fusion approach is a trade-off between detection performance, redundancy, bandwidth and computational power. However, with recent advances in network technology and processing power, the latter two have become less significant as they have been several years ago. Looking into the future, feature-level and raw data fusion can thus have a greater role in the development of perception algorithms for autonomous vehicles. Especially with recent advances in fusing data with deep neural networks, multimodal raw data fusion has significantly improved with respect to accuracy and computational performance [67, 68, 119]. To this end, this dissertation focuses on early sensor fusion approaches, combining the strengths of LIDAR and camera sensors.

1.4 Multimodal Deep Learning

Recent developments in machine learning, and especially deep neural networks, led to enormous progress in several research areas. For an excellent introduction to the topic of deep learning, refer to Goodfellow et al. [34].

For some years now, deep learning has had a severe impact on image understanding. Convolutional Neural Network (CNN)s pushed the limits of conventional tasks such as image classification [61], optical flow [48, 76], stereo[13, 76], object tracking [81, 114] and segmentation [72].

Motivated by the progress in the 2D image domain, many works applied neural networks also on 3D data such as dense depth maps or sparse point clouds [45, 88, 89].

Research in the field of autonomous driving pushed the development even further. Datasets, such as KITTI [30], Apolloscape [46], Cityscapes [16] and many others offer a variety of different input modalities (RGB images, stereo, LIDAR) and desired output tasks (semantic segmentation, 2D and 3D object detection, localization, dense depth estimation). Many of those tasks can be solved best by leveraging the benefits of both, 2D and 3D data. Therefore, several multimodal neural networks have been proposed recently [67, 68, 87, 126].

However, there occur several challenges in fusing 2D and 3D data with neural networks. First, many approaches require an accurate transformation from one sensor to the other to fuse features across modalities. But, obtaining such a transformation remains challenging. Second, the data representation needed for fusing two modalities remains an open question as e.g. RGB images are represented in a 2D grid while depth measurements are often represented as point clouds. Third, combining sparser depth measurements with dense 2D images often requires handling data entries with missing information. Typical image convolution operations are not designed for this purpose. Finally, multimodal deep learning requires large-scale datasets. While there are multimodal datasets available for some tasks, there are many other tasks not covered yet such as depth estimation in real world street scenes.

1.5 Dissertation Goals

In Section 1.1, we discussed the need for an alternative mobility solution to address safety and sustainability issues and improve the quality and availability of rides for everyone. Autonomous driving was identified as a promising future technology which is able to tackle those issues. However, besides recent progress, there are still many open challenges, especially with regard to reliably sensing and understanding the surrounding environment.

Most of today's approaches for autonomous driving rely on various sensors for environment perception. In Section 1.2 we compared their strengths and weaknesses and concluded that one sensor alone cannot meet the high requirements autonomous driving demands. Hence, this dissertation focuses on a sensor fusion which can leverage the benefits of each individual sensor. The fusion of camera and LIDAR sensors turns out to be particularly promising as both sensors have complementary strengths. Accordingly, we will investigate fusion approaches that leverage the benefits of those two in this dissertation. In Section 1.3 different fusion strategies were presented and the potential of a raw data fusion was pointed out as the loss of information is minimal. As a promising instrument for raw data fusion, deep neural networks and their success on multimodal data processing were identified in Section 1.4. However, open challenges of multimodal data fusion were identified. In particular, the handling of sparse data within neural networks and CNNs. Those challenges are addressed in this dissertation.

Convenient and accurate extrinsic calibration Multimodal fusion approaches typically combine features from different sensors and make decisions based on their correlation. This requires an accurate transformation between sensors to be able to associate features. If the sensor data is not properly aligned, this can result in a performance decrease. Obtaining a transformation between different sensors is referred to as *extrinsic calibration* and is still a challenging and often laborious task. Most available cross-sensor multimodal calibration approaches are typically limited to provide either a good initial calibration *or* a continuous online calibration. While accurate calibration approaches often involve human interaction, artificial calibration targets, and are computationally expensive, most online calibration algorithms require a good initial calibration in order to find matches of handcrafted multimodal features. The question arises if an approach can be developed which involves neither calibration targets nor human interaction, is fast and suited for both, a good initial calibration and a continuous online calibration.

Common representation for images and depth While there is an increasing trend towards directly using point clouds as an input representation [87–89], more commonly, depth measurements are projected onto a 2D (e.g. an RGB image) or 3D grid prior to processing. However, in the resulting

grid, typically only fractions of the grid cells contain depth measurements. A prominent case is sparse LIDAR data which is projected onto a dense RGB image. The obtained mapping is incomplete resulting in undefined correlations between depth and RGB. The challenge which is addressed in this dissertation is how to obtain accurate dense depth maps beyond simple interpolation. In doing so, a 1-to-1 correspondence between RGB and depth measurements is obtained.

Handling sparse data with CNNs As discussed in Section 1.4 multimodal deep learning is a promising research direction. A common approach to combine point clouds and 2D images is to project point clouds onto high resolution images and then use convolutional neural networks to process both, RGB images and sparsely populated depth maps. However, the output of a convolution is expected to have similar activations for a similar semantic content or structure. This is not the case for random sparsity as the filter response varies with the level of sparsity in the input and thus makes it difficult to train CNNs in this setting. While completing the projected depth points before processing is one way to tackle this, there are several advantages to process sparse data directly. It would get rid of the often computationally expensive preprocessing and avoids altering the input and with that removes a possible source of errors. Accordingly, there is a need for convolutions, which consider the level of sparsity in the input layer.

Large-Scale Datasets Training and evaluating deep neural networks requires a large amount of data. Therefore, several multimodal datasets for autonomous driving have been published in the last few years. However, dense depth datasets for tasks like depth completion and single image depth prediction are either relatively small or rely on synthetic rendering. Hence, there is a strong demand for a large-scale real world dense depth dataset.

In this dissertation those open research questions are addressed by following a common recipe. We scan the literature for existing solutions, identify remaining issues, and address these by investigating new approaches. While many of the presented approaches are also applicable to other tasks, we keep an automotive environment in mind and work towards a system that combines the strengths of LIDAR and camera sensors.

1.6 Contributions and Outline

There are four major contributions in this dissertation:

CNN for online multi sensor calibration Multimodal sensor calibrations rely on distinct features which can be identified in both modalities. For initial calibration, a structured environment containing special calibration targets is typically used. However, such an environment is time- and cost-expensive and requires autonomous vehicles to return to the calibration station, once the setup deviates from its initial calibration. In this work, a CNN is presented which provides not only a good initial calibration but is also able to update the calibration in an online case. The proposed CNN is able to fully regress the calibration between a camera and a LIDAR sensor. In contrast to previously proposed online calibration methods it can compensate for deviations in angle and translation up to 20° and 1.5 m, respectively.

Large-scale depth dataset In the context of autonomous driving, depth completion from sparse LIDAR data as well as single image depth prediction are a research area with increasing importance. Most modern approaches leverage deep neural networks for both mentioned tasks. Training and evaluation of those networks require a massive amount of data. However, there are currently no large-scale datasets for dense depth prediction tasks available. To this end, a new large-scale dataset and a corresponding benchmark is presented.

Semantically Guided Depth Completion In order to generate high resolution depth maps, stereo vision has been the method of choice in the automotive field for many years. LIDAR depth measurements are more accurate and can provide measurements in a larger distance, but are significantly sparser. To lift those limitations, an approach is presented which provides both, dense and accurate depth measurements, by completing the sparse LIDAR data under the guidance of the high resolution RGB image. To this end, semantic information is leveraged in a novel energy formulation for depth completion.

Sparsity Invariant Convolutions CNNs have impacted almost all areas of 2D and 3D data processing. However, a common assumption is that the

input to a CNN is a densely populated grid, which is the case for images. Yet, there are several applications in which the input is sparse and irregular (e.g. if a point cloud is projected onto an image). We will see that naively applying a standard convolution to this sparse data leads to suboptimal results and present a simple yet effective sparse convolution.

The work that led to those contributions will be presented in four main chapters. In this first chapter, we investigated the topic of autonomous driving and identified several open challenges for sensor fusion and multimodal deep learning. To address these challenges, the remainder of this dissertation is structured as follows. First, we start with a multimodal calibration of LIDAR and camera sensors in Chapter 2, which is a prerequisite for a successful sensor fusion. For the approach, we investigate the capability of deep neural networks for sensor data registration. Second, precise dense depth maps are calculated in Chapter 3 by leveraging the fusion of highly accurate LIDAR data with dense semantic image information. To this end, first a non-learned approach is investigated in Section 3.3 which formulates depth completion as an energy minimization that considers image gradients as well as semantic information. In the second part of the chapter a learned approach to depth completion is demonstrated. Accordingly, we investigate in Section 3.4 the influence of sparse measurements on CNNs and explore a solution which considers missing information in the input. Additionally, for training and evaluating dense depth methods, a large-scale depth dataset is introduced in Section 3.5 accompanied by a benchmark to compare depth completion and single image depth prediction methods. Note that both, Chapter 2 and Chapter 3, contain a detailed overview on related work as well as an extensive evaluation section. Finally, a conclusion of this dissertation, discussion and outlook on future work are given in Chapter 4.

2 LIDAR Camera Calibration

In order to represent data from different sensors in a common coordinate system, a transformation from each sensor to the target coordinate system is required. The transformation can be described by a rotation and translation and estimating their parameters is referred to as extrinsic calibration. Estimating a calibration between sensors from different modalities is especially challenging as matching features for association need to be extracted in two often very different modalities. This chapter presents a calibration approach which addresses these challenges by investigating a CNN to infer a 6 Degree of Freedom (DOF) extrinsic calibration, exemplified by a scanning LIDAR and a monocular camera. The conventional calibration steps, such as feature extraction, feature matching and global regression are cast into a single real-time capable CNN. The presented method does not require any human interaction and bridges the gap between classical offline and target-less online calibration approaches as it provides both, a stable initial estimation and a continuous online correction of the extrinsic parameters. The approach compares favorably to state-of-the-art methods in terms of calibration error, even for large decalibrations up to 1.5 m and 20°. The main content and structure of this chapter is based on [100].

The chapter is structured as follows. In Section 2.1 the need of an accurate online and offline calibration without human interaction is motivated, followed by an overview of the current state-of-the-art for online and offline calibration in Section 2.2. The data representation and implementation details of the neural network can be found in Section 2.3. Last, the proposed method is evaluated in Section 2.4 and the results are discussed in Section 2.5.

2.1 Introduction

Recent solutions for autonomous driving rely on a variety of optical sensors in order to have a redundant and robust system. As already mentioned in Chapter 1, especially the fusion of camera and depth sensors is promising

and has therefore been studied intensively in the last few years. However, typically the sensor data is defined in different coordinate systems. In order to fuse the data, a transformation ${}^{A}T_{B} \in SE(3)$ between the coordinate system A of the first sensor and the coordinate system B of the second sensor has to be defined. The transformation can be split into a rotation described by a 3×3 matrix $R \in \mathbb{R}^{3\times3}$ and a translation described by the vector $t \in \mathbb{R}^{3}$, i.e.

$$ {}^{A}T_{B} = \left[\begin{array}{ccc|c} & R & & t \\ \hline 0 & 0 & 0 & 1 \end{array}\right]. \tag{2.1} $$

Now a point $\bar{p}_{A} \in \mathbb{R}^{3}$ in homogeneous coordinates can be transformed from sensor coordinate system A into a point $\bar{p}_{B} \in \mathbb{R}^{3}$ in sensor coordinate system B by

$$ \bar{p}_{A} = {}^{A}T_{B}\, \bar{p}_{B}. \tag{2.2} $$

The task of obtaining the transformation ${}^{A}T_{B}$ is referred to as *extrinsic calibration* and can be divided into three steps:

1. Extracting distinct features in the sensor data, e.g. corners or artificial targets;

2. Establishing correspondences between extracted features across sensors;

3. Determining R and t by solving a system of equations or by minimizing an error function, given the set of correspondences.

Especially in the case of RGB cameras and depth sensors, the extraction of distinct features can be challenging as correspondences across different sensor modalities have to be established. Hence, most offline calibration approaches rely on special calibration targets which provide distinct features in all modalities that can easily be detected and localized [31, 79, 124]. However, most of those approaches are time-consuming as human interaction is required, e.g. for feature selection or performing the calibration in a controlled environment. Accordingly, several online calibration methods have been proposed [3, 14, 65, 83]. However, online calibration can be particularly challenging as matching patterns have to be identified in an unstructured environment. To this end, many approaches leverage handcrafted features such as depth and image edges. But the descriptors of handcrafted

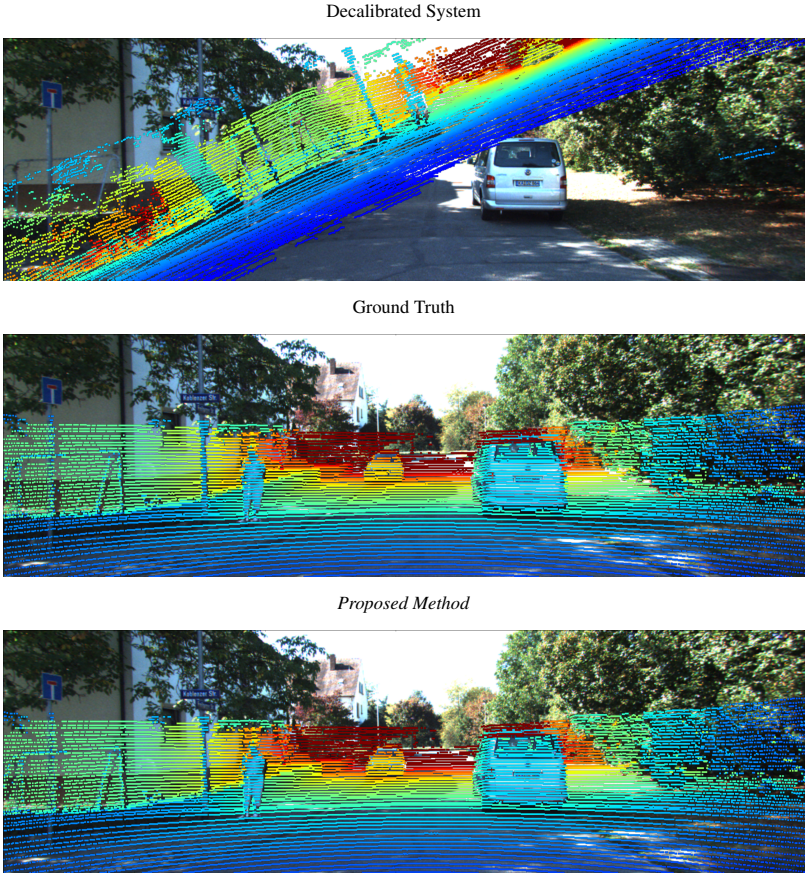


Figure 2.1: Neural Network for calibrating LIDAR and camera. The presented approach is able to correct even large decalibrations such as depicted in the top image. The inputs for the deep neural network are an RGB image and a projected depth map. The network is able to establish correspondences between the two modalities which enables it to estimate a 6 DOF extrinsic calibration. Adapted from [100].

features are often not discriminative enough, resulting in poor feature matching. Subsequently, the optimization does not lead to satisfying results. This effect becomes particularly apparent for large initial calibration errors.

In this chapter, we investigate a different approach to target the shortcomings of traditional offline and online calibration methods. In particular, we discuss a CNN to fully regress the extrinsic calibration between sensors of different modalities.

The proposed network is able to solve all classical calibration tasks, such as feature extraction, feature matching, and global optimization by using an end-to-end trained deep neural network. With the recent advances of Graphics Processing Units (GPUs), this can be efficiently done in real-time. As collecting ground-truth can be time-consuming, the approach avoids to manually calibrating a huge amount of LIDAR camera pairs. This is achieved by a simple yet effective sampling strategy, which allows to generate an infinite amount of training data from only one manually conducted extrinsic calibration. After successfully training the network, no further human interaction is required. The approach has a particular advantage if many identical setups need to be calibrated as this is the case for large fleets of vehicles. Requiring only one manually calibrated setup for training, the resulting network can then be deployed to the remaining vehicles, where it performs an initial calibration and can later be used to continuously monitor deviations online. This enables sensor systems to operate longer, without the need of returning to a controlled environment for re-calibration.

2.2 Related Work

In this section, we compare different approaches for extrinsic calibration of a LIDAR-camera system. Extrinsic calibration can be divided in offline and online. Offline calibration is presented in Section 2.2.1. It requires either special targets or demands too much processing power to be computed during the operation of the system. Online calibration on the other hand, is performed live during operation by continuously updating the calibration. Approaches for online calibration are investigated in Section 2.2.2. Independent of their assignment to online or offline calibration, all methods share the steps of *feature extraction*, *feature matching* and *global optimization*. The goal of this section is to identify the differences and weaknesses of the approaches w.r.t. those three steps.

2.2.1 Offline Calibration

In this section different approaches for offline calibration are discussed. In contrast to online calibration methods, they handle the registration between a camera and a depth sensor by using special calibration targets in a controlled environment. This greatly simplifies the feature extraction and matching part of the calibration.

Zhang et al. [125] leverage planar checkerboard targets to compute the rotation and translation between a camera and a 2D LIDAR scanner. For the camera part, the checkerboard corners are automatically extracted and a plane in camera coordinates is estimated (assuming a known distance between two adjacent checkerboard corners). For the LIDAR part however, the points on the checkerboard have to be selected manually. This omits the feature extraction and matching step, but requires human interaction. Solving for the extrinsic calibration is a 3-step approach. First, a linear solution determines the LIDAR sensor extrinsic parameters. They serve as an initialization to the second step which considers outliers in a nonlinear optimization. Finally, also the camera's intrinsic parameters are refined in a global nonlinear optimization. The approach has several drawbacks. First, feature extraction and matching are bothersome as they require human interaction. Second, the three-step approach is computationally expensive as it has 3 stages, involving two nonlinear optimizations.

Mirzaei et al. [79] use a similar approach for feature extraction and matching but additionally optimize the intrinsic parameters of a rotating multilayer LIDAR. In particular, each individual LIDAR beam is calibrated intrinsically and extrinsically with respect to the camera. Considering the intrinsic LIDAR calibration is helpful if the sensor diverged from its factory calibration. The drawbacks of the method however, are the same as the one by Zhang et al. [125].

A further weakness of both aforementioned methods is the human interaction which is needed in order to move the calibration target during the calibration process. In contrast, Geiger et al. [31] propose to determine the intrinsic camera parameters as well as the extrinsic parameters of sensors jointly with only a single image and range scan per sensor. To this end, a calibration environment is established, containing several, carefully placed checkerboard targets in one room. As previously, checkerboards are used to estimate planes in the camera space. However, Geiger et al. use an automatic segmentation step which considers point cloud normals. The

matching between camera planes and LIDAR segments is done by randomly selecting three planes, calculating the transformation between camera and LIDAR and verify it by a distance measure. In the final global optimization, gradient descent is performed on the error function which minimizes the distance of associated LIDAR and camera checkerboard points in 3d. While the approach does not rely on manually moving a checkerboard, it requires a controlled environment with several calibration targets. Furthermore, the plane segmentation in the LIDAR data can be unstable if the sensor data is noisy.

Some calibration approaches rely on human selection instead of an automatic target extraction [108, 109]. This can be helpful if a calibration is needed after the data recording has already been carried out or no calibration target is available. Those semi-automatic methods simplify the feature extraction and matching step by selecting 3D and 2D shapes manually from both sensors. The achieved results can be very accurate and can therefore be used for a suitable initial calibration. However, the approach heavily relies on a good human segmentation which is time-consuming.

2.2.2 Online Calibration

While providing an excellent initial calibration, offline calibration methods are either time-consuming [79, 108, 109, 125] or require a controlled environment [31]. However, once a sensor system starts to operate, external forces such as mechanical vibrations or temperature changes may move the sensors from their initial position, resulting in a decreased calibration quality. The task of detecting and correcting such decalibrations is referred to as online calibration. This section compares several recent studies on online calibration and compares their strengths and weaknesses.

In order to align LIDAR scans to camera images several approaches extract depth and intensity edges as features and match them in a second step. Bileschi et al. [3] project depth edges onto the RGB image and match them by selecting the nearest neighbor. This however requires a very good previous initialization of the calibration. Similarly, Levinson et al. [65] leverage depth and image gradients to estimate the extrinsic calibration. Instead of just selecting the nearest neighbor for matching, they project the depth edges onto a distance transform of the edge image. The calibration parameters are then determined by an energy minimization which encourages strong depth gradients to be projected close to image edges. However, the optimization

often gets stuck in local minima and therefore also requires a good initialization.

While correlations between depth and image edges are leveraged for many tasks such as calibration and depth completion, there are weaknesses especially at highly textured surfaces. Instead, Pandey et al. [83] leverage laser reflectivity and camera image intensity as features for matching. The final calibration parameters are obtained by maximizing the mutual information between the sensor intensity measurements. While laser intensities are dependent on object characteristics, camera image intensities can vary due to lighting conditions. This may result in a weak performance of the algorithm, especially noticeable if shadows are existent in the scene. Furthermore, both aforementioned approaches are not able to deal with the parallax between the two sensors. To address those shortcomings, Chien et al. [14] propose an extrinsic calibration which is based on a LIDAR-camera visual-odometry. For the visual-odometry they assign depth values from the LIDAR point cloud to image feature points. They argue that false extrinsic parameters result in wrong assignments which increases the error in the ego-motion estimation. The calibration is obtained by optimizing the reprojection error. To additionally constrain the problem, they leverage image and depth gradients as well as mutual information between intensities as mentioned in the two previous approaches [65, 83]. This leads to stable results if the calibration is disturbed by not more than 2° and 10 cm. However, the energy minimization often gets stuck in local minima and is therefore not able to compensate large calibration errors. Furthermore, the approach is computationally expensive as it solves for the visual odometry and extrinsic parameters iteratively, using the Levenberg-Marquardt algorithm and gradient-descent.

2.3 Multimodal Sensor Data Registration Using CNNs

In the previous section we discussed current state-of-the-art methods for multimodal extrinsic calibration and their shortcomings. While offline calibration methods are typically time-consuming and require a controlled environment, multimodal online calibration approaches experience difficulties in finding correlated robust features across different modalities. In this section we investigate an approach which is able to address those shortcomings and can be used for offline and online calibration.

As already described in Section 2.1, extrinsic calibration can be divided in three major stages: feature extraction, feature matching and global optimization. For feature extraction, most state-of-the-art approaches for calibration use hand-crafted features like image and depth gradients or intensity cues. More recently however, deep learning methods have proven to be better suited for a robust generalized feature extraction. The same applies for feature matching. Tasks involving sparse or dense matching of features, such as stereo vision or optical flow, are now dominated by deep learning approaches. And finally, global optimization can be computationally expensive and often depends on a good set of parameters. In contrast, deep neural networks are getting more and more efficient due to massive parallel computation and do not require manual fine-tuning of the optimization parameters.

Given the advantages of deep learning in each stage, we elaborate the potential of deep neural networks to calibrate multimodal sensors in this section.

Although the approach can be generalized to various combinations of multimodal sensors, this thesis focuses on the registration of LIDAR and camera sensors. Therefore, the problem is simplified to estimating the transformation matrix $T \in SE(3)$ through a neural network f_θ with weights θ, given pixels $I \in \mathbb{R}^{W \times H \times 3}$ of an RGB input image with height H and width W and a LIDAR point cloud $L \in \mathbb{R}^{N_p \times 3}$ consisting of N_p LIDAR points as

$$\begin{aligned} f_\theta : \mathbb{R}^{W \times H \times 3} \times \mathbb{R}^{N_p \times 3} &\rightarrow \mathbb{R}^{4 \times 4} \\ (I, L) &\mapsto T. \end{aligned} \tag{2.3}$$

The remainder of this section is structured as follows. First, the input and output data representation is described in Section 2.3.1 as this is most crucial for the network training. Next, the design and training of the neural network is presented in Section 2.3.2. In Section 2.3.3 we discuss how an iterative refinement improves the calibration output. Finally, the details of the training can be found in Section 2.3.4.

2.3.1 Data Representations

The performance of deep neural networks strongly correlates with the amount and quality of data presented during training. In order to solve for the parameters θ of the neural network, a large set of images and corresponding LIDAR scans accompanied by a ground truth calibration is required. In order to prevent overfitting, the ground truth calibration has to differ across

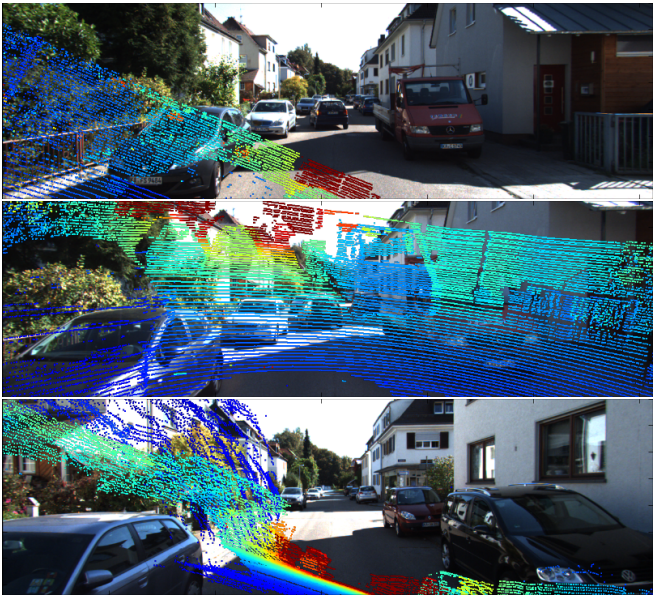


Figure 2.2: Examples of various decalibrations. The initial calibration is deviated up to 20° in rotation and up to 1.5 m in translation from the ground truth calibration. This might result in projections of the LiDAR points where most of the points are outside the image area. It is therefore difficult to establish correspondences with the RGB image. Adapted from [100].

the various pairs. However, determining the ground truth for thousands of differently arranged LiDAR-camera pairs is time-consuming and not very effective. Therefore, the problem of extrinsic calibration is reformulated as determining the transformation $T_{\text{decalib}} = {}^{gt}T_{init} \in SE(3)$ between an initial transformation $T_{\text{init}} \in SE(3)$ from camera to LIDAR coordinate system and a ground truth transformation $T_{\text{gt}} \in SE(3)$. This allows to vary T_{init} randomly to get a large amount of training data.

In order to be able to establish correspondences between LIDAR and camera, the LIDAR point cloud is projected onto the image. The projected pixel positions $P = \{(u_i, v_i, 1)^T\}_i^{N_p} \in \mathbb{R}^{3\times N_p}$ are obtained by transforming the LIDAR points $L = \{(x_i, y_i, z_i, 1)^T\}_i^{N_p} \in \mathbb{R}^{4\times N_p}$ given the initial transformation matrix T_{init} and an intrinsic calibration matrix $K \in R^{3\times 4}$. Obtaining the projected pixel coordinate of a single point LIDAR then leads to

$$z_c \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = K \, T_{\text{init}} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}. \tag{2.4}$$

Note that in this dissertation the perspective projection model is used and the intrinsic camera parameters of K, such as the focal length, skew and principal point are assumed to be known. For more information on the perspective projection model and intrinsic calibration refer to *Computer Vision: Algorithms and Applications* by R. Szeliski [106].

After projection, the inverse depth of the projected point (in camera coordinates) $\frac{1}{z_c}$ is stored at each pixel position to obtain an inverse depth map which is independent of the RGB values.

As most common LIDAR sensors have a lower resolution than modern image sensors, the resulting inverse depth images are not densely populated. To account for the sparsity, the projected LIDAR points are densified by applying max pooling on the input depth map. Furthermore, both, the inverse depth map and the camera image are mean adjusted.

In order to reduce the amount of learned parameters, the rotational component of the transformation T_{decalib} can be represented by e.g. Euler angles or quaternions, like proposed in [55]. However, both Euler angles and quaternions only describe rotations and so the translation parameters have to be estimated independently. This can lead to over-adjusting a bad calibration as often it is challenging to distinguish a rotational from a translational error. In contrast, dual-quaternions offer a unified representation of translation and rotation. A dual-quaternion σ is composed of two quaternions. The real part p and the dual part q. Combining p and q with the dual number ε leads to

$$\sigma = p + \varepsilon q. \tag{2.5}$$

To represent rigid rotations and translations, the dual quaternion can be rewritten as

$$\sigma = q_r + \varepsilon \frac{1}{2} q_t q_r, \tag{2.6}$$

where q_r is a unit quaternion describing the rotation and q_t is a quaternion describing the translation. The rotational part can be described by a rotation of an angle ϕ about a unit axis $n = (n_x, n_y, n_z)^T \in \mathbb{R}^3$, resulting in

$$q_r = cos(\frac{\phi}{2}) + sin(\frac{\phi}{2})n_x i + sin(\frac{\phi}{2})n_y j + sin(\frac{\phi}{2})n_z k. \tag{2.7}$$

The translational part q_t can be written as

$$q_t = 0 + t_x i + t_y j + t_z k, \tag{2.8}$$

where $t = (t_x, t_y, t_z)^T \in \mathbb{R}^3$ is a translation vector between the origins of the two coordinate systems. As can be seen, the dual part links rotational and translational information, forcing the neural network to optimize for both jointly. While q_r is a unit quaternion with $\|q_r\|_2 = 1$ and therefore limited in its values, q_t is represented by values without a specific range. This results in an imbalance of the dual quaternions during training. To compensate this effect, the values of the real part are multiplied by a factor f. As the real part contains only rotational parts, this results in a weighting of the rotation for the loss function of the CNN.

2.3.2 Network Architecture

In this section a neural network architecture is presented which is specifically designed to solve the tasks of feature extraction, feature matching, and regression of the calibration parameters. As both inputs to the network are regular grids, a CNN is used which can be trained end-to-end, meaning all steps from input to output are learned. The outline of the network as well as its embedding in the training pipeline is depicted as block diagram in Figure 2.3.

The network is built of several Network in Network (NiN) blocks which have been proposed by Lin et al. [69]. Such a block is composed of one k×k convolution followed by two 1×1 convolutions. Note that more complex network architectures are possible and might improve the results even further. However, NiN blocks are easy to train and have a relatively low inference time, which is needed for a fast online execution of the network.

Feature Extraction. In order to encourage the network to extract a rich feature representation for each modality individually, the RGB and LiDAR depth maps are processed separately, resulting in two parallel data network

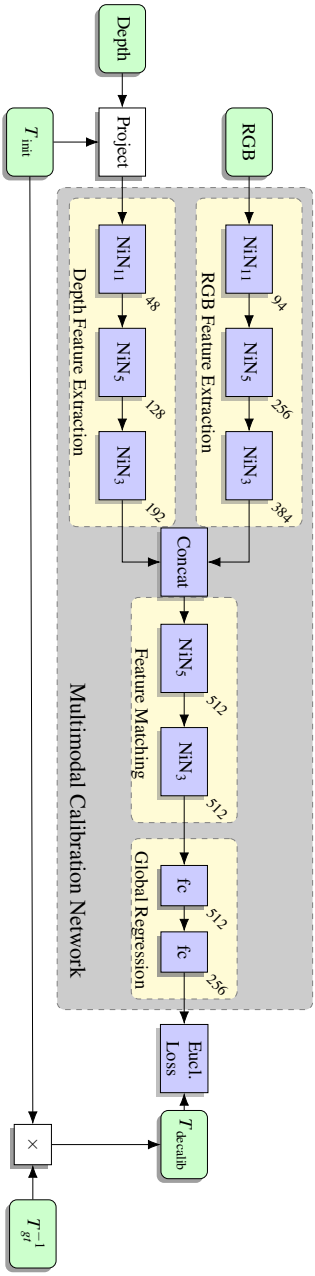


Figure 2.3: Network architecture of the calibration neural network. The neural network estimates the calibration between a depth and an RGB sensor. The depth points are projected onto the RGB image given an initial calibration T_{init}. In the first part, NiN blocks are used to extract rich features for matching. One NiN block consists of a $k \times k$ convolution followed by two 1×1 convolutions. After each convolution a RELU activation is applied. The kernel size k of the first convolutional layer of the NiN block is denoted by the indices of the block, whereas the number of used kernels is shown in the top right corner of each layer module. After feature extraction, the features of the depth map and the RGB image are concatenated and further NiN blocks are applied. Finally, the network regresses the decalibration by gathering global information using two fully connected layers. During training T_{decalib} is randomly sampled resulting in different projections of the depth points. Adapted from [100].

streams. For the RGB part, the architecture by Lin et al. [69] is used with the weights being initialized with their pretrained version for ImageNet [17]. The last NiN block is skipped as it is only needed for classification. In contrast to the RGB part, the depth stream is trained from scratch. While the number of hidden layers is identical, the number of kernels is reduced to enable faster convergence.

Feature Matching. After extracting features from both streams, the network should find correspondences between the two input modalities. To this end, the feature maps are first concatenated and then additional convolutions are applied on those fused feature maps. The resulting representation can then be used for the final global regression step. The described architecture is motivated by Dosovitskiy et al. [23] who also proposed a specific correlation layer that performs multiplicative patch comparisons between two feature maps. However, the reported results with and without correlation layers are almost identical. Therefore, they are omitted for simplicity in this work. Similar to the feature extraction part, the convolutions in the matching part are realized as a stack of NiN blocks.

Global Regression. In order to obtain the deviation from the original calibration, the joined representation needs to be gathered to obtain the final output representation. This step can be interpreted as an information comparison of all matched features between depth and RGB image. This is comparable to a global optimization in classical calibration algorithms. To realize this final step, two fully connected layers are connected in series. Note that separating the network to handle translation and rotation in different branches, results in worse results. A similar behavior was reported by Kendall et al. [55].

2.3.3 Further Calibration Refinement

Guessing the calibration from a single input scan and an RGB image can be difficult. We therefore discuss some further refinement steps in this section.

Iterative Refinement. The appearance of the projected depth map strongly depends on the given initial calibration as depicted in Figure 2.2. Some calibrations result in projections with most LIDAR points outside the image area. Accordingly, only few correspondences between the inverse depth map and the RGB image can be established. However, although a pixel-level matching might not be possible in this case, the network can still improve the calibration by considering the position and rotation of the whole projected

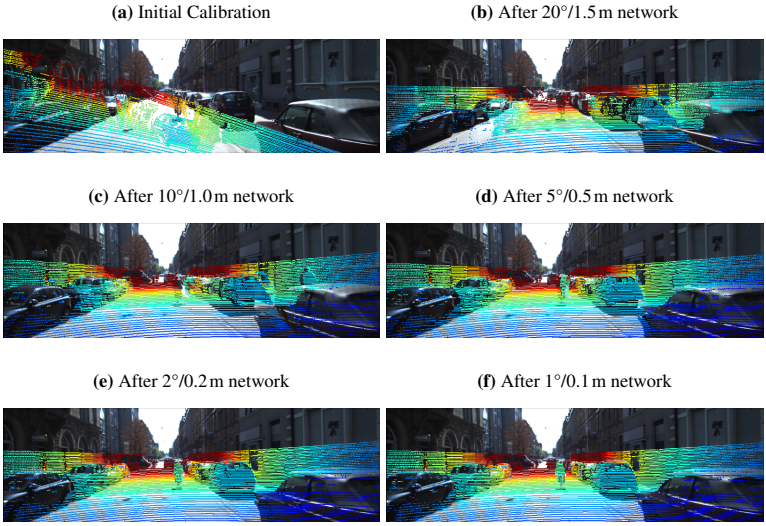


Figure 2.4: Visualization of the iterative calibration refinement. For the iterative refinement, the estimated calibration of one expert network is used to improve the projection of the depth points. The refined depth map is then forwarded to the next network. From top left to bottom right we can see a constant improvement in each iteration step. Adapted from [100].

scan. The transformation after calibration $\hat{T} = T_{\text{init}} \hat{T}_{\text{decalib}}^{-1}$ can then be used to obtain a better projection which allows establishing more correlations between depth map and RGB image. This step can then be iterated to improve the calibration. An example of an iterative refinement can be seen in Figure 2.4.

Temporal Filtering. During the online execution of the calibration, there might be some scenes which contain better features for calibration than others (e.g. an urban scenario might contain more structure than a highway scenario). This can result in noisy calibration estimations. However, often an instant registration between the sensors is not needed. If the measurements are filtered over time, e.g. by a moving average, the approach yields more robust results. This can be leveraged especially in the context of autonomous driving to either obtain a good initial calibration (where a whole drive with all measurements can be taken into account), or to correct a calibration online (which is typically done at a lower frequency).

2.3.4 Training Details

The implementation and training of the network was done in the Caffe library which was introduced by Jia et al. [52]. After each convolutional layer but the last Rectified Linear Unit (RELU) is applied as activation function. Experiments with Exponential Linear Unit (ELU) and sigmoid as activation function did not yield improved results. As optimizer, the Adam solver [58] performed better than Stochastic Gradient Descent (SGD). The parameters of the solver are set to the suggested default values $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\varepsilon = 10^{-8}$ with a fixed learning rate of $\alpha = 10^{-5}$. To infer the deviation from the ground truth decalibration, a Euclidean loss is chosen. The network is trained for three million iterations with a batch size $b = 1$ - a larger batch size did not improve the results. The RGB stream is initialized by the pretrained ImageNet weights. All other weights are initialized using Xavier initialization [33].

2.4 Experiments

Several experiments on real sensor data are performed in order to evaluate the presented approach. Due to focus of this thesis on autonomous driving and LIDAR camera fusion, this section evaluates only a LIDAR-camera setup, although the network might be used on other combinations of multimodal sensors. Note that from earlier experiments it became clear that the rotational components have a larger impact on the quality of the resulting registration. On top of that, they are harder to determine by the network. Therefore, the comparisons between different methods are mainly based on the rotational components.

2.4.1 Dataset

For the experiments, the KITTI dataset [29] was chosen as it contains both, a camera and LIDAR sensor. In detail, the sensor data consists of 1392×512 pixel RGB images as well as LIDAR scans from a Velodyne HDL-64E LiDAR scanner. Furthermore, KITTI provides enough training samples to train the presented model and evaluate its online performance in an urban real world environment. As ground truth calibration, the extrinsic parameters of the dataset were calculated using the method of Geiger et al. [31]. In order to have a huge variety of data during training, validation and testing, the raw

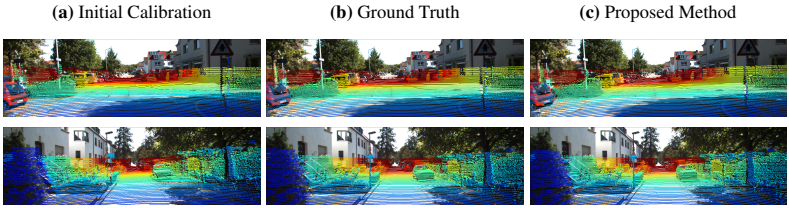


Figure 2.5: Calibration results on minor decalibrations. Examples of calibration results for an online scenario based on decalibrations up to $0.2m$ and $2°$, where only two networks are executed iteratively (2°/0.2 m and 1°/0.1 m network). Adapted from [100].

sequences of the KITTI dataset are leveraged. For each recording day, different intrinsic and extrinsic calibrations were calculated. During training and validation, only sequences of the first recording day (09/26/2011) are used. Two challenging sequences (drive 0005 and 0070 with 574 frames in total) are selected for validation while all other sequences are only used for training (14863 frames). At training time, the initial calibration T_{init} is varied randomly for each frame as described in Section 2.3.1. This yields a potentially infinite amount of training data. The final evaluation (Section 2.4.4) is executed on a separate day and sequence (09/30/2011 drive 0028 with 5177 frames) in order to have more independent test and training calibrations.

2.4.2 Decalibration Representation

In this section, three different representations of the decalibration are compared: Euler angles with translation, quaternions with translation and dual quaternions. As explained in Section 2.3.1, for dual quaternions the values of the real part p are distributed differently in comparison to the values of the dual part q. Therefore, a factor $f_{dq} = 100$ is used to balance the loss between real and dual part. As the real part contains only rotational components, this has a direct influence on the quality of the rotation estimate. Using larger values of f result in volatile translations whereas smaller values result in worse rotation estimates.

The quaternion representation is also composed of a rotational part described by a unit quaternion and an unbound translation. Similarly to the dual quaternion representation, a factor $f_q = 100$ being applied to the loss of the quaternion components yields the best results for both rotation and translation.

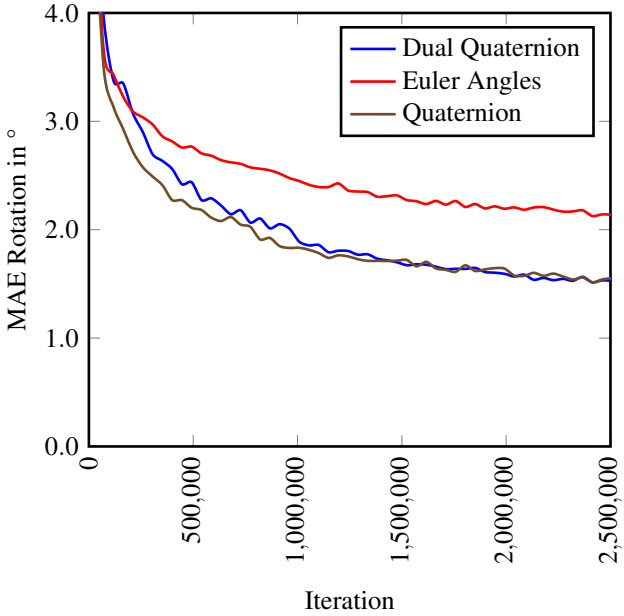


Figure 2.6: MAE over training iteration of different rotation representations. Euler angles are represented in red, quaternions in brown and dual quaternions in blue. Both quaternion representations outperform the Euler angles representation. Adapted from [100].

In Figure 2.6 the mean absolute error of the estimated rotation is plotted over the number of training iterations. It is noticeable that both quaternion representations outperform the Euler angles. As there are only small differences in the two quaternion variants, but a slight tendency towards dual quaternions, the subsequent experiments are performed using only dual quaternions.

2.4.3 Different Decalibration Ranges

During the training of the network, it is challenged to compensate for random decalibrations in the range of [−1.5 m, 1.5 m] for translation and [−20°, 20°] for rotation. Using a Euclidean loss penalizes large deviations which puts a strong focus on large decalibrations during training. However, for

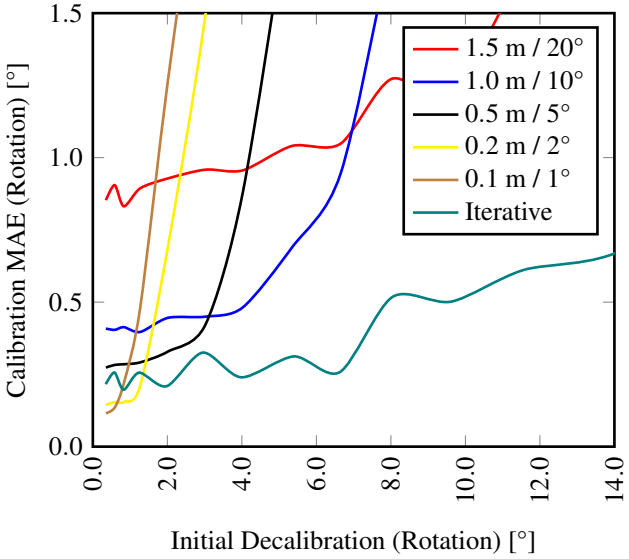


Figure 2.7: Calibration performance over decalibration magnitude. Analysis of the rotation estimation error over the decalibration magnitude for different networks. The networks have been trained on random initial decalibrations, varying from 0.1 m/1° to 1.5 m/20°. It can be seen that the networks perform better on certain decalibrations, depending on the range they have been trained on. Therefore, an iterative execution of experts is proposed. Adapted from [100].

smaller decalibrations, the relative improvement is less, which is depicted in Figure 2.7.

Another interesting finding is that the values of the filter weights that are applied on the depth image are quite randomly arranged for the networks which have seen larger deviations as can be seen in Figure 2.8. This indicates that the larger networks mostly memorized how to translate and rotate point clouds based on their overall position and alignment in the image. However, networks which have seen smaller deviations are forced to extract more meaningful features to be able to register depth and RGB images.

In order to compensate for this behavior, it is suggested to train several expert networks on different decalibration ranges. As the networks are executed one after another iteratively, the ranges are chosen based on the worst Mean Absolute Error (MAE) of the previous network. This results in

(a) 20°/1.5 m network **(b)** 2°/20 cm network

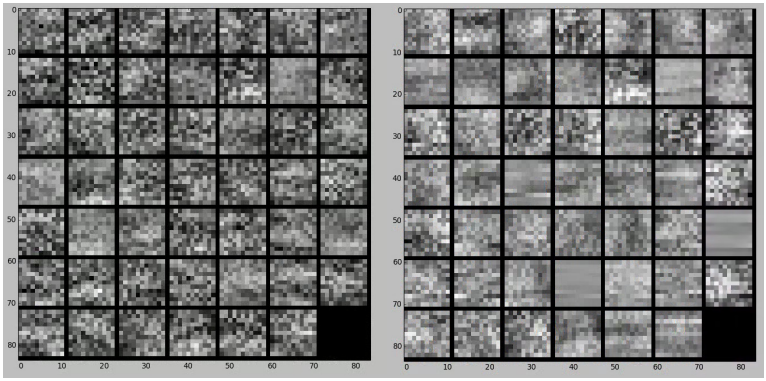


Figure 2.8: Visualization of filter values of first LIDAR layer. The two images show the values of the first filter layer which is applied on the projected LIDAR depth image normalized to the range $[0,255]$. The model handling larger deviations (left side) seem less structured yielding the possibility that the network has mostly memorized how to shift the overall point cloud without having to actually extract meaningful features for matching. The kernels of the second model however show a lot more structure, hinting at the extraction of depth edges and other features.

the following expert networks: $[-x,x]$ / $[-y,y]$ (translation / rotation) for $x = \{1.5\,\text{m}, 1.0\,\text{m}, 0.5\,\text{m}, 0.2\,\text{m}, 0.1\,\text{m}\}$ and $y = \{20°, 10°, 5°, 2°, 1°\}$.

2.4.4 Temporal Filtering

In the previous experiments, only one frame was used to determine the calibration. However, the result can be quite noisy due to missing structure, sensor artifacts like rolling shutter or dynamic objects. In order to achieve further improvements, the results can be analyzed over time as mentioned in Section 2.3.3. To this end, the distribution of the results over the whole test sequence is determined, while keeping the decalibration fixed.

Figure 2.9 depicts two examples of the distributions of the individual components by means of box plots. Most of the estimated decalibrations $\hat{T}_{\text{decalib}}$ are distributed well around the ground truth values. Selecting the median of all calibration estimates over the whole sequence results in the best performance on the validation set.

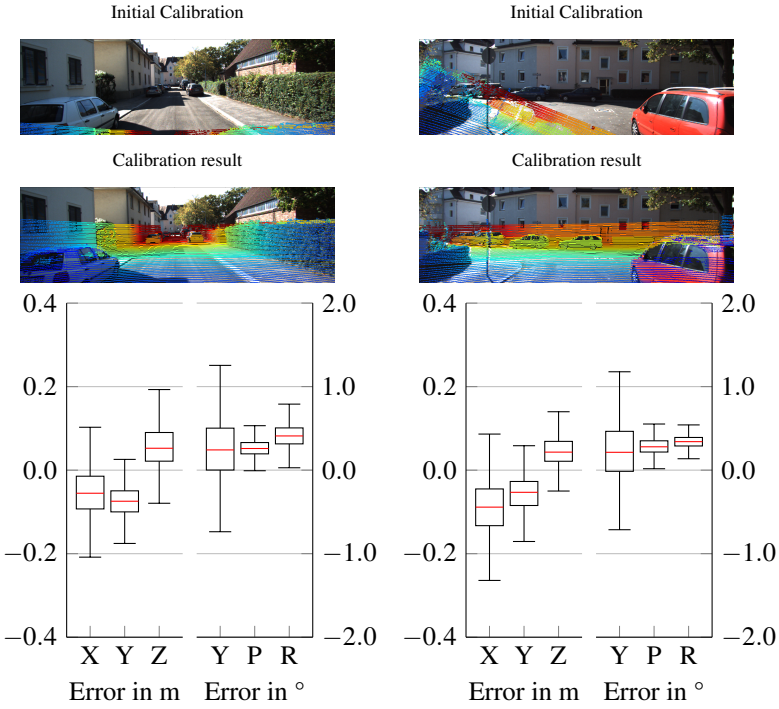


Figure 2.9: Calibration error on test scene with 2 fixed decalibrations. Two examples of the distribution of the calibration error for decalibrations, which are fixed over the test sequence (5177 frames). Five networks are executed iteratively (20°/1.5 m, 10°/1.0 m, 5°/0.5 m, 2°/0.2 m and 1°/0.1 m). The left side of the box plots shows the translation error in meters. The right side shows the rotational error of yaw, pitch and roll (Y, P, R) in degrees. Adapted from [100].

For a quantitative evaluation on the test set, decalibrations are sampled in the range of /[−20°, 20°/] / /[−1.5 m, 1.5 m/].

To simulate a fixed calibration error, the decalibration is kept fixed for one pass of the test set. In total 100 runs on the whole test set with different decalibrations are performed. Both aforementioned strategies, iterative refinement and the median over the whole sequence, are applied. The approach achieves a mean angle error of 0.28° (yaw, pitch, roll: 0.24°, 0.25°, 0.36°) and a mean translation error of 6 cm (x, y, z: 7 cm, 7 cm, 4 cm). The results of the approach are visualized in Figure 2.1 and Figure 2.10. It can

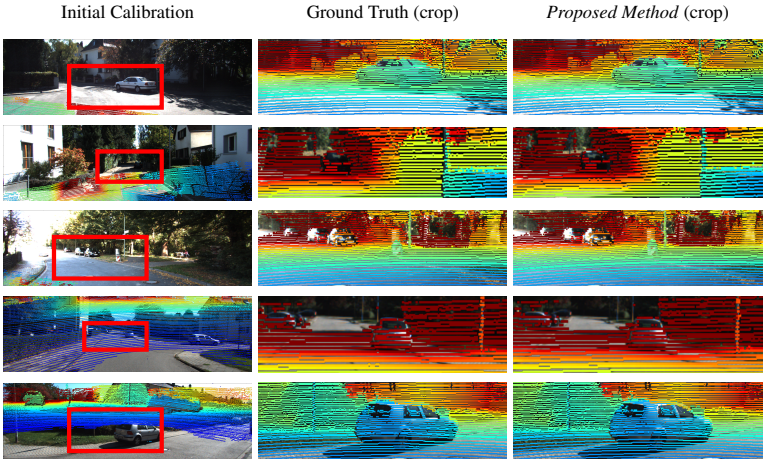


Figure 2.10: Result of iterative single shot calibration. Five networks, trained on different decalibration ranges (20°/1.5 m, 10°/1.0 m, 5°/0.5 m, 2°/0.2 m and 1°/0.1 m), are executed iteratively. Although the initial calibration is bad, the proposed method delivers accurate results. Adapted from [100].

be seen that the calibration approach can handle even large deviations from the actual ground truth.

2.5 Discussion

In this chapter a novel approach for extrinsic calibration of multimodal sensors based on a deep convolutional neural network was presented. Compared to existing approaches, a deep neural network replaces all conventional calibration steps (feature extraction, feature matching and global regression) and infers the 6 degrees of freedom of the calibration directly. In order to iteratively refine the calibration output, several expert networks are trained on different deviations from the ground truth calibration.

The presented approach can be used to handle different calibration tasks: on the one hand, a post-drive target-less calibration can be applied to obtain a calibration from scratch, where the only human interaction is to assure an initial calibration within 1.5 m for translation and 20° for rotation. This offline calibration is performed by using temporal information over the whole drive to reduce noise and reject outliers. On the other hand, the approach

is well-suited for online calibration by correcting small deviations in real-time. The process can be made more robust by applying a moving average or sliding window filter to smooth the calibration.

The calibration from scratch yields a mean calibration error of 6 cm for translation and 0.28° for rotation with decalibration magnitudes of up to 1.5 m and 20°. This competes with state-of-the-art online and offline methods.

3 Depth Completion

In this chapter the topic of depth completion is addressed, where sparse data with depth information is projected on the high resolution 2D image plane. Depth completion is a challenging problem, as there are only few pixels with valid information and reconstructing the missing values requires to either incorporate strong prior knowledge, or to interpret the additional information present in the according color image.

In order to address this challenge, we discuss two different approaches and further introduce a large-scale dataset and benchmark to allow training, evaluation and comparison of depth completion methods. The first approach, *Semantically Guided Depth Completion* (SGDC), is based on the work of Schneider et al. [101] in which the first two authors, Nick Schneider (the author of this dissertation) and Lukas Schneider, agreed to share the contributions equally.

The second approach, Sparsity Invariant CNNs (SICNN) and the large-scale depth dataset are based on the work of Uhrig et al. [111], whose contributions are equally shared between the first two authors, Nick Schneider (the author of this dissertation) and Jonas Uhrig.

The remainder of this chapter is structured as follows. In Section 3.1 the problem of depth completion is introduced, followed by an analysis of current state-of-the-art methods in Section 3.2. We will see that current methods have several shortcomings which is why two separate methods for depth completion are presented. The first is the aforementioned image-guided depth completion, which is described in Section 3.3. The second is the learned approach, which is presented in Section 3.4. For training and evaluation, a new dataset is developed in Section 3.5. Finally, both methods are evaluated in Section 3.6, followed by a discussion in Section 3.7.

3.1 Introduction

High resolution dense depth maps are used in many computer vision applications such as 3D reconstruction [1], scene flow computation [78] as well as

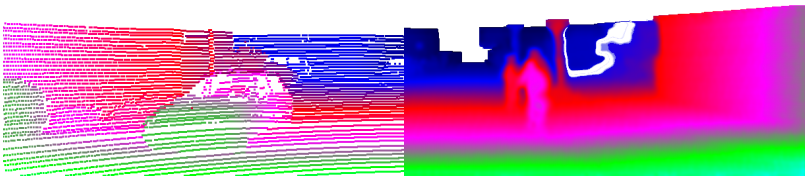


Figure 3.1: Visualization of the task of *Depth Completion*. Depth completion is the task of obtaining per pixel depth information (right) from sparse input (left). In this view the projected LIDAR points are dilated for illustration. Typically, the projected depth points contain only a fraction of the displayed points. Adapted from [111].

object detection [54] and tracking [107]. In order to obtain per pixel depth estimates, stereo vision has been the method of choice for a long time. However, classic stereo algorithms suffer from high computational complexity and, although there have been many advances thanks to deep learning in recent years, the accuracy of stereo methods is still bound by the image resolution and the focal length. More recently, Light Detection and Ranging (LIDAR) sensors have become increasingly popular in many applications due to their superior range accuracy and constant improvements in size and cost. However, the resolution is typically very limited and does not compare with the density of images. For indoor range sensing on the other hand, compact and inexpensive Time of Flight (ToF) cameras have become state-of-the-art as they provide per pixel depth information at high frame rates. Yet, the depth measurements suffer from acquisition noise as the illumination energy is limited. Moreover, the resolution is still not comparable to conventional RGB camera systems.

The task of obtaining dense depth maps from accurate low-resolution depth data, is referred to as *depth completion* and is illustrated in Figure 3.1. Many approaches use a high resolution RGB image to guide the completion process, yielding significant improvements, c.f. [26, 53, 71, 73]. Key assumption of most approaches is a correlation between intensity edges and depth boundaries of the projected point clouds. While leveraging this correlation improves the depth completion accuracy, there are two typical failure cases. First, highly textured surfaces are often misinterpreted as depth discontinuities and second, weak gradients at actual depth boundaries are ignored.

To this end, Section 3.3 proposes to guide the depth completion process not only by image edges but also by a pixel wise semantic segmentation image. This introduces a new constraint which encourages depth boundaries at *object* boundaries rather than at plain intensity edges. The presented approach incorporates both, image gradients and object boundaries, in an energy minimization which optimizes local planes. While the object boundaries improve the result, the method can also be used without the semantic part and does thus not rely on massive data for training or fine-tuning.

However, the approach also comes with some disadvantages. First, the energy minimization requires many parameters which have to be tuned manually and vary with the resolution of the inputs. Second, the semantic segmentation and edge detection are not specifically designed to guide the depth completion process and thus, some important depth discontinuities are still not modeled.

In recent years, more and more large datasets have become available, enabling machine learning methods, and especially Convolutional Neural Networks, to impact almost all areas in computer vision. CNNs are able to learn various complex tasks and have surpassed most of the classical computer vision algorithms in terms of speed and accuracy.

Considering depth completion as a learning task, CNNs can be directly applied to transform sparse point clouds into dense depth maps. This allows to learn the completion end-to-end without the need of modeling the individual steps as described in Section 3.3. However, CNNs are typically applied on dense regular grids and not on randomly sparse inputs. Thus, Section 3.4 considers the problem of learning convolutions which are invariant to sparsity in the input.

Although depth completion has been an active area of research in the past, there have not been many datasets for training and evaluation. Most of the approaches were therefore evaluated on the Middlebury dataset [98] which provides accurate high resolution depth images. However, the dataset was never designed for the task of depth completion and thus, the dense depth maps are typically sampled randomly to obtain sparse depth. This does not reflect the characteristics of an actual sparse depth sensor, neither in terms of noise nor in terms of data distribution. Furthermore, it contains only few images and is therefore not suited to train data intensive machine learning models such as deep neural networks. In order to circumvent the shortcomings of Middlebury, many learned depth completion approaches leverage artificial datasets such as Sintel [6] or Synthia [95]. While providing many

dense depth maps, the datasets do not contain data from an artificial LIDAR or ToF sensor. Furthermore, translating from synthetic to real world scenarios is still an open challenge as the level of realism is often not sufficient. To this end, a real world dataset is presented in Section 3.5, consisting of 93k semi-dense depth annotated images. The RGB images are accompanied by raw LIDAR scans which can serve as input for training and evaluating depth completion approaches.

3.2 Related Work

Completion of sparse information with or without the help of guidance data is an active area in optical flow, stereo, and Time of Flight (ToF) research. We discuss three different lines of related work. First, non-guided depth completion approaches are discussed in Section 3.2.1. Second, methods which leverage guidance by a high resolution RGB image are presented in Section 3.2.1. And last, learned depth completion are handled separately in Section 3.2.3 due to their increasing importance in the last few years.

3.2.1 Non-Guided Depth Completion

The task of depth completion is to assign a value to locations where no depth information is present. Non-guided approaches consider only the sparse depth information to estimate the missing values. This section gives a short overview on those approaches.

A straight-forward solution for non-guided depth completion is nearest neighbor interpolation, where to each location, the value of the closest location with a valid value is assigned. Instead of simply assigning existing values to missing locations, there exist several model-based interpolation strategies, such as linear interpolation, cubic interpolation or spline interpolation. They allow for a more smooth interpolation, but fail if there are depth gaps in the actual scene. More advanced early approaches have leveraged repetitive structures to identify similar patches across different scales in 2D [32, 74] and 3D [44]. However, they assume that the data is located on a regular grid and therefore cannot be applied for completing sparse and irregularly distributed 3D laser scan data as considered in this thesis.

3.2.2 Guided Depth Completion

In contrast to the non-guided approaches, there are several methods which leverage additional data to guide the depth completion, such as high resolution RGB images.

The underlying assumption is that the target domain shares commonalities with the guidance data, e.g. that image edges align with depth discontinuities.

A popular choice for guided depth completion is Bilateral Filtering [8, 20, 53, 71, 110, 123]. In general, it describes a mapping of an input image $P = \{p_j\}_j^{N_p} \in \mathbb{R}$ with N_p pixel to a target image $Q = \{q_i\}_i^{N_p} \in \mathbb{R}$, which involves a guidance image $I \in R^{N_p}$. For the case of depth completion, P is an image with sparse measurements. The filtered output of Q at pixel position i can then be described as a weighted average

$$q_i = \sum_j w_{ij}(I) p_j, \tag{3.1}$$

with w_{ij} being a weight function of the guidance image I at pixels i, j which is independent of p. The classical bilateral filter [110] considers the Euclidean distance between two pixels $\left\|x_i - x_j\right\|_2$ as well as their absolute intensity difference $\left|I_i - I_j\right|$. Typically, this is combined with a Gaussian kernel, resulting in

$$W_{ij}^{bf} = \frac{1}{K_i} e^{-\frac{(\left\|x_i - x_j\right\|_2)^2}{2\sigma_s^2}} e^{-\frac{(\left|I_i - I_j\right|)^2}{2\sigma_r^2}}, \tag{3.2}$$

where K_i is a normalizing parameter and σ_s and σ_r adjust the spatial and intensity similarity, respectively. While the original bilateral filter assumes that p and I relate to the same source image, Petschnigg et al. [86] introduce Joint Bilateral Filtering, where I is a separate guidance image which contains more detailed information than p. Next to the work of Tomasi [110] and Petschnigg [86] more advanced bilateral filtering versions exist, extending the formulation of w_{ij}. He et al. [39] assume a local linear model between the guidance image I and the filter output q, dropping the spatial correlation between the pixels. Liu et al. [71] define the weight function by geodesic distances which improves the depth completion especially at fine scene details. All aforementioned approaches have a limited receptive field which is bound by the kernel size. Therefore, they fail to reconstruct larger

gaps in the input data. To this end, Barron et al. [2] propose the Fast Bilateral Solver, which considers depth completion as an optimization problem in the bilateral grid. The grid is a remapping of the original pixels based on their color value and location. The solver iteratively refines the depth map, which allows handling very sparse input data. While the solver produces good results, it requires careful parameter tuning in order to achieve a meaningful mapping in the bilateral grid. Typical errors are that pixels with similar colors are assigned a similar depth value across the image.

Besides filter-based methods, there are many approaches which rely on a pre-segmentation of the RGB image via superpixels, combined with a post-processing which smooths adjacent segmentation regions. Lu et al. [73] segment the image using a graph-based algorithm. In a post-processing step, smoothing operations are performed on each segment. Matsuo et al. [75] interpret superpixels as tangent planes, which avoids typical surface flattening artifacts of filter-based methods. Both methods rely on a good initial superpixel segmentation, which is independent of the sparse depth values. This often results in depth map errors, especially at fine structures.

More complex approaches are based on a global energy minimization [18, 26, 84, 92]. Diebel et al. [18] use a Markov Random Field (MRF) with the sparse depth measurements as data term and exploit features of the high resolution image in the smoothness term. Park et al. [84] incorporate multi-cue neighborhood weighting and non-local means regularization into an MRF in order to preserve fine local details. Ferstl et al. [26] formulate a convex optimization problem which leverages an anisotropic diffusion tensor to handle fine structures. Furthermore, they include a regularization term which encourages piecewise affine solutions. In general, all global minimization approaches produce smooth results and some are also able to handle fine structures. However, the optimizations are computationally expensive and require many parameters.

3.2.3 Deep Learning for Depth Completion

The approaches presented in the previous section rely on the pure intensity and color information of the guidance image. By additionally considering the context of the sparse depth data itself or the semantics of the guidance image, the accuracy could be further enhanced. However, extracting this kind of information requires powerful algorithms. Hence, in this section we

discuss solutions which leverage the capabilities of deep learning to address the depth completion problem.

Deep Learning Methods for Depth Completion In the past few years, traditional techniques for depth and image superresolution have been surpassed by deep-learning-based methods in terms of accuracy and efficiency [21, 22, 56, 93, 122]. Some approaches exploit end-to-end models for depth completion with the help of a high resolution guidance image [47, 104]. However, the aforementioned methods assume that the input data is located on a regular grid and therefore cannot be applied to less structured depth data e.g. from a LIDAR sensor.

More recently, after publishing the depth benchmark presented in this dissertation, several methods for completing projected LIDAR point clouds were proposed [10–12, 90, 112, 120]. Their results exceed previous non-learned methods by a large margin. However, none of the aforementioned methods consider that the data, the CNNs operate on is randomly sparse. In fact, only few methods considered varying sparsity in the input. In contrast, the sparse convolution module presented in this dissertation is invariant to the level of sparsity in the input.

CNNs with Sparse Inputs Most naïve approaches handle sparsity in the input by either zeroing the invalid values by providing an additional mask to the network which encodes the validity of each pixel. For object detection, Chen et al. [9] and Li et al. [66] project depth from a LIDAR scanner onto a low resolution image, zero the missing values and run a standard CNN on this input. Zweig et al. [128] and Köhler et al. [60] pass an additional binary validity mask to the network to solve the tasks of optical flow interpolation and inpainting respectively. In Section 3.6.2 we will see that both strategies are suboptimal compared to explicitly considering sparsity inside the convolution layers.

Jampani et al. [51] formulate bilateral filters as layers which are incorporated into a CNN. Accordingly, they are able to learn the parameters of the corresponding permutohedral convolution kernel. The layer can handle sparse irregular inputs but requires guidance information to construct an effective permutohedral representation and is computationally expensive for large grids. Compared to their approach the sparse convolutional layers pre-

sented in Section 3.4 yield significantly better results for depth completion while being more efficient.

Additionally, Graham [35, 36] and Riegler et al. [94] consider sparse 3D inputs to improve computational efficiency and memory demands by partitioning the space according to the input. However, they employ regular convolution layers which suffer from the same drawbacks as the naïve approach described above.

Sparsity in CNNs Several works [27, 37, 70, 85, 116] consider sparsity *within* convolutional neural networks. Liu et al. [70] show how to reduce parameter redundancy by using a sparse decomposition. This eliminates more than 90% of the parameters while reducing the accuracy by less than 1%. Wen et al. [116] propose a regularization of filters, channels and layer depth to learn a structured sparsity which is hardware-friendly. They report speed-up factors of 3 to 5 w.r.t. regular CNNs. While these works focus on improving efficiency of neural networks by exploiting sparsity *within* the network, they do not consider the problem of sparse *inputs*. A combination of the two lines of work will be an interesting direction for future research.

Invariant Representations Learning models which are robust to variations is a repeatedly occurring issue in computer vision. A straight-forward solution to handle variations in the input data is augmentation [61, 63, 102]. More recently, *geometric* invariance such as rotation or perspective transformation has been incorporated directly into the filters of CNNs [15, 42, 50, 118, 127]. In section Section 3.4 the problem of learning representations invariant to the *level of sparsity* are considered. This allows to vary the density of the input point clouds which is necessary if training and execution of the network are performed on sensors with different resolutions.

3.3 Semantically Guided Depth Completion

In this section, an image guided sparse depth completion approach is presented. The completion task is formulated as a global energy minimization problem, which is guided by a geodesic distance that considers both object-based edge information and pixel-wise semantic class labels. The geodesic distance establishes meaningful relations between pixels with and without

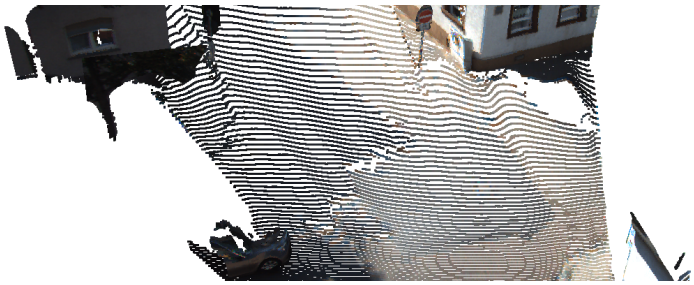


Figure 3.2: Semantically guided depth completion. Example output of the Semantically Guided Depth Completion (SGDC) approach. The result is smooth and accurate, while fine structures, e.g. the pole and the sign, are preserved. Adapted from [101].

sparse depth measurements. In order to be computationally efficient, the observed scene structure is modeled as a set of locally planar elements which allows for an efficient yet accurate approximation of the energy minimization problem. A thorough evaluation on several public datasets with different types of depth data demonstrates the benefits of the approach, particularly in terms of efficiently computing globally consistent solutions while preserving fine structures and sharp depth boundaries.

The remainder of this section is structured as follows. First, the depth optimization problem is described in Section 3.3.1. In order to acquire meaningful context for each pixel, a gradient and semantic aware geodesic distance is derived in Section 3.3.2. Next, the geodesic distance is approximated in Section 3.3.3, dropping most of the free parameters of the energy function. This then allows for an efficient optimization which is presented in Section 3.3.4. Finally, experiments and ablation studies are discussed in Section 3.6.1.

3.3.1 Optimization Problem

Image-based depth completion aims at estimating a depth value for each pixel in a high resolution image, while measurements are only available for a subset of these pixels. The challenge of computing an accurate depth map is to consider piecewise smooth surfaces while retaining fine detailed structures. A common strategy is to model the surface by a set of planes which are constrained by their relationship. By enforcing local smoothness but still allowing depth gaps between adjacent planes, both piecewise consistent surfaces and detailed structures can be accurately modeled. SGDC therefore uses

a plane formulation to solve the task of depth completion. This transfers the problem from estimating a per pixel depth to estimating per pixel plane parameters. Finding the best set of plane parameters is a major challenge as it requires to consider many inputs such as sparse depth measurements and dense RGB appearance features. At the same time, the parameters need to be constrained to encourage smooth surfaces. To this end SGDC treats depth completion as an energy minimization problem.

The energy formulation consists of three parts. First, a unary term which encourages the planes to align with an existing depth measurement, if available. Second, a pairwise term which penalizes inconsistencies to neighboring planes. And third, an additional term that considers the probability of existing measurements being outliers.

To describe the energy formulation in detail, first, a sparsely populated depth map with pixels $i \in I$ is considered. In order to estimate the inverse depth $\frac{1}{z}$ for each pixel in the dense target depth map, plane parameters $\theta_i \in \theta$ representing plane normals are assigned to each pixel i. The function $\theta_i(i)$ evaluates to the inverse depth given the plane parameters θ_i at pixel i via $\theta_i \cdot [u_i, v_i, 1]^T$. To solve for the plane parameters, an energy function $E(\theta, \mathbf{o}, \mathbf{c}, \mathbf{z})$ is formulated, where $\mathbf{o}$ are outlier flags, $\mathbf{c}$ are coplanarity flags and $\mathbf{z}$ are sparse depth measurements. Additionally, $\mathscr{M}$ denotes the set of all pixels to which a depth measurement z_i is assigned. This results in the following energy function:

$$\begin{aligned} E(\theta, \mathbf{o}, \mathbf{c}, \mathbf{z}) = & \sum_{i \in I} E_{una}(\theta_i(i), o_i, z_i) + \\ & \sum_{i,j \in I} w_{i,j} \cdot E_{pair}(\theta_i(i), \theta_j(i), c_{i,j}) + \\ & \sum_{i \in I} E_o(o_i), \end{aligned} \tag{3.3}$$

where $w_{i,j}$ are pairwise weights between any two pixels i and j which consider relationships, e.g. given by the RGB guidance image. In the following, the individual energy formulations are explained in detail.

Unary Term The unary term enforces consistency to an existing sparse depth measurement. However, in most cases, no measurement is available and so the unary term is discarded. Additionally, if the measurement has been flagged as an outlier, the unary term is also discarded. This leads to the following formulation of the unary term:

$$E_{una}(\theta_i(i), o_i, z_i) = w_{una} \cdot \begin{cases} \left(\theta_i(i) - \frac{1}{z_i}\right)^2 & \text{, if } i \in \mathscr{M} \wedge o_i = 0 \\ 0 & \text{, otherwise.} \end{cases} \tag{3.4}$$

Pairwise Term The pairwise term encourages coplanarity by penalizing the L_2 distance of the inverse depth at pixel i evaluated by the two different plane functions $\theta_i(i)$ and $\theta_j(i)$ via

$$E_{pair}(\theta_i(i), \theta_j(i), c_{i,j}) = w_c \cdot \begin{cases} (\theta_i(i) - \theta_j(i))^2 & \text{, if } c_{i,j} = 1 \\ \lambda_c & \text{, otherwise,} \end{cases} \tag{3.5}$$

with a constant penalty λ_c for non-consistent planes ($c_{i,j} = 0$). By definition, the coplanarity flag is optimized to 1 if $(\theta_i(i) - \theta_j(i))^2$ is smaller than a predefined parameter. The choice of this parameter is critical to allow depth discontinuities but at the same time encourage smoothness. Furthermore, it is worth noting that with this formulation only, the energy minimization does not consider texture or semantics provided by the high resolution RGB image. This will be considered in the next section.

Outlier Term Finally, the outlier term sets a constant penalty $\lambda_o \in \mathbb{R}$ if a measurement is set to be an outlier $o_i = 1$ as

$$E_o(o_i) = \begin{cases} \lambda_o & \text{, if } i \in \mathscr{M} \wedge o_i = 1 \\ 0 & \text{,otherwise.} \end{cases} \tag{3.6}$$

As the unary term considers the distance from plane to measurement only if $o_i = 0$, the two energy functions $E_{una}()$ and $E_o()$ are involved in optimizing for $\mathbf{o}$. The choice of λ_o and the unary weight w_{una} balance how far a measurement z_i can diverge from $\theta_i(i)$ until it is cheaper to set $o_i = 1$.

3.3.2 Geodesic Distance

The key challenge of the proposed approach is to model the influence of neighboring pixels onto the currently considered pixel. In Equation (3.5) only coplanarity is regarded, discarding any other information such as edges

or semantics given by the high-resolution RGB image. To better model the relationship between pixels, $w_{i,j}$ was defined in Equation (3.3). If the pairwise weight is set to 1 the coplanarity term is influencing the energy function at its maximum, while setting the weight to zero diminishes its influence. Weighing pixels based on a certain distance measure $D_{i,j}$ is a common computer vision task and there exist many approaches. We already explored bilateral filtering in Section 3.2. It considers pixel distances in the image space as well as intensity measurements for the pairwise weight. Bilateral filtering is great in smoothing larger homogeneous areas with only few artifacts as well as in separating data which have a significant different appearance in the guidance image. However, one of the key issues of bilateral filtering is that the influence of another pixel gets small if the pixel is very far away in the image space. This is especially critical, if only very few valid measurements are available. Additionally, there can also be negative influences of nearby pixels which are similar in appearance but resemble a different object with another depth. In order to address these shortcomings a new distance measure needs to be defined which, in addition to appearance, also considers the object semantics. With the goal to preserve depth discontinuities and tiny structures, an edge- and semantics-aware geodesic distance is introduced to guide the depth completion even if only few depth measurements are available. Geodesic distances are typically used in geometry and define the shortest path from one point to another on a surface. More general, the geodesic distance can also be described as a path π which accumulates minimal costs. Applied to the image space, we can interpret the geodesic distance as the path with minimal costs from pixel i to pixel j. Defining a good cost term is critical to guide the depth completion. Intuitively, a path should have higher costs if it crosses either an object border or if it is further away in the image space. Therefore, a good choice is to consider edge costs $c_E(\pi)$ and semantic costs $c_S(\pi)$ to describe an object border as well as costs $c_T(\pi)$ which penalize the length N_π of a considered path in the image space:

$$D_g(i,j) = \min_{\pi} \frac{w_E * c_E(\pi) + w_S * c_S(\pi) + w_T * c_T(\pi)}{w_E + w_S + w_T} + \varepsilon. \tag{3.7}$$

A small value ε is added to avoid $D_g(i,j)$ becoming zero. In the following, the different cost terms are described in detail: The *edge cost term* $c_E(\pi) \in$

$[0,1]$ accumulates high costs if the path traverses pixels which have a strong gradient in the absolute gradient image $G \in [0,1]$:

$$c_E(\pi) = \frac{1}{N_\pi} \sum_{k \in \pi} G(k). \tag{3.8}$$

The *semantic cost term* $c_S(\pi) \in [0,1]$ increases if semantically different pixels are traversed. In order to determine a pixel's semantic the pseudo probabilities of a high resolution semantic segmentation are used [72]. Considering a specific label $l \in L$, high semantic costs $c_S(\pi)$ are collected if the pseudo probability for this class at pixel k $p(l,k)$ is small. Therefore, finding the label with maximal probability for a given path results in the lowest cost. This can be written as

$$c_S(\pi) = 1 - \max_l \prod_{k \in \pi} p(l,k). \tag{3.9}$$

The *traverse cost term* $c_T(\pi) \in [0,1]$ simply considers the amount of pixels N_π traversed

$$c_T(\pi) = \frac{1}{N_\pi} \sum_{k \in \pi} 1. \tag{3.10}$$

As the individual cost terms range between $[0,1]$ it follows that $D_g(i,j) \in (0,1]$ This leads to

$$D_g(i,j) \in (0,1] \Rightarrow w_{i,j} = -log(D_g(i,j)) \in \mathbb{R}^+, \tag{3.11}$$

resulting in larger weights for small geodesic distances.

3.3.3 Approximation: From Pixel to Planes

Optimizing the energy function, which considers unary and pairwise terms for each pixel, is computationally inefficient. Therefore, Revaud et al. [91] propose an approximation $\hat{D}_g(i,j)$ of the geodesic distance described in Equation (3.7). It allows to compute all pixel distances more efficiently and with that simplifies the energy function significantly. To this end, for each pixel i the nearest pixel with a valid measurement (measurement pixel) $n_i \in \mathscr{M}$ is determined, by calculating the geodesic distance D_g to it. Note that this requires far fewer computations than calculating the distance to all available pixels, especially if the amount of pixels with valid measurements

is low as in the case of depth completion of sparse LIDAR point clouds. By assigning each pixel to the closest measurement pixel, we receive Voronoi cells $\mathcal{V}_i$ with measurement pixels as seeds. This is depicted in Figure 3.3. The Voronoi cells enable several approximation steps which will simplify the computation without sacrificing too much accuracy. The first approximation step is to constrain the path from pixel i to pixel j such that it traverses the seeds of the Voronoi cells. This allows to split the distance function in three parts, resulting in

$$\hat{D}_g(i,j) = D_g(i,n_i) + D_g^g(n_i,n_j) + D_g(j,n_j). \tag{3.12}$$

In the first part the distance from pixel i to its Voronoi seed n_i is denoted as $D_g(i,n_i)$. The second part defines the geodesic distance from the closest Voronoi seed of pixel i to the closest Voronoi seed of pixel j, denoted as $D_g^g(n_i,n_j)$. This can still be computationally expensive, which is why it is further restricted to only traverse through a distance graph constructed of only measurement pixels. The graph-based distance $D_g^g(n_i,n_j)$ between measurement n_i and measurement n_j can be efficiently computed using the Dijkstra algorithm. For more details, consult the work of Revaud et al. [91]. The last part of $\hat{D}_g(i,j)$ describes the geodesic distance of the pixel j to its corresponding Voronoi seed n_j.

The distances to the closest measurement pixels $D_g(i,n_i)$ and $D_g(j,n_j)$ are typically small (assuming a reasonable number of measurements). Thus, the sum of $D_g(i,n_i)$ and $D_g(j,n_j)$ can be further approximated by a very small constant ε, leading to

$$\hat{D}_g(i,j) \approx D_g^g(n_i,n_j) + \varepsilon. \tag{3.13}$$

One of the main advantages of the proposed approximation is a significant reduction of parameters which need to be optimized.

Considering two pixels i,j of the same cell $\mathcal{V}_k$, after the approximation, the geodesic distance between those pixels simplifies to

$$\hat{D}_g(i,j) = \varepsilon. \tag{3.14}$$

From $\varepsilon \to 0$ results $\hat{D}_g(i,j) \to 0$ and thus the negative logarithm in Equation (3.11) causes large pairwise weights $w_{i,j} \to \infty$. From Equation (3.3) we can now directly observe that all weights between pixels of the *same*

Figure 3.3: Semantic aware geodesic distance. Given an input image the pixel-wise edge costs and semantics are calculated. The edge and semantic cues are combined to calculate a geodesic distance between each pixel. The approximation in Section 3.3.3 assigns to their geodesic closest measurement resulting in Voronoi cells. The geodesic distance is then used to determine a graph that connects semantically related measurements (the yellow line on the right image shows the 20 closest measurements to the measurement selected in green). Adapted from [101].

cell $w_{i,j}$ with $i,j \in \mathscr{V}_j$ cause the energy function $E(\theta,\mathbf{o},\mathbf{c},\mathbf{z})$ to be extremely large if the corresponding pairwise energy $E_{pair}(\theta_i(i),\theta_j(i),c_{i,j})$ does not approach 0. Hence, the solution of the energy function forces $\theta_i(i) = \theta_j(i)$ if $\varepsilon \to 0$. Since this observation holds for all pairs of pixels in a cell, the plane parameters θ_i and θ_j have to be identical and can thus be summarized by θ_{n_i}.

Regarding two *different cells* with corresponding measurement pixels n,m and picking any pair of pixels i,j from those, each of the pairwise weights can be replaced by $w_{n,m}$. We can thus reorder the summation and write

$$\begin{aligned}&\sum_{i,j\in I} w_{i,j}\cdot E_{pair}(\theta_i(i),\theta_j(i),c_{i,j}) = \\ &\sum_{n,m\in\mathscr{M}} w_{n,m}\cdot \sum_{p\in\mathscr{V}_n} E_{pair}(\theta_n(p),\theta_m(p),c_{n,m}).\end{aligned} \tag{3.15}$$

Also, the unary and outlier terms can be simplified as they evaluate to zero at pixels without valid measurements, leading to

$$\sum_{i\in I} E_{una}(\theta_i(i),o_i,z_i) = \sum_{n\in\mathscr{M}} E_{una}(\theta_n(n),o_n,z_n) \tag{3.16}$$

and

$$\sum_{i\in I} E_o(o_i) = \sum_{n\in\mathscr{M}} E_o(o_n). \tag{3.17}$$

Note that instead of summing the energies over all pixels of the image $i \in I$, only the energies at pixels which valid measurements $n \in \mathscr{M}$ are considered.

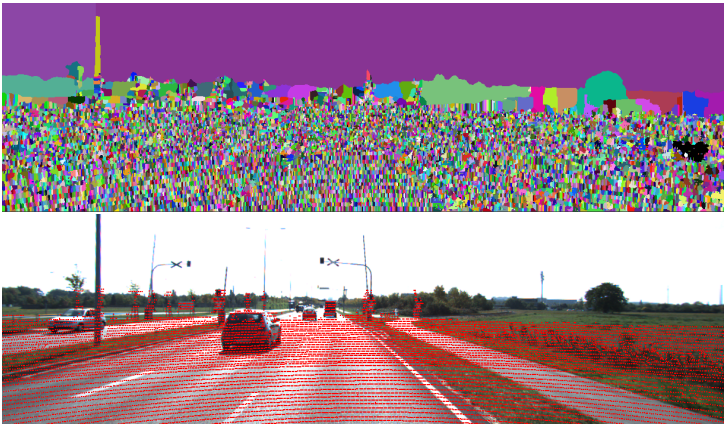


Figure 3.4: Approximation from pixels to planes. Instead of optimizing over all pixels, which is computationally intractable, the approximation in Section 3.3.3 leads to the displayed geodesic Voronoi cells. Pixels within this cell share the same plane parameters. Adapted from [101].

Combining Equations (3.15) to (3.17), the overall energy function results in

$$\begin{aligned} E(\theta, \mathbf{o}, \mathbf{c}, \mathbf{z}) = & \\ & \sum_{n \in \mathscr{M}} E_{una}(\theta_n(n), o_n, z_x) + \\ & \sum_{n,m \in \mathscr{M}} w_{n,m} \cdot \sum_{p \in \mathscr{V}_n} E_{pair}(\theta_n(p), \theta_m(p), c_{n,m}) + \\ & \sum_{n \in \mathscr{M}} E_o(o_n). \end{aligned} \tag{3.18}$$

Regarding Equation (3.18) it becomes apparent that the energy function is now defined over $|\mathscr{M}|$ cells instead of over all pixels. This reduces the number of free parameters in the optimization significantly.

Figure 3.4 shows a visual representation of the approximation. As the pixels within a cell share the same plane parameters, they can be grouped to superpixels. Those superpixels are then treated as plane patches which are encouraged to be coplanar if no depth jump or outlier was detected.

To further reduce computational complexity, in practice, pairwise costs are only calculated for the N closest cells. Additionally, only cells with maximal distance $\hat{D}_g < D_{max}$ are considered. All other pairwise weights are

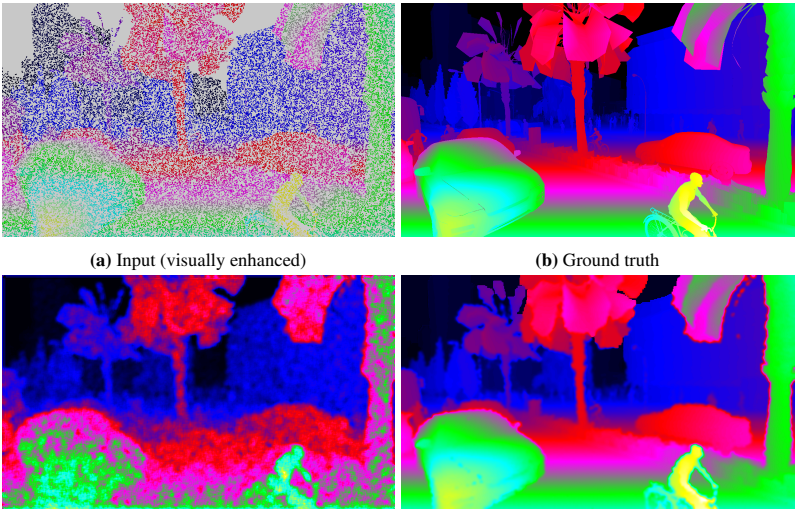
set to $w_{n,m} = 0$. This leads to a consistent set of cells which allows for an efficient, robust yet accurate depth optimization.

3.3.4 Solving the Optimization Problem

While the previously presented approximation of the energy in Equation (3.18) allows for a more efficient optimization, solving the optimization problem still remains difficult, as it contains a large set of discrete and continuous variables. A similar problem was present in the work of Yamaguchi et al. [121], in which they optimized superpixel planes for stereo estimation by iteratively optimizing coplanarity, hinge and occlusion flags as well as plane parameters one after another. By adapting their approach to the optimization problem at hand, this allows solving for all variables effectively by gradually refining them with each iteration. One iteration in the optimization consists of two steps which are consecutively executed.

Due to the convexity of the energy in θ, one plane parameter can be efficiently solved in closed form, assuming the rest of the discrete variables (coplanarity flags $\mathbf{c}$ and outlier flags $\mathbf{o}$) and other plane parameters are fixed. Thus, in the first step, all other parameters are fixed and θ are optimized in parallel for all planes. By using polynomial coefficients to evaluate the unary and pairwise potentials, this can be done in constant time. Second, the discrete outlier and coplanarity flags are optimized in parallel. Both steps are iteratively applied to continuously improve the depth estimation. Note that there is no guarantee for the energy to decrease in each iteration. However, later experiments show that the measured depth error decreases consistently with an increasing number of iterations, c.f. Section 3.6.1.

An iterative optimization scheme typically requires a good initialization. Yet, throughout most of the later experiments, only a simple initialization is applied by setting $o_i = 0, c_{i,j} = 1$ and $\theta_i = [0,0,z_i]$. With enough iterations, the optimization still converges to satisfying results. However, computation time can be especially critical for some applications, such as autonomous driving. Therefore, a second initialization is used if label input is available. Cells dominantly consisting of pixels belonging to the ground class (e.g. road, sidewalk, grass) are initialized with upright normals. Cells containing objects (e.g. cars, pedestrians, walls, buildings) are initialized with normals parallel to the principal axis. Given this initialization, the number of iterations needed for convergence decreases significantly.



(a) Input (visually enhanced) **(b)** Ground truth

(c) Standard CNN **(d)** SICNN

Figure 3.5: Depth Map Completion with CNNs. Using sparse, irregular depth measurements Figure 3.5a as inputs leads to noisy results when processed with standard CNNs. In contrast, SICNN, the approach presented in this thesis, predicts smooth and accurate depth maps by explicitly considering sparsity during convolution. Adapted from [111].

3.4 Sparsity Invariant CNNs

In this section an alternative depth completion approach is presented. It is based on the recent success of convolution neural networks. One of the key challenges in learning depth completion is how the sparse depth data can be represented and processed. Still, the most popular choice is to project the sparse point clouds onto the high resolution image, resulting in a depth map where only a small percentage of the image contains valid information. As typically CNNs process dense and complete image data, we analyze the effect of presenting sparse and irregular input to CNNs.

Several works choose a naïve approach to handle sparse data by assigning a distinct value to all non-informative pixels [9, 66].

Unfortunately, this approach might lead to unsatisfactory results, especially if the network has limited capacity. In conventional CNNs, filter kernels of the first layer typically reflect orientation, color and frequency of

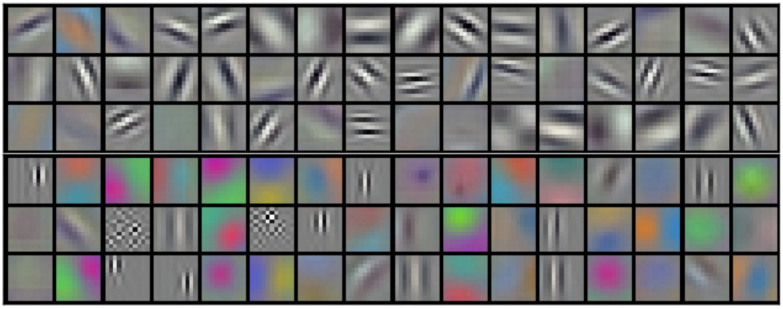


Figure 3.6: Conventional CNN Filter Kernels. In conventional dense image-based CNNs, learned kernels in the first layer typically describe certain frequency and orientation of image structures as well as different colors [61]. Applying one of those kernels to sparse input data would result in different activations depending on the density of the input instead of the underlying image structures. Source: [61]

image pixels as can be seen in Figure 3.6. However, in the case of varying input density, it is impossible to distinguish if a certain activation was caused by the underlying image structure or the level of sparsity it was presented. It therefore becomes less clear how the convolution operation should be defined.

In this section, we investigate a simple yet effective sparse convolution layer which outperforms the naïve approach and allows using lower capacity networks with a high convergence rate. In particular, the elements of the convolution kernel are weighed according to the validity of the input pixels. Additionally, a mask is generated in each filter step in order to share information about the validity of pixels with subsequent layers of the network. This enables to handle large levels of sparsity without significantly compromising accuracy. Most importantly, this representation is invariant to the level of sparsity which is presented to the convolutional layer.

Experiments show that training a network that uses sparse convolution layers, allows to be inferred a sparsity level different from the sparsity level at training time without significantly deteriorating the results. This is especially important in the context of robotics and especially autonomous driving where algorithms must be robust to changes in sensor configuration.

The remainder of this section is structured as follows. First, the problem of feeding sparse data into standard CNNs is described in Section 3.4.1. Next, in Section 3.4.2, two naïve approaches to this problem are analyzed. Finally,

sparse convolutions are introduced in Section 3.4.3 in order to tackle the weaknesses of the previously described naïve approaches.

3.4.1 Problem Formulation

In order to find a mapping f from an input image domain $\mathscr{X} \in \mathbb{R}^{W \times H}$ (e.g. depth, intensity) with width W and height H to an output image domain $\mathscr{Y} \in \mathbb{R}^{W \times H}$ (e.g. depth, semantics), CNNs have become one of the preferred implementations in the last few years.

A multi-layer CNN consists of several hidden layers in the form of convolutional layers which are stacked to allow complex feature representations. Applying a standard convolutional layer in a CNN to an input $\boldsymbol{x} = \{x_{u,v}\}_{u,v=0}^{WxH} \in \mathscr{X}$ results in

$$f_{u,v}(\boldsymbol{x}) = \sum_{i,j=-k}^{k} x_{u+i,v+j} w_{i,j} + b, \tag{3.19}$$

with 2k+1 being the kernel size and w and b being the weight and bias, respectively. Note that if the input contains multiple channels, $x_{u,v}$ and $w_{i,j}$ become vectors whose length depends on the number of given input channels.

While for most applications of CNNs the inputs are complete and dense images, in this section, we consider the case, where the input $\boldsymbol{x} = \{x_{u,v}\} \in \mathscr{X}$ consists of only partially observed depth values. In the next section we will discuss how standard convolutions can be applied in a naïve way to this problem.

3.4.2 Naive Approach

When applying CNNs to irregular sparse input, there are two naïve ways to deal with unobserved inputs. One is to encode the invalid inputs $x_{u,v}$ with a default value, e.g. zero or a negative value like -1. The idea is that during network training, the CNNs learn to distinguish between observed and unobserved inputs and respective weights are learned to consider both, sparsity and information input. However, this is a difficult task as the number of possible sparsity patterns grows exponentially with the kernel size which leads to longer convergence times and requires more network capacity. Furthermore, CNNs trained on a specific level of sparsity tend to overfit, resulting in degraded performance if applied to inputs with another sparsity level.

Another popular approach to deal with sparse irregular input is to provide a mask $\boldsymbol{o} = \{o_{u,v}\}$ which indicates if an input is observed ($o_{u,v} = 1$) or not ($o_{u,v} = 0$). Again, the idea is that during the training process the network learns to distinguish between observed and unobserved input. There are several ways to process the mask together with the input image. One is to concatenate both $o_{u,v}$ and $x_{u,v}$, adding the mask as an extra channel. A second approach is to process $o_{u,v}$ in a separate stream and combine it with the sparse input stream at a later stage. While the map theoretically allows the whole network to be robust to varying levels of sparsity, the convolutional layer itself is still not directly invariant to sparsity. This results in longer convergence times and still tends to overfit if only certain sparsity levels are present during training.

As evidenced by experiments, the described naïve approaches struggle to obtain a robust representation (see Section 3.6). Therefore, an alternative type of convolution is presented in the next section.

3.4.3 Sparse Convolutions

In the previous section, the main disadvantage of the naïve approaches was to rely on the training process to handle and distinguish missing data. Therefore, a convolution operation is introduced in this section which explicitly considers sparsity. The main motivation of the approach is to render the filter output invariant to the actual number of observed inputs by evaluating only observed pixels and normalizing the output appropriately:

$$f_{u,v}(\boldsymbol{x}, \boldsymbol{o}) = \frac{\sum_{i,j=-k}^{k} o_{u+i,v+j}\, x_{u+i,v+j}\, w_{i,j}}{\sum_{i,j=-k}^{k} o_{u+i,v+j} + \varepsilon} + b. \tag{3.20}$$

To avoid division by zero if no input pixels $x_{u+i,v+j}$ are observed, a small ε is added to the denominator. It is important to note that the sparse convolution (3.20) evaluates to a (scaled) standard convolution if the input is fully observed.

In contrast to techniques which first interpolate the sparse input before processing [93], the proposed approach operates directly on the sparse data and therefore omits to introduce additional distractors.

As modern CNNs typically consist of several convolutional layers, it is important to handle sparse input not only in the first but in all subsequent layers. Therefore, it is essential to track the visibility state and provide it

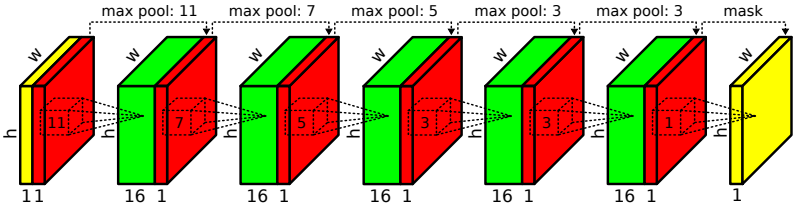


Figure 3.7: Sparsity Invariant CNN. The Sparsity Invariant CNN (SICNN) consists of several layers with Sparse Convolution Modules (see Figure 3.8) with decreasing kernel sizes from 11×11 to 3×3. Inputs to the CNN are a sparse depth map (in yellow) and a binary observation mask (in red). Adapted from [111].

to the next layers of the network. In particular, we can state that an output location carries information if at least one input of the filter was observed. On the contrary, if none of the filter's input was observed it can be stated that the output is also still unobserved. The resulting mask $f^o_{u,v}(\boldsymbol{o})$ after one sparse convolution can thus be determined via the max pooling operation

$$f^o_{u,v}(\boldsymbol{o}) = \max_{i,j=-k,..,k} o_{u+i,v+j}. \tag{3.21}$$

$f^o_{u,v}(\boldsymbol{o})$ evaluates to 1 if at least one of the inputs was observed and 0 otherwise. A graphical illustration of the *Sparse Convolution Module* is given in Figure 3.8.

In a multi-layer CNN the output of the sparse convolution (Equation (3.20)) and the max pooled mask then serve as an input to the next sparse convolutional layer. Like this the layers can be stacked to get a Sparsity Invariant CNN (SICNN). In the following, a very lean and simple CNN is used which is depicted in Figure 3.7.

Skip Connections In the previous sections the classical convolution operation was analyzed, and a version was derived which considers sparse input data. However, modern CNNs comprise many concepts and mathematical operations which cannot be applied to sparse data without modification. In this paragraph, especially skip connections are regarded as they are widely used in modern networks such as ResNet [40], GoogleNet [105] or VGG [103]. Skip connections combine multiple layers into a new layer e.g., by summation, multiplication or concatenation. To consider sparse input data, additional observation indicators can be used to define a normalized oper-

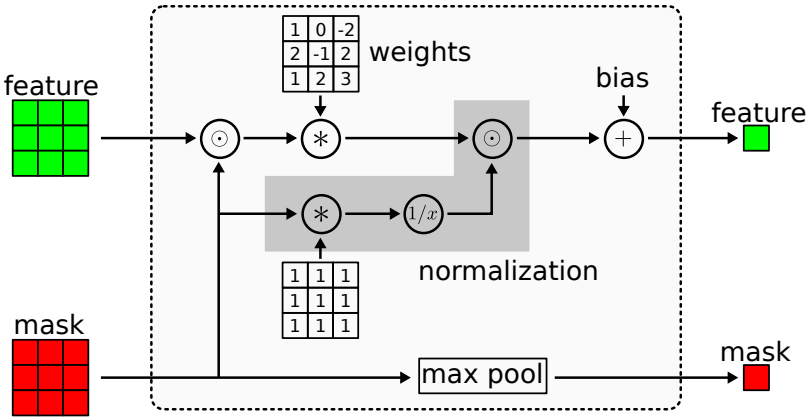


Figure 3.8: The Sparse Convolutional Module. Schematic of the sparse convolution operation. Here, $\odot$ denotes elementwise multiplication, $*$ convolution, $1/x$ inversion and "max pool" the max pooling operation. The input feature can be single channel or multi-channel. Adapted from [111].

ation over the observed inputs x^l. In the case of a summation this can be redefined as

$$f^+(\boldsymbol{x}, \boldsymbol{o}) = \frac{\sum_{l=1}^{n} o_l x_l}{\sum_{l=1}^{n} o_l}, \tag{3.22}$$

where n denotes the number of input layers. As in the previously defined sparse convolution, this expression simplifies to the standard operation $\sum_{l=1}^{n} x_l$ if all input variables were observed.

Note that in this dissertation skip connections are used for the application of semantic segmentation from sparse depth data. The network used for depth completion as pictured in Figure 3.7 does not use any skip connections.

3.5 Large-Scale Depth Dataset

In order to train and evaluate the proposed depth completion approaches, ground truth depth maps are required at a large scale. Synthetic datasets [28, 76, 95] offer the possibility of rendering a vast number of perfect depth ground truth data. This allows to conduct experiments under perfect conditions, which can be important to e.g. prove theoretical claims (there are

e.g. experiments on synthetic data to show the benefit of sparse convolution layers in Section 3.6). However, it remains an open question if this allows to conclude about the algorithm's behavior in challenging real world scenes which are measured by sensor with different noise characteristics. Especially if completing raw sensor depth measurements like e.g. LIDAR scanner data projected on a high resolution image, both raw sensor input data and high resolution ground truth depth maps are required. Another important aspect of evaluating state-of-the-art approaches is to make the data available to the research community and allow for a fair comparison of existing approaches. Especially in the last few years public benchmarks like ImageNet [96] and Cityscapes [16] have shown that it is important to compare methods by using standardized inputs and metrics.

Summarizing the previous mentioned requirements results in a list of desired features for a depth completion dataset:

- **Large Scale:** The dataset should include enough samples to allow training and a proper evaluation.
- **Accurate Ground Truth:** The ground truth depth maps should provide a high accuracy and reliability.
- **Real Sensor Input:** The data which needs completion should origin from a real sensor in order to get realistic sensor characteristics.
- **Public Availability:** The dataset should be made publicly available to be able to reproduce the results.
- **Public Benchmark:** To allow for a fair comparison of depth completion approaches a public benchmark would be preferable.

Unfortunately, existing datasets mostly lack at least one of the mentioned requirements: The Middlebury benchmark [98, 99] provides highly accurate depth estimates, however there exist only a dozen images in controlled laboratory conditions. In order to obtain sparse data for depth completion, the high resolution depth maps are typically subsampled and, if desired, noise is added to the measurements. The resulting synthetic sensor measurements are comparable to characteristics of a time of flight camera, but it is questionable if the artificial noise does reflect actual real world sensor models. Next, the Make3D dataset [97] consists of 500 outdoor images. The ground truth depth maps have a lower resolution than the corresponding provided

RGB images and also no raw sensor data input for depth completion is provided. While 500 images are more than the previously mentioned Middlebury dataset, it is still relatively few in order to train high-capacity deep neural networks. Another commonly used dataset for depth completion was presented by Ferstl et al. [26]. The dataset to the paper provides noisy low resolution input data as well as high resolution accurate ground truth depth and corresponding RGB images. While this is a dataset specifically designed for the task of depth completion it is only usable for non-learning approaches due to its small size. Besides, the KITTI stereo benchmark [30, 78] provides 400 high resolution depth ground truth images of street scenes. The raw data section of KITTI contains some of those scenes which makes it possible to gain the raw LIDAR scans corresponding to the ground truth depth maps in order to use them for depth completion. However, none of the above-mentioned datasets is large enough for end-to-end training of neural networks. Furthermore, there exists no public benchmark for the task of depth completion.

Therefore, there is a need of a dedicated dataset for depth completion which we will discuss in the following sections. The remainder of this section is structured as follows. First, the KITTI depth dataset is introduced in Section 3.5.1. Next, details of the automatic ground truth depth map generation are described in Section 3.5.2. In Section 3.5.3 the quality of those depth maps is assessed by comparing it to manually cleaned ground truth depth maps. Finally, the KITTI depth benchmark is presented in Section 3.5.4.

3.5.1 KITTI Depth Dataset

In order to address the shortcomings of the previously mentioned datasets, the KITTI Depth Dataset is introduced in this section. It is based on the KITTI raw dataset [29] which comprises over 94k frames of urban and rural street scenes. Each frame contains rectified RGB stereo camera images with a resolution of maximum 1382x512 pixels, depending on the crop for the rectification. Additionally, raw Velodyne scans and odometry data are provided. For the depth dataset, stereo and LIDAR information is fused to obtain semi-dense depth ground truth (details on the depth map generation are summarized in the next section). Next to the semi-dense ground truth maps, also the raw LIDAR point cloud of each frame is provided as a projection on the high resolution RGB image.

The dataset is publicly available and can be downloaded from the KITTI homepage[1].

For training and validation the depth dataset is split into 93k and 1k frames, respectively. Furthermore, additional 1000 frames with images and raw LIDAR scans are provided for a public depth completion benchmark. As depth data can also be leveraged for single image depth estimation, a second benchmark was established where 500 RGB images without corresponding sparse depth measurements are provided. For both benchmarks the ground truth is not revealed to ensure a fair comparison of the approaches. Further information on the benchmark can be found in Section 3.5.4.

Data format The depth maps (ground truth as well as raw Velodyne scans) are saved as uint16 PNG images. Invalid pixels (i.e. no ground truth exists, or the estimation algorithm didn't produce an estimate for that pixel) have a value of zero. All other pixels contain a scaled depth value. In order to convert the scaled value back to the actual distance, the pixel value has to be divided by 256. This results in the following reconstruction of the depth map $D(u,v)$ and valid pixel mask $M(u,v)$ at pixel position (u,v):

$$D(u,v) = I(u,v)/256.0 M(u,v) = I(u,v) > 0 \tag{3.23}$$

Note that this limits the accuracy to $1\,\mathrm{m}/256.0 \approx 3\,\mathrm{mm}$ and the maximal value for the range to $65535/256 \approx 256$. However, the Velodyne HDL-64E data sheet reports a 2 cm accuracy, so the discretization error is neglectable. Also, the maximal range was never exceeded in the existing data.

Data characteristics As the depth dataset originates from the KITTI raw data, the frames are sampled at 10 Hz. This sampling is quite dense, especially if the sensor vehicle is driving at a slow speed or is even stopping. However, as it was possible to provide a dense depth map for each frame without extra effort, the depth dataset contains all those frames even if the information content is redundant. To improve the training and add a larger variety to the data, the included GNSS data could be leveraged to filter similar frames.

[1] `http://www.cvlibs.net/datasets/kitti/eval_depth_all.php`

3.5.2 Dataset Generation

The KITTI raw dataset provides depth information in the form of raw Velodyne scans, however it needs some effort to generate accurate semi-dense depth maps from those. In order to increase the density of the depth maps, Geiger et al. [30] proposed to accumulate 11 laser scans. This includes the current considered frame as well as 5 past and 5 future frames. However, there are several artifacts like sensor noise, artifacts due to occlusions (e.g., due to the different mounting positions of laser scanner and camera), reflecting/transparent surfaces or time synchronization issue due to moving obstacles or the moving sensor platform. While Geiger et al. [30] cleaned this data manually, this is not an option for all the 94k frames in the dataset.

Therefore, an automatic processing of the data is proposed in the following. Similar to Geiger et al. [30], first 11 laser scans are accumulated to obtain a semi-dense depth map. Second, outliers in the resulting depth map are compared to results from a stereo reconstruction approach. In this case Semi-Global Matching (SGM) by Hirschmüller et al. [43] was used.

Both depth measurement methods however have their unique error pattern. While stereo reconstruction can lead to bleeding artifacts at depth edges, LIDAR sensors show streaking artifacts along their motion direction. To mitigate both effects, consistency between laser scans and stereo reconstruction is enforced by removing all LIDAR points exhibiting large relative errors in comparison to the stereo reconstruction. Note that this reduces the density of the depth map but increases the accuracy. Another option would have been to try to reconstruct the depth as accurate as possible by using a global optimization. However, using only the transformed raw laser measurements has the advantage to introduce no bias. This is especially important if methods are to be compared in a benchmark.

While the laser scans are accumulated over time, only the current SGM depth map is used to clean the depth measurements. This is important as moving objects can result in streaks in the accumulated laser scan projection. With this all the typical outliers can be removed in one step: Occlusions, dynamic motion and measurement artifacts. Furthermore, most errors due to reflecting and transparent surfaces can also be removed with this simple technique as SGM and LIDAR rarely agree in those regions (although both tend to have huge errors in those regions).

3.5.3 Dataset Evaluation

While having an enormous amount of data for training is important to train complex networks, the quality of the provided ground truth is at least equally important. Therefore, in this section the depth maps of the dataset are evaluated by comparing them to the manually cleaned training set of the KITTI 2015 stereo benchmark [78] as reference data. Note that the KITTI 2015 stereo benchmark data uses accumulated LIDAR scans as well as 3D CAD models. In order to obtain a dense depth map of all static elements, 7 LIDAR scans are accumulated and the rolling shutter effect as well as the camera motion are corrected by combining IMU data with ICP fitting of the point clouds. As the benchmark uses the same laser scanner which was used to generate the automatically generated dense depth maps described in Section 3.5.2 the evaluation is biased. However, the KITTI 2015 benchmark data experienced massive manual inspection followed by a cleaning and correcting step. Furthermore, the 3D CAD models yield a dense reconstruction of all dynamic objects. Therefore, while still biased, the evaluation gives valuable insights into the quality of the data, especially for areas where automatic accumulation failed for the KITTI 2015 benchmark.

In the following, various error metrics are computed. As the KITTI 2015 stereo benchmark provides disparity maps, the metrics are reported in the disparity space. The disparity d is only evaluated at N valid pixel locations $\mathscr{V}$. The following metrics are used throughout this section:

- **Mean Absolute Error (MAE):**
  $MAE = \frac{1}{N_v} \sum_{d_{u,v} \in \mathscr{V}} \left| d_{u,v} - \hat{d}_{u,v} \right|$
- **Root Mean Squared Error (RMSE):**
  $RMSE = \sqrt{\frac{1}{N_v} \sum_{d_{u,v} \in \mathscr{V}} (d_{u,v} - \hat{d}_{u,v})^2}$
- **KITTI outliers:**

$$o_{kitti} = \frac{1}{N_v} \sum_{d_{u,v} \in \mathscr{V}} o_{u,v},$$

$$o_{u,v} = \begin{cases} 1, & \text{if } |d_{u,v} - \hat{d}_{u,v}| \geq 3px \text{ and } \frac{|d_{u,v} - \hat{d}_{u,v}|}{\hat{d}_{u,v}} \geq 0.05 \\ 0, & \text{otherwise} \end{cases}$$

	Density [%]	MAE [px]	RMSE [px]	KITTI outliers	δ_i inlier rates δ_1	δ_2	δ_3
COLMAP	86.9	1.64	4.86	8.91	93.98	95.89	97.36
SGM	82.4	1.07	2.80	4.52	97.00	98.67	99.19
Raw LIDAR	4.0	**0.35**	2.62	1.62	98.64	99.00	99.27
Acc. LIDAR	30.2	1.66	5.80	9.07	93.16	95.88	97.41
KITTI Depth	16.1	**0.35**	**0.84**	**0.31**	**99.79**	**99.92**	**99.95**

Table 3.1: Evaluation of reference depth maps using the manually curated ground truth depth maps of the KITTI 2015 training set [78]. Note that the KITTI depth dataset is generated fully automatically and achieves the highest accuracy while providing high density. All metrics are computed in disparity space.

- **Inlier ratios:**

$$\delta_j = \frac{1}{N_v} \sum_{d_{u,v} \in \mathcal{V}} f_i(d_{u,v}),$$

$$f_i(d_{u,v}) = \begin{cases} 1, & \text{if } \max(\frac{d_{u,v}}{\hat{d}_{u,v}}, \frac{\hat{d}_{u,v}}{d_{u,v}}) \leq 1.25^j \\ 0, & \text{otherwise} \end{cases}$$

Figure 3.9 sums up the qualitative comparison of the depth maps while a quantitative comparison is shown in Table 3.1.

The SGM stereo camera reconstruction is dense but lacks accuracy compared to the raw laser scans. The same applies to COLMAP, which is a framework that provides Structure-from-Motion (SfM) and Multi-View Stereo (MVS) approaches to reconstruct a 3d scene given image sequences. For this comparison several hyperparameters where swiped to achieve the best 3d reconstruction (to the best knowledge of the author), however, the resulting depth maps still yield larger errors especially for scenes with dynamic objects. The depth maps of the automatically generated dataset have approximately the same MAE as the raw LIDAR scans. However, regarding RMSE, KITTI outlier and inlier ratios the auto-generated dataset outperforms the other baseline methods. While the dataset depth maps are not as dense as the stereo and SfM based methods, it is 4x denser than the raw LIDAR sweeps and therefore a good trade-off between accuracy and density.

As dynamic objects are very challenging for approaches which use accumulated LIDAR scans, the quality of those objects is of particular interest.

Figure 3.9: Qualitative comparison of the large-scale depth dataset. From left to right: depth maps of the manually curated KITTI 2015 dataset, the proposed automatically generated data, raw and accumulated LIDAR scans, and SGM [43] results. Differences to the KITTI 2015 depth maps, serving as ground truth, are shown in the last row from 0 (green) to 2 (red) meters. Adapted from [111].

Depth Map	MAE [px]	RMSE [px]	KITTI outliers	δ_i inlier rates δ_1	δ_2	δ_3
SGM	1.2	3.0	5.9	97.6	98.2	98.5
Raw LIDAR	3.7	10.0	17.4	84.3	86.1	88.6
Acc. LIDAR	7.7	12.0	59.7	55.7	73.7	83.0
KITTI Depth	**0.9**	**2.2**	**3.0**	**98.6**	**99.0**	**99.3**

Table 3.2: Evaluation of Table 3.1 split according to foreground (car) regions.

Depth Map	MAE [px]	RMSE [px]	KITTI outliers	δ_i inlier rates δ_1	δ_2	δ_3
SGM	1.1	2.8	4.4	96.9	98.7	99.3
Raw LIDAR	**0.2**	1.9	0.9	99.3	99.6	99.7
Acc. LIDAR	1.1	4.8	4.3	96.7	98.0	98.8
KITTI Depth	0.3	**0.8**	**0.2**	**99.8**	**99.9**	**99.9**

Table 3.3: Evaluation of Table 3.1 split according to background (non-car) regions.

Therefore, the benchmark depth maps are manually separated in fore- and background. Subsequently, the errors present on dynamic objects (Table 3.2) and background (Table 3.3) are evaluated.

The results show that the method proposed in Section 3.5.2 is able to remove outliers in the accumulated LIDAR scans while significantly increasing the density of the data. Qualitatively, there are only little errors in the proposed dataset. In comparison, SGM stereo yields significantly worse results at large distances due to the measurement principle. As for raw LIDAR scans, parallax of laser scanner and camera result in occlusion errors (the image plane of the KITTI reference camera is used as reference for all the experiments). While there still remain some errors on dynamic objects, c.f. Figure 3.9 (bottom), the comparison to SGM greatly reduces the errors in the accumulated LIDAR scans (e.g., car on the left).

3.5.4 KITTI Depth Benchmark

In this section a depth dataset was presented which is not only large but also sufficiently accurate to train and evaluate machine learning and other methods. For scientific use, the data has been uploaded and is provided via the KITTI benchmark page[2].

[2] (`http://www.cvlibs.net/datasets/kitti/eval_depth_all.php`)

In recent years there has been an increased number of publications of depth completion methods, both guided by RGB images and non-guided. However, at the time the dataset was published there was no common benchmark to evaluate those methods. Therefore, the KITTI depth completion benchmark was created by adding disjunct test data, containing raw depth maps and RGB images[3]. The user can upload the results in a pre-defined format which is described in the development kit attached to the website. The results are automatically evaluated against a ground truth which is not published. For the ranking four different metrics are used:

Mean Absolute Error:

$$\varepsilon_{MAE} = \frac{1}{N} \sum_{u,v} \left| (\hat{D}_{u,v} - D_{u,v}) \right| \tag{3.24}$$

Inverse Mean Absolute Error:

$$\varepsilon_{iMAE} = \frac{1}{N} \sum_{u,v} \left| (\frac{1}{\hat{D}_{u,v}} - \frac{1}{D_{u,v}}) \right| \tag{3.25}$$

Root Mean Squared Error:

$$\varepsilon_{RMSE} = \sqrt{\frac{1}{N} \sum_{u,v} (\hat{D}_{u,v} - D_{u,v})^2} \tag{3.26}$$

Inverse Root Mean Squared Error:

$$\varepsilon_{iRMSE} = \sqrt{\frac{1}{N} \sum_{u,v} (\frac{1}{\hat{D}_{u,v}} - \frac{1}{D_{u,v}})^2} \tag{3.27}$$

The MAE is less sensitive to large deviations. However, in automotive applications large errors might be fatal. Therefore, the Root Mean Squared Error (RMSE) has been chosen as the leading metric as it punishes large deviations more than the MAE. Additionally, the Inverse Mean Absolute Error (iMAE) and Inverse Root Mean Squared Error (iRMSE) are reported which are less sensitive to errors in far distances. This makes sense as the LIDAR point cloud is less dense on far away objects.

[3] `http://www.cvlibs.net/datasets/kitti/eval_depth.php?benchmark=depth_completion`

Another important field of work where depth images are needed is the task of single image depth prediction. Given a monocular RGB image, the depth is estimated for each pixel. The proposed dataset can be used to train methods for this task, too. Therefore, an additional single image depth prediction benchmark was published in parallel[4]. This benchmark was also part of the Robust Vision Challenge 2018 and 2020[5].

3.6 Experiments

In this section the two previously presented depth completion approaches are evaluated first separately and then jointly on the large depth completion dataset presented in the previous section. First, the optimization-based semantically guided depth completion approach from Section 3.3 is evaluated in Section 3.6.1. The section focuses on benchmarks without semantics as well as an evaluation of a subset of the KITTI dataset. Second the learning-based depth completion approach from Section 3.4 is presented in Section 3.6.2. This section focuses on the evaluation of the sparsity invariant convolution and shows several experiments on synthetic data. Finally, in Section 3.6.3 both methods are evaluated on the large scale depth dataset for better comparison.

3.6.1 Semantically Guided Depth Completion

The semantically guided optimization-based depth completion approach is evaluated on three different depth inputs. For better readability, from now the approach is named SGDC (Semantically Guided Depth Completion). First, SGDC is assessed on subsampled depth data of the very precise ground truth of the Middlebury 2014 stereo dataset [98]. As the quality of the depth data does not reflect low-cost sensor data as mostly used in robotics and especially in the field of autonomous driving, two further datasets are used for evaluation: the ToF indoor dataset introduced by Ferstl et al. [26] and the KITTI dataset ([30], [29] and [78]) which provides LIDAR scanner measurements. It is important to mention that the first two datasets do not offer a training dataset for semantic segmentation. Without semantic segmentation

[4] `http://www.cvlibs.net/datasets/kitti/eval_depth.php?benchmark=depth_prediction`

[5] `http://www.robustvision.net`

as an extra cue, a reliable edge detector is important to guide the depth completion. Thus, for all conducted experiments, the edge detector of Dollar et al. [19] is used. It leverages trained structured forests in order to predict semantic edge scores for each pixel. Therefore, in contrast to traditional color-gradient-based edge detectors such as the Canny Edge Detector ([7]), the edges resemble more generic object boundaries.

Middlebury Benchmark Evaluation In order to test the performance of depth completion algorithms on clean, almost noise-free, input data, the Middlebury 2014 Stereo Benchmark [98] is a popular choice. The dataset comprises high resolution images and corresponding ground truth depth maps which are generated by structured lighting. In order to simulate a highly accurate but sparse depth sensor, the depth data is sampled at equidistant points, resulting in a downsampled version of the high resolution depth map. This is in contrast to some previous works [26, 84] which use Bicubic or Bilinear downsampling to simulate the input. However, this kind of interpolation methods distort the input data especially at depth edges and blur tiny structures which are of particular interest.

To evaluate SGDC in terms of speed and accuracy it is benchmarked against four other approaches. The first three are simple interpolation methods: nearest neighbor, Bicubic and Bilinear interpolation. The forth is a state-of-the-art approach by Ferstl et al. [26]. The parameters of SGDC and Ferstl's algorithm [26] were empirically determined for each downsampling factor. Table 3.4 shows quantitative results in terms of the Mean Absolute Error (MAE). SGDC significantly outperforms all baselines on all three images and all downsampling factors. In particular, it shows superior performance on very sparse inputs.

The runtime of SGDC is visualized in Figure 3.10. It strongly depends on the number of input depth points, whereas the target image resolution is only critical for calculating the edge and semantic features (which is rather fast in comparison). In contrast, the approach of Ferstl et al. [26] is mostly bound by image resolution (they report moderate runtimes of 318.2 ms for completing 0.8 megapixel images and 1900 ms for 1.4 megapixel images). Both behaviors have advantages on different problems. In the field of autonomous driving, real world depth sensors typically have a very limited resolution, and thus it is preferable to increase performance and speed for

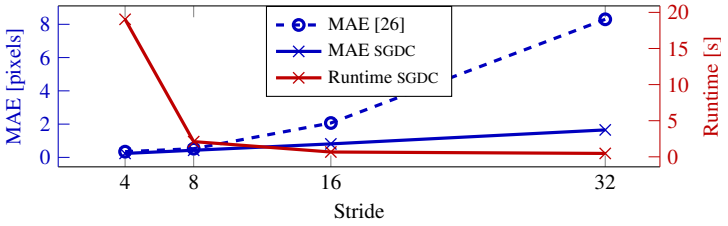


Figure 3.10: Comparison of runtime and MAE of SGDC. Comparison of runtime and MAE in disparity [px] for different strides on the Middlebury 2014 Benchmark [98]. Adapted from [101].

	Straw Hat				Bicycle			
	4x	8x	16x	32x	4x	8x	16x	32x
Bicubic	1.72	2.94	4.65	7.15	1.08	1.79	2.90	4.29
Bilinear	1.44	2.56	4.23	6.65	0.93	1.53	2.52	3.80
Nearest	1.28	2.40	4.11	6.75	0.79	1.44	2.39	3.75
Ferstl [26]	0.49	0.75	2.61	9.42	0.24	0.33	0.92	3.13
SGDC	**0.43**	**0.70**	**1.07**	**1.83**	**0.17**	**0.29**	**0.45**	**0.71**

	Photo Set				Backpack			
	4x	8x	16x	32x	4x	8x	16x	32x
Bicubic	0.77	1.26	2.18	4.06	2.22	3.44	4.76	6.07
Bilinear	0.67	1.10	1.90	3.64	1.95	3.21	4.56	6.20
Nearest	0.56	0.94	1.69	3.37	2.65	3.95	5.24	6.74
Ferstl [26]	0.15	0.22	1.37	9.10	0.50	0.79	3.37	11.54
SGDC	**0.14**	**0.21**	**0.28**	**0.49**	**0.15**	**0.27**	**0.47**	**0.90**

Table 3.4: Results on the Middlebury [98] dataset. The table displays the MAE in disparity [px] for the Middlebury 2014 dataset for four chosen images.

completing a small amount of input points to a very high-resolution depth map.

Figure 3.10 visualizes the performance and runtime of SGDC in comparison to the number of input measurements.

Results for ToF depth completion While the Middlebury dataset might be a good choice to evaluate the general ability of sensors to solve the problem of depth completion it does not reflect real acquisition setups which are typically significantly noisier. Therefore, Ferstl et al. [26] published a small

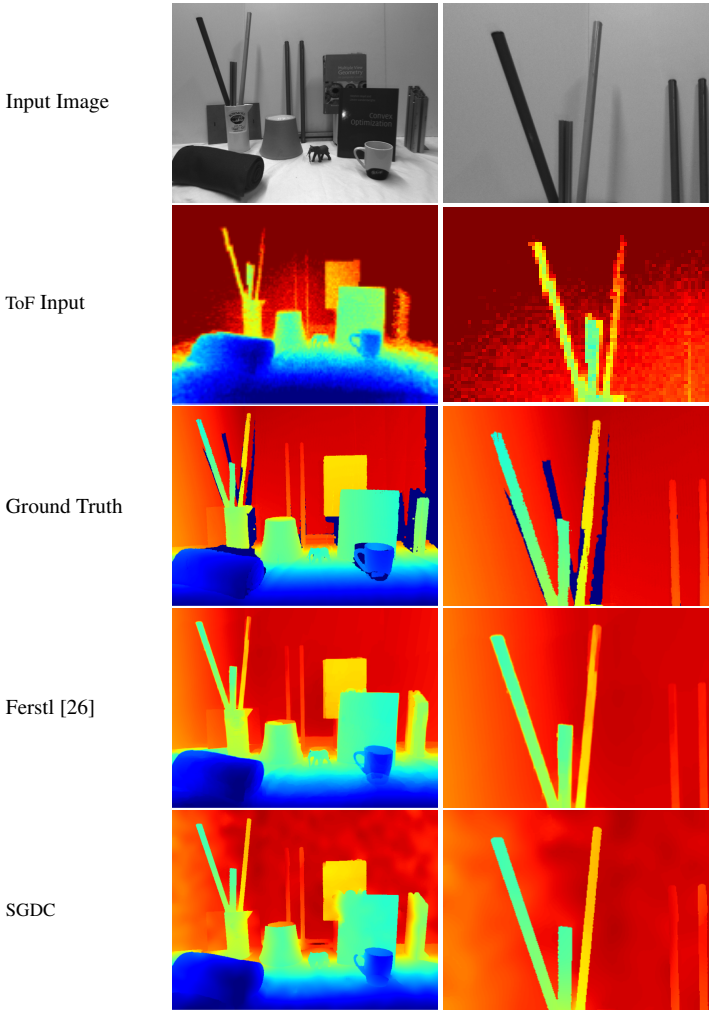


Figure 3.11: Depth completion of Time-of-Flight data. Result for completing the ToF data on the 'books' scene of the Ferstl dataset [26]. In the second row a zoomed-in view is shown for better visual comparison. Although input data is noisy, SGDC preserves sharp edges. Adapted from [101].

	Books	Shark	Devil
Kopf [53]	16.03	18.79	27.57
He [39]	15.74	18.21	27.04
Ferstl [26]	12.36	**15.29**	14.68
SGDC	**12.12**	15.46	**14.03**

Table 3.5: Results on the Ferstl Dataset [26]. The reported results are given by the MAE in depth [mm].

dataset with the paper. It consists of three different scenes captured by a 120×160 pixel wide ToF camera as well as a 810×610 pixel wide grayscale CMOS camera. The ground truth depth maps for those scenes were generated using a structured light scanner with a high-speed projector as well as two 2048×2048 high-speed cameras. The resulting depth maps have an uncertainty of 1.2 mm.

In the work of Ferstl et al. [26], the results of two other depth completion algorithms are reported on the Ferstl dataset: Joint Bilateral Upsampling by Kopf et al. [53] and Guided Image Filtering by He et al. [39]. Table 3.5 shows a quantitative comparison of the different approaches. The approaches of He et al. [39] and Kopf et al. [53] are sensitive to noise and perform worse in comparison to the other two approaches. Both the method of Ferstl et al. Figure 3.11 and SGDC handle the noisy ToF depth measurements well. A qualitative comparison is depicted in Figure 3.11. SGDC produces smooth surfaces while preserving sharp depth edges and thin elements. The average runtime of SGDC is 107ms on an Intel Xeon E5 for a single frame. In comparison, the method of Ferstl et al. [26] needs on average 318.2 ms.

In summary, this series of experiments show that SGDC is not only able to complete noise-free depth measurements but is also competitive in completing noisy depth measurements of low-budget ToF cameras. However, the resolution of ToF cameras is rather high in comparison to typical LIDAR sensors which are used today in the context of assisted and automated driving. Thus, in the next paragraph further experiments are conducted, considering a vehicle with a LIDAR-camera setup.

Sparse Automotive LIDAR Data Completion As already discussed in Section 3.5, Menze et al. [78] proposed a dataset for stereo evaluation which provides dense data not only for static but also dynamic objects. Al-

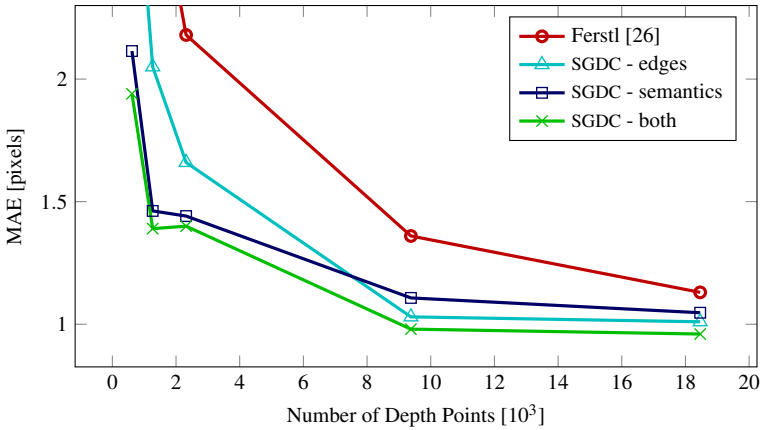


Figure 3.12: MAE over number of points on KITTI 2015 [78]. Quantitative results on the KITTI 2015 dataset for varying number of input measurements. Results in terms of disparity MAE depending on the number of input points are reported for a baseline [26] and three input configurations (edge guidance only, semantic guidance only, both inputs as guidance) of the proposed method. Adapted from [101].

though the dataset size is a lot smaller than the previously proposed large-scale KITTI depth dataset, the depth maps have a high quality due to manual filtering. Furthermore, the fitted CAD model projection for the cars fit tightly, providing sharp depth edges at vehicle boundaries. Therefore, experiments on this dataset are valuable, especially to analyze accuracies on object borders. For the depth completion experiments in this paragraph, raw LIDAR measurements are used as an input to the algorithms. In contrast to the ground truth the raw scans differ significantly in terms of outlier frequency and density. After projecting the raw LIDAR points onto the image, there are still many errors due to the sensor displacement, especially at top borders of objects. SGDC can handle those outliers by rejecting them in the optimization instead of heuristically removing them, c.f. Section 3.3. Additionally, pixel-level semantic information helps to cluster corresponding depth points and therefore allows rejecting measurements with similar semantics but largely different depth values. For the semantic input to SGDC a Fully Convolutional Network [72] is pre-trained on the Cityscapes Dataset [16] and then fine-tuned on a disjunct part of the KITTI dataset. The images were annotated with pixel-wise labels by Ladicky et al. [62]. The seman-

tic labels are split into 11 classes including road, sidewalk, car, pole and building.

As an evaluation metric the mean absolute error is calculated over all pixels which have a valid ground truth disparity. In order to evaluate the performance of the proposed method and the baseline on different input densities, the raw LIDAR point clouds are further downsampled vertically and horizontally. This resembles LIDAR scanners with fewer scan lines and a higher rotation frequency. It is apparent from Figure 3.12 that SGDC performs well for all densities and outperforms the method of Ferstl et al. [26] especially if the measurements are getting sparser. In fact, the method of Ferstl et al. [26] completely fails to complete scenes with very sparse measurements as depicted in Figure 3.13, although the parameters of the algorithm were optimized using all frames of the validation set. Note that the sampling strategy is different to the one in Section 3.6.3. As the projected points are much further away in the image space due to the vertical sampling, this is especially hard for the method of Ferstl et al. [26]. While the improvement of SGDC due to the semantic cue is only minor for higher densities, it plays a greater role for very sparse measurements.

3.6.2 Sparsity Invariant CNNs

Depth Map Completion In this section the *Sparse Convolution Module* of section is evaluated. While the module can be used for several tasks with sparse inputs, this section focuses mainly on the task of depth completion. In order to analyze the impact of sparse input to a convolutional neural network in a controlled setting, depth data from a synthetic dataset is used without any RGB guidance. In particular, high resolution depth maps from the Synthia dataset of Ros et al. [95] are used, and depth values are randomly dropped in order to obtain sparse point clouds. This gives full control over the level of sparsity presented to the network. For the experiment series different sparsity levels from 0% to 95% are applied indicating the probability of a pixel to be randomly dropped.

In order to compare regular convolutions to the sparse convolution modules, three different variants of a Fully Convolutional Network (FCN) are trained. The networks contain 5 layers with a kernel size of 11, 7, 5, 3, and 3, each with a stride of one and 16 output channels. After each convolution a rectifying linear unit is applied as nonlinear activation function. As a solver, Adam [58] is used with momentum terms of $\beta_1 = 0.9$ and $\beta_2 = 0.999$

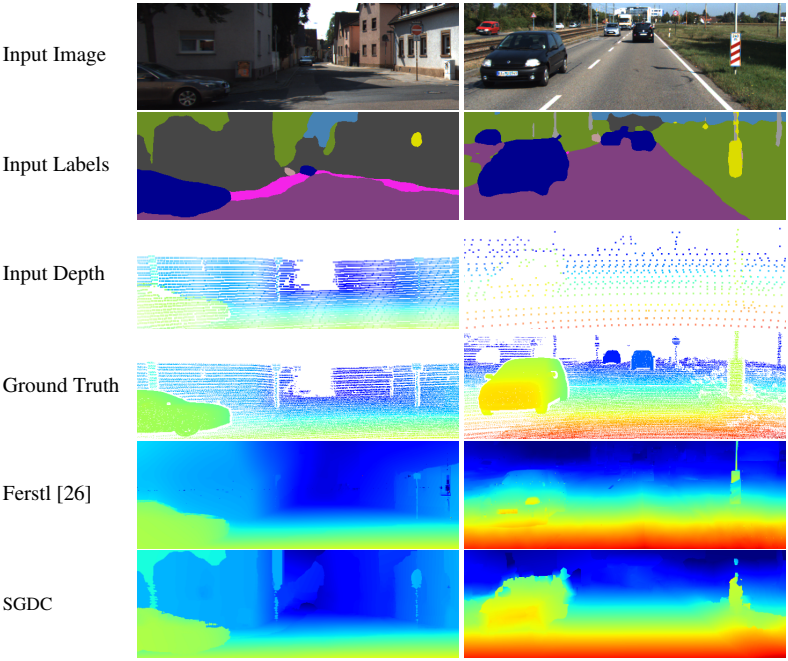


Figure 3.13: Sparsity analysis on the KITTI 2015 data [78]. Example outputs of the proposed algorithm compared to the result of Ferstl [26], ground truth and the inputs for two cases with all (left) and $\frac{1}{12}$ (right) measurements. The semantic input guides SGDC to respect depth jumps as can be seen at the objects, e.g. car, sign and poles, in the lower example. The baseline discards the few measurements on the car, while SGDC recognizes it correctly. The 3D reconstruction using the inferred depth is presented in Figure 3.2. Note that the sign on the right (top) is correctly represented without available depth measurements, by propagating the depth along the sign and pole. Adapted from [101].

together with a fixed learning rate of $\alpha = 1 \cdot 10^{-3}$ and weight decay of $5 \cdot 10^{-4}$. As baselines two standard approaches are considered. First, plain convolutions with only sparse depth as input. Second, additionally to the plain convolutions, a valid pixel map is concatenated to the input. Those two approaches are then compared to a network which uses sparse convolution modules as depicted in Figure 3.7. From now on this particular network is called Sparsity Invariant CNN, or short SICNN.

For each level of sparsity a new training dataset is used and with that only a single level of sparsity is presented to the networks during training. The training data consists of Synthia *Summer* sequences, whereas evaluation is performed on the Synthia *Cityscapes* dataset. First, all models are evaluated on the sparsity level they have been trained on. This gives an idea of the ability of the network to be able to fill gaps and deal with a specific amount of sparsity. Due to the random dropping of depth samples and the limited capacity of the network, this is already a challenging task.

Next, to test the generalization ability of the different models, other sparsity levels, which the models have not seen during training, are used for evaluation.

In Figure 3.14 the result of various networks being trained and evaluated on 5% of the input is shown. We can observe that given a very sparse input plain convolutions perform poorly. As all pixels are considered equally during training and evaluation, a large degree of randomness is induced resulting in strong variations in performance. Additionally, concatenating a valid pixel mask performs slightly better than using only the depth input. However, the results are still not convincing, especially if very sparse input is presented to the network. In contrast, SICNN performs consistently well over all sparsity levels. This is an important property for depth completion methods as it increases the robustness towards random perturbations of the input data. Furthermore, this allows to generalize to different sensor setups with depth sensors of different density. Interestingly, even if trained as an autoencoder (100% of the pixels are considered in training and evaluation), the averaging module helps to guide the training process resulting in a slightly better performance than the standard convolutions. In this case, the concatenated mask does deteriorate the result as it further complicates the training.

A qualitative comparison of the generated dense depth maps is shown in Figure 3.14. All three approaches were trained with depth maps where only 5% of the pixels contained a valid depth. To improve readability, the input in Figure 3.14 has been visually enhanced using dilation, thus appearing more dense than the actual input to the networks. As a further comparison, the exact same models (trained on 5% sparsity) are used to complete depth maps with 20% sparsity. As can be seen in Figure 3.16 the performance drops drastically for both baselines. While CNNs with input masks lead only to noisy results, the standard CNNs have a systematic bias as they are unaware of the level of sparsity in the input. In contrast, the sparse convolution mod-

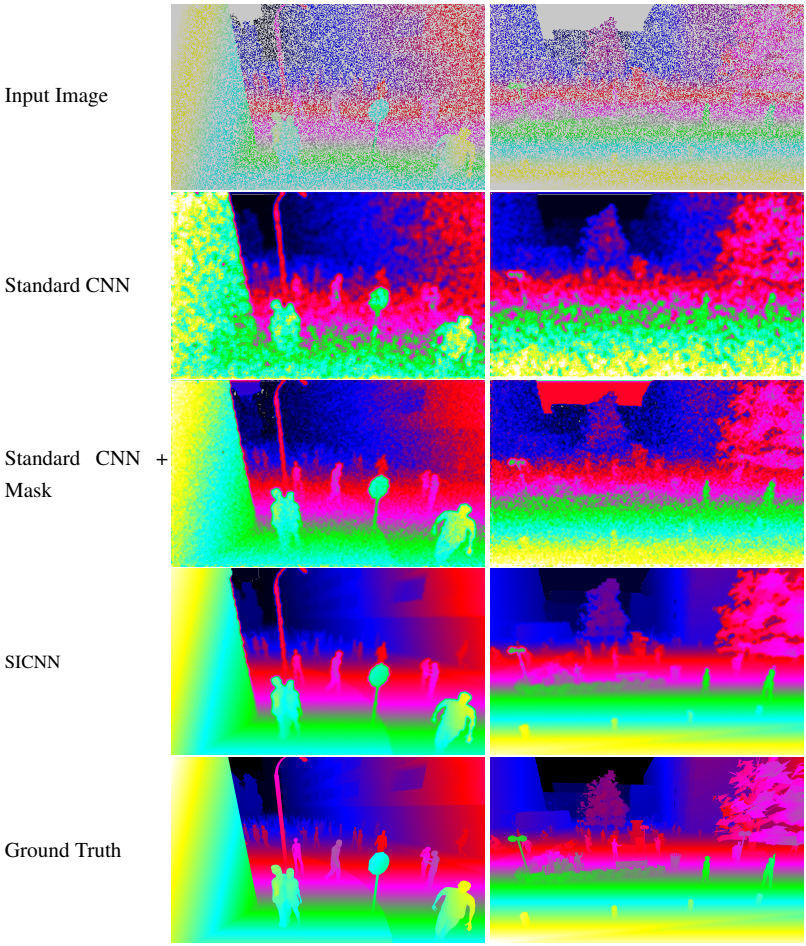


Figure 3.14: Comparison of sparse and standard convolution Qualitative comparison of the sparse convolutional network to a standard CNNs on Synthia [95], trained and evaluated at 5% sparsity. The Standard CNNs suffer from large invalid regions in the input leading to noisy results. Using a valid mask as input reduces noise slightly. In contrast, SICNN predicts smooth and accurate outputs. Adapted from [111].

	5%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
5%	3.6	6.1	9.3	10.7	11.5	12.1	12.4	12.7	13.0	13.2	13.4
10%	7.1	2.4	5.4	7.2	8.3	8.9	9.3	9.5	9.5	9.4	9.1
20%	18.0	7.4	2.6	3.8	4.9	5.8	6.2	6.5	6.6	6.4	6.2
30%	23.6	11.9	4.1	1.9	2.6	3.4	3.9	4.0	4.1	3.7	3.7
40%	30.1	16.3	7.1	3.4	2.2	2.7	3.7	4.3	4.5	4.7	5.1
50%	46.5	24.9	26.7	5.8	2.8	1.8	4.3	3.2	3.8	4.3	4.9
60%	41.6	24.3	12.2	6.9	4.1	2.4	1.8	2.3	2.9	3.4	4.1
70%	64.9	38.4	19.5	11.8	7.1	4.2	2.5	1.7	2.2	2.9	3.8
80%	61.2	38.3	20.9	13.1	8.7	5.7	3.6	2.2	1.5	2.1	2.8
90%	159.6	84.4	41.1	25.2	16.5	11.0	7.2	4.4	2.4	1.4	2.4
100%	1288.5	886.9	506.6	318.9	202.9	128.0	76.6	37.7	14.8	6.1	1.2

(a) Standard Convnet

	5%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
5%	10.0	5.2	6.7	7.3	7.4	8.0	8.6	10.2	12.4	11.5	12.6
10%	16.6	13.3	5.7	8.7	10.3	11.2	13.0	12.0	12.5	12.5	14.2
20%	7.7	16.6	8.4	4.5	5.1	7.0	7.5	8.5	9.6	10.4	11.1
30%	8.5	6.4	5.8	6.3	4.3	7.1	14.2	33.4	46.4	35.1	23.6
40%	9.9	7.2	5.3	4.7	4.6	3.7	3.8	4.6	5.2	6.3	8.0
50%	12.1	8.3	5.5	3.9	4.8	5.1	4.5	5.1	6.6	17.4	166.5
60%	10.4	8.9	7.1	5.7	4.4	3.6	3.6	3.4	3.3	5.1	6.6
70%	14.7	9.9	7.8	6.9	5.8	4.4	3.3	3.5	3.9	6.1	8.0
80%	15.8	10.6	8.8	7.9	7.1	6.0	4.7	4.1	3.4	3.2	4.9
90%	13.4	10.0	7.9	6.7	5.7	5.3	4.7	3.7	3.0	6.4	17.4
100%	16.6	16.0	17.4	18.4	18.6	17.0	14.2	12.5	10.0	6.6	2.9

(b) Standard Convnet + Mask

	5%	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
5%	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99	0.99
10%	0.98	0.98	0.97	0.98	0.97	0.97	0.97	0.97	0.97	0.97	0.97
20%	0.98	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
30%	0.98	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.98
40%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
50%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
60%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
70%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
80%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
90%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98
100%	0.99	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98	0.98

(c) SICNN

Figure 3.15: Quantitative analysis of sparsity invariance. Comparison of three different networks on the Synthia dataset [95] while varying the sparsity level of the training split (left) *and* the sparsity of the test split (top). From top-to-bottom: Standard CNN, CNN with concatenated validity mask and the proposed Sparsity Invariant CNN (SICNN). All numbers represent mean average errors (MAE). Adapted from [111].

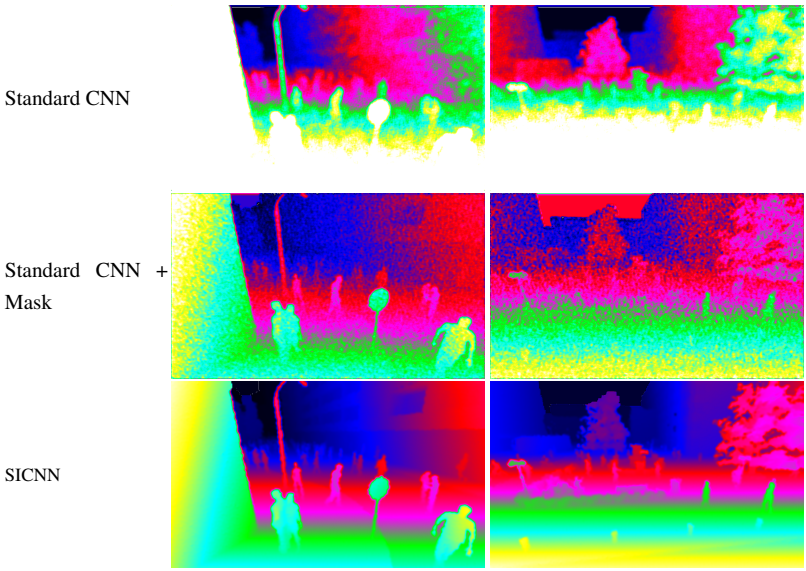


Figure 3.16: Qualitative analysis of the sparsity invariance. Visualized are network predictions for the scene of Figure 3.14, but with all networks trained at 5% sparsity and evaluated at 20% sparsity. While standard CNNs with and without visibility mask produce substantially worse results, the results of the proposed Sparsity Invariant CNN do not degrade. Adapted from [111].

ules are able to recover the dense depth map without noise or biased depth estimations.

Semantic Labeling from Sparse Depth To analyze the general applicability of SICNN to handle sparse input, the ability of the network to predict dense semantic segmentation masks from sparse depth input is investigated. To this end a VGG16 architecture [72] is modified by replacing regular convolutions by sparse convolution modules. Additionally, weighted skip connections, which were presented in Section 3.4.3, are used to generate high-resolution predictions from the small, spatially downsampled FC7 layer and at the same time incorporating visibility masks of the respective network stages.

Table Table 3.6 depicts the mean intersection over union after training on all Synthia “Sequence” frames (left camera to all directions, summer

Network	IoU[%]
VGG - Depth Only	6.4
VGG - Depth + Mask	4.9
VGG - Sparse Convolutions	31.1

Table 3.6: Semantic segmentation evaluation on Synthia Intersection-over-Union (IoU) of different network variants on the Synthia Cityscapes subset after training on all Synthia sequences (mean over all 15 known classes).

only) followed by an evaluation on the Synthia "Cityscapes" subset. All unknown classes in the validation set are mapped to corresponding classes in the training set while all other unknown classes are ignored. While there is still room for improvement for all the results, it is clearly visible that the sparse convolution module outperforms the two baselines by a large margin. The bad results can be explained by the different nature of the validation set which contains more people and also very different viewpoints (bird's eye vs. street-level).

The preceding experiments show that sparse inputs are indeed a challenge for conventional CNNs and that sparse convolutions help in the context of unstructured randomly sparse depth input. However, dropping depth samples from a high resolution depth map does not resemble actual depth sensors and their behavior in the wild and in particular in the setting of autonomous driving. Therefore, in the next section the sparse convolution modules are compared against further baselines on the presented real world depth completion dataset from Section 3.5.

3.6.3 Results on large-scale dataset

In the previous sections, two different depth completion approaches were evaluated independently, each on their own domain. The optimization-based method showed strengths in recovering small details, also if no training data is available for the presented scenes. The Sparsity Invariant CNNs showed superior performance on varying levels of sparsity of synthetic depth maps, given that enough training data is available. The open question remains how both approaches perform on real world settings, especially for sensor setups which are common for autonomous driving. Therefore, in this section both approaches are evaluated on the previously presented large depth dataset.

Note that there are major differences between the two presented approaches. While the optimization-based algorithm needs training data for training the semantic segmentation to guide the depth completion, SICNN require dense depth maps, but no semantic annotations. Furthermore, SICNN in contrast to SGDC does not use RGB images for the completion task. Using RGB images as guidance can have advantages and disadvantages. While leveraging dense color information can be very useful to control the interpolation of sparse depth points, relying on camera information in multi-modal sensor setups in such an early processing stage is not always recommended. Especially if bad weather and night scenes can diminish the benefit of image data or even worsen the result.

Table 3.7 (top) shows a comparison of several RGB guided depth completion approaches, including SGDC, the method by Barron et al. [2], Ferstl et al. [26], and Jampani et al. [51]. For a fair comparison the same amount of convolutional layers are added for both, SICNN and the approach of Jampani et al. [51]. For all other baseline methods a hyperparameter search was performed via grid search on the training split.

In addition, several depth-only algorithms, including SICNN are compared in Table 3.7 (bottom). First, a simple pooling approach is considered that takes the closest (distance to sensor) valid point to fill in unseen regions within a given window. Second, the Nadaraya-Watson regressor [80, 115] is applied using a Gaussian kernel on the sparse depth input. For both approaches a hyperparameter search was conducted on the training data. Furthermore, the same CNN baselines as in the previous section, are trained on the real world dataset: a standard Fully Convolutional CNN with and one without visibility mask as additional feature channel.

It is notable that both approaches perform well on the real world dataset, especially in terms of RMSE which indicates that errors are rather well distributed around the actual value. This is especially critical for safety relevant applications. SGDC does not require depth ground truth and produces sharp edges with clear depth discontinuities. The best performing method, The Fast Bilateral Solver by Barron et al. [2], yields the best performance in terms of RMSE but has streaking artifacts at depth edges. SICNN performs second-best overall in terms of RMSE, without using RBG images. This shows the enormous potential of learning-based methods for the task of depth completion if sparsity is handled well. Standard convolutions perform slightly worse, but benefit from being provided a valid depth mask. Surprisingly, the standard convolutions handle the real world data a lot better than

Method	RMSE [m]		MAE [m]	
	val	test	val	test
Bilateral NN [51]	4.19	5.23	1.09	1.09
Fast Bilateral Solver [2]	**1.98**	**1.75**	0.65	0.52
TGVL [26]	4.85	4.08	**0.59**	**0.46**
SGDC	2.50	2.02	0.72	0.57
Closest Depth Pooling	2.77	2.30	0.94	0.68
Nadaraya Watson[80, 115]	2.99	2.86	0.74	0.66
ConvNet	2.97	2.69	0.78	0.62
ConvNet + mask	2.24	1.94	0.79	0.62
SICNN	**2.01**	**1.81**	**0.68**	**0.54**

Table 3.7: Performance comparison on the presented large-scale depth dataset. SGDC and SICNN perform comparable to state-of-the-art methods that incorporate RGB (top), while SICNN outperforms all depth-only variants (bottom).

the experiments with random sampling in the previous section indicate. An educated guess is that the pattern of the Velodyne scanner is much less random than dropping pixels of an HD depth map by chance. The current trend on the Depth Completion Benchmark also shows that sparsity of laser scanner projections is much less random as we might think. With high-capacity neural networks the sparsity pattern can be learned and even yield useful information.

Sparsity Evaluation on KITTI In the KITTI depth completion dataset, a 64-layer LIDAR scanner is used as an input to the completion task. If the point cloud of this scanner is projected onto the image, the depth measurements cover approximately 5% of the image. For industrial applications such as autonomous driving, often low-budget scanners with fewer scanlines, e.g. 32, 16 or fewer layers, are used. This results in very sparse depth maps after projection. To analyze the impact of low resolution depth sensors for the task of depth completion, SGDC, SICNN and several baselines are analyzed w.r.t. different levels of sparsity. For this purpose, all learning-based methods are trained on the original resolution but evaluated on varying density of the input using random dropout.

As depicted in Figure 3.17, both presented methods, SGDC and SICNN can handle the sparser input data well, while standard convolutions as well as the approach of Jampani et al. [51] completely fail to complete the input. The RGB guided methods of Ferstl et al. [26] and Barron et al. [2] perform slightly better than SICNN but require RGB images. It is worth to note that in contrast to the experiments in Section 3.6.1, the GT depth maps

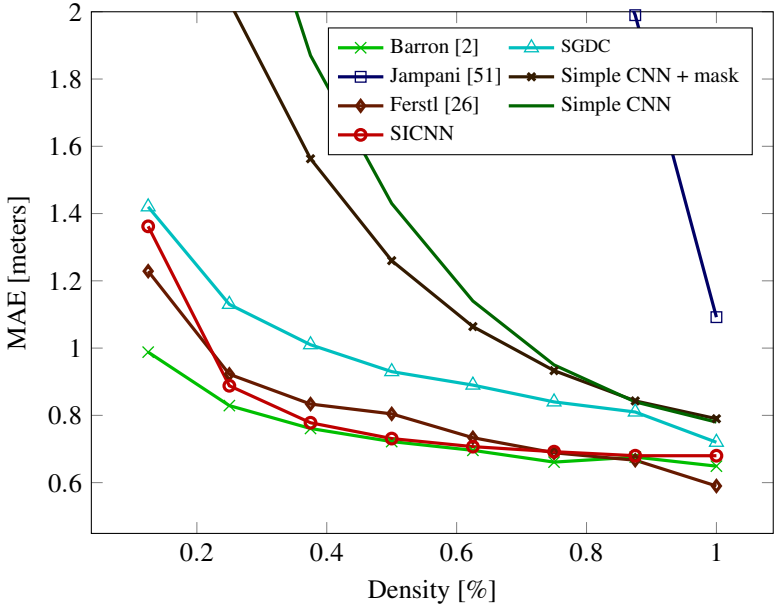


Figure 3.17: Quantitative results on the KITTI depth dataset. Quantitative results in MAE [m] on a subset of the KITTI depth dataset for varying levels of input density. The two presented methods, SGDC and SICNN are compared to several baselines [2, 26, 51] which leverage RGB guidance for depth completion as well as to two standard convolutional neural networks with and without valid mask concatenated to the input. Adapted from [111].

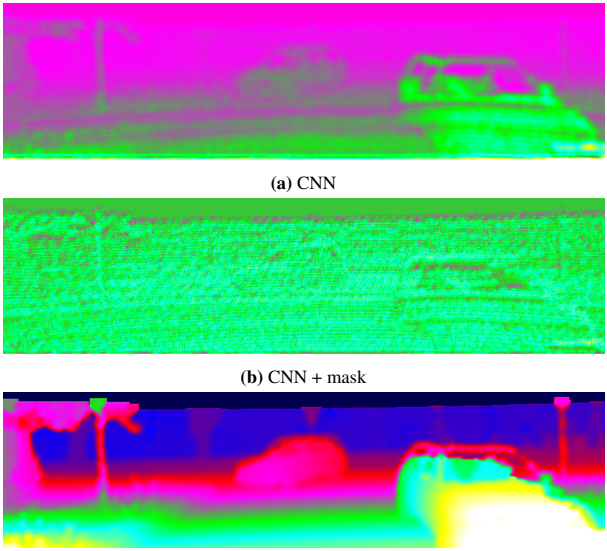
are automatically generated and the sampling strategy for the sparsity level experiment is different. Instead of resembling a lower vertical resolution as in Section 3.6.1, here depth measurements are randomly dropped which benefits the approach of Ferstl et al. [26].

Synthetic-to-Real Domain Adaptation To evaluate the domain adaption capabilities of SICNN, an additional experiment is conducted where models trained on the Synthia dataset are evaluated on the proposed KITTI depth completion validation set.

Table 3.8 shows the performance of SICNN in comparison to the two regular CNN baselines using the same number of learnable parameters. The experiments demonstrate that SICNN performs as well on KITTI as on Synthia,

Train density:	10%	20%	30%	40%	50%	60%	70%
CNN	13.48	10.97	8.44	10.02	9.73	9.57	9.90
CNN+mask	16.44	16.54	16.16	15.64	15.27	14.62	14.11
SICNN	**0.72**	**0.73**	**0.73**	**0.73**	**0.73**	**0.73**	**0.73**

Table 3.8: Sparsity evaluation from Synthia to KITTI. Performance comparison in terms of MAE [m] of different methods trained on different sparsity levels on Synthia and evaluated on the proposed KITTI depth dataset.



(a) CNN

(b) CNN + mask

(c) SICNN

Figure 3.18: Domain adaptation from Synthia to KITTI. Qualitative comparison of the best network variants from Table 3.8 trained on synthetic Synthia [95] and evaluated on the proposed real world KITTI depth dataset. While SICNN adapts well to the novel domain, standard convolutional neural networks fail completely in recovering sensible depth information, even if a mask is concatenated to the input depth map. Adapted from [111].

while the dense baselines are not able to adapt to the new domain and fail completely. Qualitative results of this experiment are shown in Figure 3.18.

3.7 Discussion

In this chapter the task of completing sparse depth measurements was introduced. In order to solve this task, two different depth completion approaches were presented as well as a large-scale dataset to train and evaluate those methods in a typical autonomous driving setting.

The first proposed method, SGDC, exploits boundary cues and pixel-wise semantic class labels obtained via a high resolution guidance image and fully convolutional networks. In order to preserve depth boundaries and fine structures, those cues are combined in a geodesic distance measure. The depth completion task is then formulated as a pixel-wise global energy minimization problem. In order to reduce runtime, a suitable approximation is chosen which allows to reduce the number of parameters to enable real-time optimization. A thorough evaluation on three different public datasets was carried out at different application-levels. Competitive results are achieved, especially in randomly structured scenes. Furthermore, without the semantic cue, SGDC does not rely on large training datasets and is therefore also suited for applications where no ground truth depth is available. SGDC can particularly exploit its strength at very sparse depth measurements or a high target resolution. Here, the performance gap to the baselines increases in terms of both computational demands and depth accuracy. Furthermore, the experiments demonstrate robustness to noise and false depth measurements, e.g. due to occlusions.

However, there still remain some open problems. Especially for the target application of autonomous driving, where only very sparse LIDAR data is available, there are many outliers and the accuracy is still not satisfactory. In order to push the limits even further, a second approach, SICNN, was presented which leverages the potential of deep learning. SICNN introduces sparse convolution modules in order to handle sparse input data. In contrast to regular convolutions these sparse convolutions are invariant to the level of sparsity resulting an improved performance. Furthermore, SICNN generalizes well to novel domains and sparsity levels. Although SICNN does not use RGB images as guidance, it shows superior performance to SGDC on the newly presented large-scale KITTI depth completion dataset. This is mostly due to fewer larger errors which can be critical in the case of autonomous driving. As both approaches have their strengths and weaknesses an interesting future direction is to combine SGDC and SICNN. While today deep learning is essential to achieve state-of-the art results, model based approaches

still have their validity as they introduce physical constrains. Using such a model-based approach but learning the optimization could result in a much better performance while being able to ensure that physical conditions are still considered.

A drawback of deep learning-based approaches like SICNN is that they need a large amount of data to be trained. For the task of depth completion, no actual real world depth completion dataset was available at the time those works were published. Therefore, another contribution was to provide a newly annotated dataset with 93k depth annotated images for training and evaluating depth prediction and depth completion techniques. Additionally, a benchmark for the task of depth completion and single image depth prediction was created. Since the publication of the dataset and the benchmark, the research community achieved considerable advances in the task of depth completion and single image depth prediction. The online benchmark for depth completion now lists over 125 methods (22nd of July 2022) and the number is still increasing.

Both presented methods as well as the dataset and benchmark motivated several follow-up works [10–12, 25, 64, 90, 112, 120]. The trend on the benchmark shows that using RGB images as guidance, larger networks, multiscale features as well as attention-modules are key for an accurate depth completion if structured LIDAR data is used as an input. Worth to mention is the work of Eldesokey et al. [25] which extends SICNN by propagating confidence instead of only binary masks through the network. It is one of the few works which explicitly handles sparse data in the input.

4 Discussion and Outlook

In this dissertation, we worked towards robust data representations for camera and LIDAR sensors in order to exploit their advantages for the task of autonomous driving. In Chapter 2, we discussed the need of an accurate extrinsic calibration as a prerequisite for a successful sensor fusion. We learned that deep neural networks are able to register multimodal sensor data providing both an accurate initial and a continuous online calibration without human interaction. Next, in Chapter 3 we investigated how to combine highly accurate LIDAR scanner measurements with camera features to obtain dense depth maps. To this end, two different depth completion approaches were presented, accompanied by a large-scale depth dataset. In Section 3.3 we learned how we can formulate the problem of depth completion as a set of planes and optimize their parameters by considering image gradients and semantic information in an energy minimization. To push the depth map accuracy even further, we investigated the potential of deep learning for the task of depth completion in Section 3.4. We noticed that standard CNNs fail to handle randomly sparse input and worked towards a sparse convolution module which is invariant to the level of sparsity it is presented. In order to train and evaluate the dense depth methods a large-scale depth dataset in Section 3.5 was introduced which was published together with a public benchmark for the tasks of depth completion and single image depth prediction.

In the subsequent sections, the research questions posed in Section 1.5 are addressed, and the lessons learned are discussed in Section 4.1. Finally, this dissertation is concluded by discussing open research questions and possible future work in Section 4.2.

4.1 Discussion

Convenient and accurate extrinsic calibration An accurate extrinsic calibration of multimodal sensors is a critical prerequisite for most sensor data fusion approaches, including the ones presented in this work. We have

found that available extrinsic calibration approaches are suited for either an accurate initial or a continuous online calibration, but not for both. Furthermore, most initial calibrations are computationally expensive and require human interaction. Therefore, in Chapter 2, we investigated a multimodal sensor registration system which uses deep learning to address those shortcomings. We found that the presented method is able to provide both, an accurate extrinsic post-drive calibration and a continuous online monitoring and improvement of a given calibration. We noticed that the initial calibration can be a very rough estimate of the relative pose which in the range of up to $0\,\text{m} - 1.5\,\text{m}$ for the translational components and $0° - 20°$ for the Euler angles. This indeed enables an accurate and convenient calibration process as it keeps human interaction minimal (only the rough calibration needs to be known). We also learned that deep learning, though barely applied to the task of extrinsic calibration, seems to be a very promising tool for sensor registration. This should encourage further research into this direction, applying multimodal sensor registration to other domains and datasets.

Common representation for images and depth We learned that the high resolution camera grid is a popular choice to fuse raw camera and LIDAR measurements as it is convenient and allows using well-studied image operations and powerful CNNs. However, after projecting the LIDAR depth measurements onto the image, only few pixels are valid which prevents a 1-to-1 correspondence between depth estimates and RGB features. In this dissertation, we have investigated in Chapter 3 how we can obtain high resolution dense depth maps by completing projected sparse depth points. We have learned in Section 3.3 that exploiting boundary cues and pixel-wise semantic information of the RGB image as guidance can yield accurate depth maps with sharp depth edges. Additionally, we have noticed in Section 3.4 that by leveraging deep learning for the task of depth completion, we can further boost the accuracy of the obtained dense depth maps. We also saw that both approaches have their advantages and disadvantages. While the deep learning-based approach is more accurate, it requires access to ground truth data which is often hard to acquire. The optimization-based approach instead does not require ground truth depth maps but can have larger errors in the far distance if the input gets very sparse. In conclusion, the two presented approaches need to be chosen depending on the application and available data at hand.

Handling sparse data with CNNs In Section 3.4 we have investigated the impact randomly sparse input has on conventional CNNs. We found out that naïve methods such as assigning a default value to the missing input or concatenating a valid pixel mask does both lead to unsatisfactory results. For this reason we worked towards a sparsity invariant convolution module and showed that this module can indeed handle different levels of sparsity in the input. Furthermore, we learned that those sparse convolutions generalize well to novel domains and different levels of sparsity. We also looked into applications where handling sparse data is required and saw that by stacking several sparse convolutions to a sparsity invariant CNN we are able to solve tasks like depth completion or semantic segmentation given sparse input data. One of the key aspects we noticed was that those models are able to perform well on a different level of sparsity than the one the network was presented during training. This is an important ability as it enables handling different sensor setups without retraining.

Large-scale depth dataset Due to the increasing success of deep neural networks, large-scale datasets for autonomous driving have become increasingly important. However, we learned that for the task of depth prediction and depth completion no large-scale real world dataset was available. Therefore, the KITTI depth dataset was introduced in Section 3.5. The dataset contains 93k depth annotated images for training and evaluating. Furthermore, we saw a need of a unified comparison of depth estimation methods. Therefore, the large-scale depth dataset is accompanied by a corresponding public benchmark for the task of depth completion and single image depth prediction. The benchmark contains a leaderboard and has test data with ground truth hidden to the public allowing for a fair comparison of submitted depth estimation methods.

4.2 Future Work

In this final section we discuss potential future work that goes beyond the scope of this dissertation. In particular, shortcomings of the presented approaches are addressed, open questions raised and future research directions suggested.

LIDAR Camera Calibration In Chapter 2 an extrinsic calibration approach was presented which is one of the first to ever tackle this task with a CNN. While we have seen promising results, newer and more advanced deep learning methods can have even further potential to push the accuracy of the results. In particular, the use of *transformer-based architectures* [24, 113] seems promising as the attention mechanism could allow for a better learned registration between the multimodal image features and could give more global context than the quite restricted FOV of the CNN which was used in this dissertation. Another point of improvement is the *iterative refinement*. In the presented approach it was used to circumvent the issue of the network tending to focus on large deviations. However, the calibration ranges for the expert networks are chosen manually. Future work could aim at replacing the iterative refinement with an end-to-end trained recurrent network. This could be realized by a differentiable projection layer which allows creating depth maps from the resulting calibration. Another drawback lies in the implementation of the *temporal filtering* where the median over the whole sequence was chosen in order to filter the noisy network estimates over time. However, there might be scenes which are particularly well-suited for calibration. Allowing the network to determine those situations itself, might yield to improved results. Long Short-Term Memory (LSTM) networks might be of interest to improve this part in the future. Finally, restricting the *initialization of the calibration* to have a maximum error of 1.5 m/20° might not be too much of an effort in practice, however, it still requires the involvement of humans in the calibration process. Future work could aim at an approach which operates directly on the raw LIDAR point clouds without requiring an initial projection.

Semantically Guided Depth Completion One of the downsides of the depth completion approach presented in Section 3.3 is the amount of parameters which are used in the optimization. They need to be tuned carefully in order to achieve a good performance. A potential future direction could be to *learn the parameters* of the optimization using the large-scale depth dataset presented in Section 3.5.

Modeling the environment with planes was one of the keys in the presented approach. However, after the approximation, the pixels are clustered to larger planes which can span many objects if no additional depth measurements are available in between. This often results in a bad performance

if the projected point cloud is very sparse which is especially the case in the far distance. Therefore, the approach could be further improved by an intelligent spawning of additional depth points to mitigate this.

Using semantic segmentation to guide the depth completion leads to promising results, but semantic segmentation alone does not allow distinguishing depth jumps between two neighboring objects of the same class. Using a panoptic segmentation approach [59] instead could yield improvements here.

Sparsity Invariant CNNs In Section 3.4 we learned that conventional CNNs are not able to handle randomly sparse data in the input. However, we also found that structured sparse data as for example caused by the projection of depth sensors such as LIDAR sensors causes fewer problems. This raises the question of which forms of sparsity exist and how they affect the performance of the task. Figure 4.1 shows that missing depth measurements do not necessarily result in missing information for a certain area. Also, the regular pattern of the projected LIDAR point cloud of the presented real world dataset can reveal information at locations where no LIDAR beam was returned (e.g. no returns on a car indicate there is a windowed surface). Future research could aim at targeting both the benefits of sparsity invariant convolutions and at the same time also consider missing depth measurements as a source of information. As also pointed out by Eldesokey et al. [25] propagating confidences instead of only binary masks through the network allows adding additional information to the depth map. This is an important feature if the completed depth map is used as an input to another algorithm.

Large-Scale Depth Dataset The presented large-scale depth dataset and benchmark motivated many new approaches for depth completion and single-image depth prediction after publication. While the approaches became better and better in terms of the metrics used for ranking such as RMSE and MAE, most algorithms [10–12, 25, 64, 90, 112, 120] produce blurred depth edges. One of the reasons for that is that the dataset does not have sharp depth edges itself due to the inconsistency of LIDAR and stereo measurements in those areas. Most of the inconsistencies are caused by sensor displacement. Future depth datasets should be encouraged by this finding to provide sharper depth edges and also metrics which consider objects to be sharp especially at depth borders. Another downside of filtering accumu-

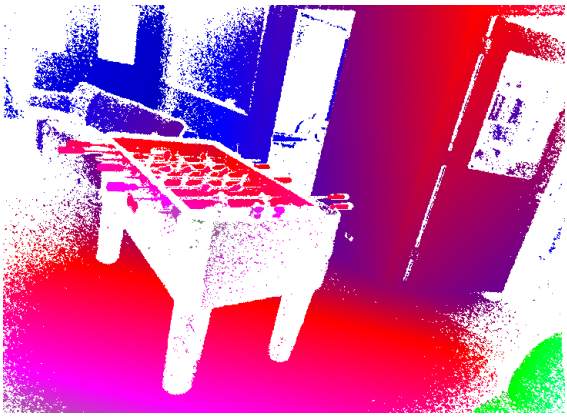


Figure 4.1: Missing data is not missing information. Projecting data of real world sensors often also reveals information at missing locations. For example, some surfaces might reflect infrared light resulting in scatter and hence missing depth measurements. This, however, can be a very important information to a neural network and including the location of pixels with missing information might in this case improve the task at hand.

lated LIDAR point clouds with stereo measurements is that dynamic objects are often very sparse. This bias could result in approaches that perform worse on those objects. A better idea could be to track the position and shape of objects over time and with that provide denser depth for all dynamic objects.

Bibliography

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. “Building Rome in a Day.” In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2009 (cit. on p. 37).

[2] Jonathan T. Barron and Ben Poole. “The Fast Bilateral Solver.” In: *Proc. of the European Conf. on Computer Vision (ECCV)* (2016) (cit. on pp. 42, 82–84).

[3] Stanley Michael Bileschi. “Fully Automatic Calibration of LIDAR and Video Streams from a Vehicle.” In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV) Workshops*. 2009 (cit. on pp. 16, 20).

[4] *Bosch and Mercedes-Benz start San José pilot project for automated ride-hailing service*. Dec. 2019. URL: https://www.bosch-presse.de/pressportal/de/en/bosch-and-mercedes-benz-start-san-jose-pilot-project-for-automated-ridesharing-service-204032.html (cit. on p. 2).

[5] *Bosch Product Sheet Ultrasonic Sensor*. Mar. 2020. URL: https://www.bosch-mobility-solutions.com/media/global/products-and-services/passenger-cars-and-light-commercial-vehicles/driver-assistance-systems/construction-zone-assist/ultrasonic-sensor/onepager_ultrasonic_sensor_en_200203.pdf (cit. on p. 4).

[6] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. “A naturalistic open source movie for optical flow evaluation.” In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2012 (cit. on p. 39).

[7] J. Canny. “A Computational Approach to Edge Detection.” In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 8.6 (Nov. 1986), pp. 679–698 (cit. on p. 70).

[8] Derek Chan, Hylke Buisman, Christian Theobalt, and Sebastian Thrun. “A Noise-Aware Filter for Real-Time Depth Upsampling.” In: *Proc. of the European Conf. on Computer Vision (ECCV) Workshops*. 2008 (cit. on p. 41).

[9] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. “Multi-View 3D Object Detection Network for Autonomous Driving.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017 (cit. on pp. 43, 54).

[10] Yun Chen, Bin Yang, Ming Liang, and Raquel Urtasun. “Learning joint 2d-3d representations for depth completion.” In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2019, pp. 10023–10032 (cit. on pp. 43, 87, 93).

[11] Xinjing Cheng, Peng Wang, Chenye Guan, and Ruigang Yang. “CSPN++: Learning Context and Resource Aware Convolutional Spatial Propagation Networks for Depth Completion.” In: *AAAI Conference on Artificial Intelligence*. AAAI Press, 2020, pp. 10615–10622. URL: https://aaai.org/ojs/index.php/AAAI/article/view/6635 (cit. on pp. 43, 87, 93).

[12] Xinjing Cheng, Peng Wang, and Ruigang Yang. “Depth estimation via affinity learned with convolutional spatial propagation network.” In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 103–119 (cit. on pp. 43, 87, 93).

[13] Xinjing Cheng, Peng Wang, and Ruigang Yang. “Learning Depth with Convolutional Spatial Propagation Network.” In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* (2019) (cit. on p. 10).

[14] Hsiang-Jen Chien, Reinhard Klette, Nick Schneider, and Uwe Franke. “Visual odometry driven online calibration for monocular lidar-camera systems.” In: *Proc. of the International Conf. on Pattern Recognition (ICPR)*. IEEE. 2016, pp. 2848–2853 (cit. on pp. 16, 21).

[15] Taco S. Cohen and Max Welling. “Group Equivariant Convolutional Networks.” In: *Proc. of the International Conf. on Machine learning (ICML)*. 2016 (cit. on p. 44).

[16] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. “The Cityscapes Dataset for Semantic Urban Scene Understanding.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on pp. 6, 10, 60, 74).

[17] Jia Deng, Wei Dong, Richard Socher, Li-jia Li, Kai Li, and Li Fei-fei. “Imagenet: A large-scale hierarchical image database.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2009 (cit. on p. 27).

[18] James Diebel and Sebastian Thrun. “An Application of Markov Random Fields to Range Sensing.” In: *Advances in Neural Information Processing Systems (NIPS)*. 2005 (cit. on p. 42).

[19] Piotr Dollar and C. Lawrence Zitnick. “Fast Edge Detection Using Structured Forests.” In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 37.8 (2015) (cit. on p. 70).

[20] Jennifer Dolson, Jongmin Baek, Christian Plagemann, and Sebastian Thrun. “Upsampling range data in dynamic environments.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2010, pp. 1141–1148 (cit. on p. 41).

[21] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. “Learning a deep convolutional network for image super-resolution.” In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014, pp. 184–199 (cit. on p. 43).

[22] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. “Image Super-Resolution Using Deep Convolutional Networks.” In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 38.2 (2016), pp. 295–307 (cit. on p. 43).

[23] A. Dosovitskiy, P. Fischer, E. Ilg, P. Haeusser, C. Hazirbas, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. “FlowNet: Learning Optical Flow with Convolutional Networks.” In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015 (cit. on p. 27).

[24] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. "An image is worth 16x16 words: Transformers for image recognition at scale." In: *Proc. of the International Conf. on Learning Representations (ICLR)* (2020) (cit. on p. 92).

[25] Abdelrahman Eldesokey, Michael Felsberg, and Fahad Shahbaz Khan. "Confidence propagation through cnns for guided sparse depth regression." In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* (2019) (cit. on pp. 87, 93).

[26] David Ferstl, Christian Reinbacher, Rene Ranftl, Matthias Ruether, and Horst Bischof. "Image Guided Depth Upsampling Using Anisotropic Total Generalized Variation." In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2013, pp. 993–1000 (cit. on pp. 38, 42, 61, 69–76, 82–84).

[27] Mikhail Figurnov, Aizhan Ibraimova, Dmitry P. Vetrov, and Pushmeet Kohli. "PerforatedCNNs: Acceleration through Elimination of Redundant Convolutions." In: *Advances in Neural Information Processing Systems (NIPS)*. 2016 (cit. on p. 44).

[28] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. "Virtual Worlds as Proxy for Multi-Object Tracking Analysis." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 59).

[29] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision meets Robotics: The KITTI Dataset." In: *International Journal of Robotics Research (IJRR)* 32.11 (2013), pp. 1231–1237 (cit. on pp. ix, 29, 61, 69).

[30] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2012 (cit. on pp. 10, 61, 63, 69).

[31] Andreas Geiger, Frank Moosmann, Oemer Car, and Bernhard Schuster. "Automatic Calibration of Range and Camera Sensors using a single Shot." In: *Proc. IEEE International Conf. on Robotics and Automation (ICRA)*. 2012 (cit. on pp. 16, 19, 20, 29).

[32] Daniel Glasner, Shai Bagon, and Michal Irani. “Super-Resolution from a Single Image.” In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2009 (cit. on p. 40).

[33] Xavier Glorot and Yoshua Bengio. “Understanding the difficulty of training deep feedforward neural networks.” In: *Conference on Artificial Intelligence and Statistics (AISTATS)*. 2010 (cit. on p. 29).

[34] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016 (cit. on p. 9).

[35] Ben Graham. “Sparse 3D convolutional neural networks.” In: *Proc. of the British Machine Vision Conf. (BMVC)*. 2015 (cit. on p. 44).

[36] Benjamin Graham. “Spatially-sparse convolutional neural networks.” In: *arXiv.org* (2014) (cit. on p. 44).

[37] Song Han, Jeff Pool, John Tran, and William J. Dally. “Learning both Weights and Connections for Efficient Neural Network.” In: *Advances in Neural Information Processing Systems (NIPS)*. 2015 (cit. on p. 44).

[38] Andrew J. Hawkins. *LIDAR sensors aren’t just for self-driving cars anymore*. Jan. 2020. URL: https://www.theverge.com/2020/1/7/21055011/lidar-sensor-self-driving-mainstream-mass-market-velodyne-ces-2020 (cit. on p. 7).

[39] K. He, J. Sun, and X. Tang. “Guided Image Filtering.” In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2010 (cit. on pp. 41, 73).

[40] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 58).

[41] Jeff Hecht. *Automotive Lidar: Safety questions raised about 1550 nm lidar*. Sept. 2019. URL: https://www.laserfocusworld.com/blogs/article/14040682/safety-questions-raised-about-1550-nm-lidar (cit. on p. 7).

[42] João F. Henriques and Andrea Vedaldi. “Warped Convolutions: Efficient Invariance to Spatial Transformations.” In: *arXiv.org* 1609.04382 (2016) (cit. on p. 44).

[43] Heiko Hirschmüller. “Stereo Processing by Semiglobal Matching and Mutual Information.” In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 30.2 (2008), pp. 328–341 (cit. on pp. 63, 66).

[44] Michael Hornacek, Christoph Rhemann, Margrit Gelautz, and Carsten Rother. “Depth Super Resolution by Rigid Body Self-Similarity in 3D.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2013 (cit. on p. 40).

[45] Jing Huang and Suya You. “Point Cloud Labeling using 3D Convolutional Neural Network.” In: *Proc. of the International Conf. on Pattern Recognition (ICPR)*. 2016 (cit. on p. 10).

[46] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. “The apolloscape dataset for autonomous driving.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 954–960 (cit. on p. 10).

[47] Tak-Wai Hui, Chen Change Loy, and Xiaoou Tang. “Depth Map Super-Resolution by Deep Multi-Scale Guidance.” In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016 (cit. on p. 43).

[48] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. “Flownet 2.0: Evolution of optical flow estimation with deep networks.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2462–2470 (cit. on p. 10).

[49] *It Began With a Race... 16 Years of Velodyne LiDAR*. 2017. URL: `https://velodynelidar.com/blog/it-began-with-a-race16-years-of-velodyne-lidar/` (cit. on p. 7).

[50] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. “Spatial Transformer Networks.” In: *Advances in Neural Information Processing Systems (NIPS)*. 2015 (cit. on p. 44).

[51] Varun Jampani, Martin Kiefel, and Peter V. Gehler. “Learning Sparse High Dimensional Filters: Image Filtering, Dense CRFs and Bilateral Neural Networks.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on pp. 43, 82–84).

[52] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B. Girshick, Sergio Guadarrama, and Trevor Darrell. “Caffe: Convolutional Architecture for Fast Feature Embedding.” In: *Proc. of the International Conf. on Multimedia (ICM)*. 2014 (cit. on p. 29).

[53] “Joint Bilateral Upsampling.” In: *ACM Trans. on Graphics (SIGGRAPH)* 26.3 (2007) (cit. on pp. 38, 41, 73).

[54] Christoph G Keller, Markus Enzweiler, Marcus Rohrbach, David Fernández Llorca, Christoph Schnorr, and Dariu M Gavrila. “The benefits of dense stereo for pedestrian detection.” In: *IEEE Trans. on Intelligent Transportation Systems (TITS)* 12.4 (2011), pp. 1096–1106 (cit. on p. 38).

[55] Alex Kendall, Matthew Grimes, and Roberto Cipolla. “PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization.” In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2015 (cit. on pp. 24, 27).

[56] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. “Accurate Image Super-Resolution Using Very Deep Convolutional Networks.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 43).

[57] W. Kim, V. Anorve, and B.C. Tefft. “American Driving Survey, 2014 – 2017.” In: (2019). URL: https://aaafoundation.org/wp-content/uploads/2019/02/18-0783_AAAFTS-ADS-Brief_r8.pdf (cit. on p. 1).

[58] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization.” In: *Proc. of the International Conf. on Learning Representations (ICLR)*. 2015 (cit. on pp. 29, 75).

[59] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. “Panoptic segmentation.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9404–9413 (cit. on p. 93).

[60] Rolf Köhler, Christian J. Schuler, Bernhard Schölkopf, and Stefan Harmeling. “Mask-Specific Inpainting with Deep Neural Networks.” In: *Proc. of the German Conference on Pattern Recognition (GCPR)*. 2014 (cit. on p. 43).

[61] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet Classification with Deep Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems (NIPS)*. 2012 (cit. on pp. 10, 44, 55).

[62] Lubor Ladicky, Jianbo Shi, and Marc Pollefeys. "Pulling Things out of Perspective." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2014 (cit. on p. 74).

[63] Dmitry Laptev, Nikolay Savinov, Joachim M. Buhmann, and Marc Pollefeys. "TI-POOLING: transformation-invariant pooling for feature learning in Convolutional Neural Networks." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on p. 44).

[64] Sihaeng Lee, Janghyeon Lee, Doyeon Kim, and Junmo Kim. "Deep Architecture With Cross Guidance Between Single Image and Sparse LiDAR Data for Depth Completion." In: *IEEE Access* 8 (2020), pp. 79801–79810 (cit. on pp. 87, 93).

[65] J. Levinson and S. Thrun. "Automatic Online Calibration of Cameras and Lasers." In: *Robotics Science Systems*. 2013 (cit. on pp. 16, 20, 21).

[66] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle Detection from 3D Lidar Using Fully Convolutional Network." In: *Proc. Robotics: Science and Systems (RSS)*. 2016 (cit. on pp. 43, 54).

[67] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. "Multi-task multi-sensor fusion for 3d object detection." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7345–7353 (cit. on pp. 9, 10).

[68] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. "Deep continuous fusion for multi-sensor 3d object detection." In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2018, pp. 641–656 (cit. on pp. 9, 10).

[69] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. "Microsoft COCO: Common Objects in Context." In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014 (cit. on pp. ix, 25, 27).

[70] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. “Sparse Convolutional Neural Networks.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. June 2015 (cit. on p. 44).

[71] Ming-Yu Liu, Oncel Tuzel, and Yuichi Taguchi. “Joint Geodesic Upsampling of Depth Images.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2013 (cit. on pp. 38, 41).

[72] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully Convolutional Networks for Semantic Segmentation.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on pp. 10, 49, 74, 80).

[73] Jiajun Lu and David Forsyth. “Sparse Depth Super Resolution.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015. ISBN: 9781467369640 (cit. on pp. 38, 42).

[74] Oisin Mac Aodha, Neill DF Campbell, Arun Nair, and Gabriel J Brostow. “Patch based synthesis for single depth image super-resolution.” In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2012, pp. 71–84 (cit. on p. 40).

[75] Kiyoshi Matsuo and Yoshimitsu Aoki. “Depth Image Enhancement Using Local Tangent Plane Approximations.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on p. 42).

[76] N. Mayer, E. Ilg, P. Haeusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. “A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation.” In: *CVPR*. 2016 (cit. on pp. 10, 59).

[77] H. H. Meinel. “Evolving automotive radar — From the very beginnings into the future.” In: *IEEE European Conference on Antennas and Propagation*. 2014, pp. 3107–3114 (cit. on p. 4).

[78] Moritz Menze and Andreas Geiger. “Object Scene Flow for Autonomous Vehicles.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on pp. 37, 61, 64–66, 69, 73, 74, 76).

[79] Faraz M Mirzaei, Dimitrios G Kottas, and Stergios I Roumeliotis. "3D LIDAR–camera Intrinsic and Extrinsic Calibration: Identifiability And Analytical Least-squares-based Initialization." In: *International Journal of Robotics Research (IJRR)*. 2012 (cit. on pp. 16, 19, 20).

[80] E.A. Nadaraya. "On Estimating Regression." In: *Theory of Probability and its Applications*. 1964 (cit. on pp. 82, 83).

[81] Hyeonseob Nam and Bohyung Han. "Learning multi-domain convolutional neural networks for visual tracking." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4293–4302 (cit. on p. 10).

[82] World Health Organization. "Global status report on road safety 2018." In: (2018) (cit. on p. 1).

[83] Gaurav Pandey, James R. McBride, Silvio Savarese, and Ryan M. Eustice. "Automatic Extrinsic Calibration of Vision and Lidar by Maximizing Mutual Information." In: 2015 (cit. on pp. 16, 21).

[84] Jaesik Park, Hyeongwoo Kim, Yu-Wing Tai, Michael S. Brown, and In-So Kweon. "High quality depth map upsampling for 3D-TOF cameras." In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2011 (cit. on pp. 42, 70).

[85] Jongsoo Park, Sheng R. Li, Wei Wen, Hai Li, Yiran Chen, and Pradeep Dubey. "Holistic SparseCNN: Forging the Trident of Accuracy, Speed, and Size." In: *arXiv.org* 1608.01409 (2016) (cit. on p. 44).

[86] Georg Petschnigg, Richard Szeliski, Maneesh Agrawala, Michael Cohen, Hugues Hoppe, and Kentaro Toyama. "Digital photography with flash and no-flash image pairs." In: *ACM transactions on graphics (TOG)* 23.3 (2004), pp. 664–672 (cit. on p. 41).

[87] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. "Frustum pointnets for 3d object detection from rgb-d data." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 918–927 (cit. on pp. 10, 11).

[88] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 652–660 (cit. on pp. 10, 11).

[89] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. "Pointnet++: Deep hierarchical feature learning on point sets in a metric space." In: *Advances in Neural Information Processing Systems (NIPS)*. 2017, pp. 5099–5108 (cit. on pp. 10, 11).

[90] Jiaxiong Qiu, Zhaopeng Cui, Yinda Zhang, Xingdi Zhang, Shuaicheng Liu, Bing Zeng, and Marc Pollefeys. "Deeplidar: Deep surface normal guided depth prediction for outdoor scene from sparse lidar data and single color image." In: *CVPR*. 2019, pp. 3313–3322 (cit. on pp. 43, 87, 93).

[91] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui, and Cordelia Schmid. "EpicFlow: Edge-Preserving Interpolation of Correspondences for Optical Flow." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on pp. 49, 50).

[92] Gernot Riegler, David Ferstl, Matthias Rüther, and Horst Bischof. "A deep primal-dual network for guided depth super-resolution." In: 2016 (cit. on p. 42).

[93] Gernot Riegler, Matthias Rüther, and Horst Bischof. "ATGV-net: Accurate depth super-resolution." In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2016, pp. 268–284 (cit. on pp. 43, 57).

[94] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. "OctNet: Learning Deep 3D Representations at High Resolutions." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017 (cit. on p. 44).

[95] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2016 (cit. on pp. 39, 59, 75, 78, 79, 85).

[96] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. “Imagenet large scale visual recognition challenge.” In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252 (cit. on p. 60).

[97] A. Saxena, M. Sun, and A. Y. Ng. “Make3D: Learning 3D Scene Structure from a Single Still Image.” In: *IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI)* 31 (2009), pp. 824–840 (cit. on p. 60).

[98] Daniel Scharstein, Heiko Hirschmüller, York Kitajima, Greg Krathwohl, Nera Nesic, Xi Wang, and Porter Westling. “High-Resolution Stereo Datasets with Subpixel-Accurate Ground Truth.” In: *Proc. of the German Conference on Pattern Recognition (GCPR)*. 2014 (cit. on pp. 39, 60, 69–71).

[99] Daniel Scharstein and Richard Szeliski. “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms.” In: *International Journal of Computer Vision (IJCV)* 47 (2002), pp. 7–42 (cit. on p. 60).

[100] N. Schneider, F. Piewak, C. Stiller, and U. Franke. “RegNet: Multimodal sensor registration using deep neural networks.” In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. 2017, pp. 1803–1810 (cit. on pp. 15, 17, 23, 26, 28, 30–32, 34, 35).

[101] Nick Schneider, Lukas Schneider, Peter Pinggera, Uwe Franke, Marc Pollefeys, and Christoph Stiller. “Semantically Guided Depth Upsampling.” In: *Proc. of the German Conference on Pattern Recognition (GCPR)*. Springer. 2016, pp. 37–48 (cit. on pp. 37, 45, 51, 52, 71, 72, 74, 76).

[102] Patrice Y. Simard, Dave Steinkraus, and John C. Platt. “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis.” In: *ICDAR*. 2003 (cit. on p. 44).

[103] K. Simonyan and A. Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition.” In: *Proc. of the International Conf. on Learning Representations (ICLR)*. 2015 (cit. on p. 58).

[104] Xibin Song, Yuchao Dai, and Xueying Qin. “Deep Depth Super-Resolution: Learning Depth Super-Resolution Using Deep Convolutional Neural Network.” In: *Proc. of the Asian Conf. on Computer Vision (ACCV)*. 2016 (cit. on p. 43).

[105] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. “Going deeper with convolutions.” In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015 (cit. on p. 58).

[106] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022 (cit. on p. 24).

[107] Ashit Talukder and Larry Matthies. “Real-time detection of moving objects from moving vehicles using dense stereo and optical flow.” In: *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*. Vol. 4. IEEE. 2004, pp. 3718–3725 (cit. on p. 38).

[108] Levente Tamas, Robert Frohlich, and Zoltan Kato. “Relative Pose Estimation and Fusion of Omnidirectional and Lidar Cameras.” In: *Proc. of the European Conf. on Computer Vision (ECCV) Workshops*. 2014 (cit. on p. 20).

[109] Levente Tamas and Zoltan Kato. “Targetless Calibration of a Lidar - Perspective Camera Pair.” In: *ICCV Workshops*. 2013 (cit. on p. 20).

[110] Carlo Tomasi and Roberto Manduchi. “Bilateral filtering for gray and color images.” In: *ICCV*. 1998, pp. 839–846 (cit. on p. 41).

[111] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. “Sparsity Invariant CNNs.” In: *International Conference on 3D Vision (3DV)*. 2017 (cit. on pp. 37, 38, 54, 58, 59, 66, 78–80, 84, 85).

[112] Wouter Van Gansbeke, Davy Neven, Bert De Brabandere, and Luc Van Gool. “Sparse and noisy lidar completion with rgb guidance and uncertainty.” In: *International Conference on Machine Vision Applications (MVA)*. IEEE. 2019, pp. 1–6 (cit. on pp. 43, 87, 93).

[113] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. “Attention is All You Need.” In: *Advances in Neural Information Processing Systems (NIPS)*. 2017 (cit. on p. 92).

[114] Lijun Wang, Wanli Ouyang, Xiaogang Wang, and Huchuan Lu. "Visual tracking with fully convolutional networks." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 3119–3127 (cit. on p. 10).

[115] G.S. Watson. "Smooth regression analysis." In: *Sankhyā: The Indian Journal of Statistics*. 1964 (cit. on pp. 82, 83).

[116] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. "Learning Structured Sparsity in Deep Neural Networks." In: *Advances in Neural Information Processing Systems (NIPS)*. 2016 (cit. on p. 44).

[117] Hermann Winner, Stephan Hakuli, Felix Lotz, and Christina Singer, eds. *Handbuch Fahrerassistenzsysteme*. 3rd ed. Springer Vieweg, 2015 (cit. on pp. 5, 6, 8).

[118] Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. "Harmonic Networks: Deep Translation and Rotation Equivariance." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017 (cit. on p. 44).

[119] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. "Pointfusion: Deep sensor fusion for 3d bounding box estimation." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 244–253 (cit. on p. 9).

[120] Yan Xu, Xinge Zhu, Jianping Shi, Guofeng Zhang, Hujun Bao, and Hongsheng Li. "Depth completion from sparse lidar data with depth-normal constraints." In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2019, pp. 2811–2820 (cit. on pp. 43, 87, 93).

[121] Koichiro Yamaguchi, David McAllester, and Raquel Urtasun. "Efficient Joint Segmentation, Occlusion Labeling, Stereo and Flow Estimation." In: *Proc. of the European Conf. on Computer Vision (ECCV)*. 2014 (cit. on p. 53).

[122] Jianchao Yang, John Wright, Thomas S Huang, and Yi Ma. "Image super-resolution via sparse representation." In: *IEEE Trans. on Image Processing (TIP)* 19.11 (2010), pp. 2861–2873 (cit. on p. 43).

[123] Qingxiong Yang, Ruigang Yang, James Davis, and David Nister. "Spatial-Depth Super Resolution for Range Images." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2007 (cit. on p. 41).

[124] Qilong Zhang and R. Pless. "Extrinsic calibration of a camera and laser range finder." In: *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*. 2004 (cit. on p. 16).

[125] Qilong Zhang and Robert Pless. "Extrinsic Calibration of a Camera and Laser Range Finder (improves Camera Calibration)." In: *Proc. IEEE International Conf. on Intelligent Robots and Systems (IROS)*. 2004 (cit. on pp. 19, 20).

[126] Wenwei Zhang, Hui Zhou, Shuyang Sun, Zhe Wang, Jianping Shi, and Chen Change Loy. "Robust Multi-Modality Multi-Object Tracking." In: *Proc. of the IEEE International Conf. on Computer Vision (ICCV)*. 2019, pp. 2365–2374 (cit. on p. 10).

[127] Yanzhao Zhou, Qixiang Ye, Qiang Qiu, and Jianbin Jiao. "Oriented Response Networks." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 2017 (cit. on p. 44).

[128] Shay Zweig and Lior Wolf. "InterpoNet, A brain inspired neural network for optical flow dense interpolation." In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)* (2017) (cit. on p. 43).

Schriftenreihe
Institut für Mess- und Regelungstechnik
Karlsruher Institut für Technologie
(1613-4214)

Band 001 Hans, Annegret
Entwicklung eines Inline-Viskosimeters auf Basis eines magnetisch-induktiven Durchflussmessers.
ISBN 3-937300-02-3

Band 002 Heizmann, Michael
Auswertung von forensischen Riefenspuren mittels automatischer Sichtprüfung.
ISBN 3-937300-05-8

Band 003 Herbst, Jürgen
Zerstörungsfreie Prüfung von Abwasserkanälen mit Klopfschall.
ISBN 3-937300-23-6

Band 004 Kammel, Sören
Deflektometrische Untersuchung spiegelnd reflektierender Freiformflächen.
ISBN 3-937300-28-7

Band 005 Geistler, Alexander
Bordautonome Ortung von Schienenfahrzeugen mit Wirbelstrom-Sensoren.
ISBN 978-3-86644-123-1

Band 006 Horn, Jan
Zweidimensionale Geschwindigkeitsmessung texturierter Oberflächen mit flächenhaften bildgebenden Sensoren.
ISBN 978-3-86644-076-0

Band 007 Hoffmann, Christian
Fahrzeugdetektion durch Fusion monoskopischer Videomerkmale.
ISBN 978-3-86644-139-2

Band 008 Dang, Thao
Kontinuierliche Selbstkalibrierung von Stereokameras.
ISBN 978-3-86644-164-4

Band 009 Kapp, Andreas
Ein Beitrag zur Verbesserung und Erweiterung der Lidar-Signalverarbeitung für Fahrzeuge.
ISBN 978-3-86644-174-3

Band 010 Horbach, Jan
Verfahren zur optischen 3D-Vermessung spiegelnder Oberflächen.
ISBN 978-3-86644-202-3

Band 011 Böhringer, Frank
Gleisselektive Ortung von Schienenfahrzeugen mit bordautonomer Sensorik.
ISBN 978-3-86644-196-5

Band 012 Xin, Binjian
Auswertung und Charakterisierung dreidimensionaler Messdaten technischer Oberflächen mit Riefentexturen.
ISBN 978-3-86644-326-6

Band 013 Cech, Markus
Fahrspurschätzung aus monokularen Bildfolgen für innerstädtische Fahrerassistenzanwendungen.
ISBN 978-3-86644-351-8

Band 014 Speck, Christoph
Automatisierte Auswertung forensischer Spuren auf Patronenhülsen.
ISBN 978-3-86644-365-5

Band 015 Bachmann, Alexander
Dichte Objektsegmentierung in Stereobildfolgen.
ISBN 978-3-86644-541-3

Band 016 Duchow, Christian
Videobasierte Wahrnehmung markierter Kreuzungen mit lokalem Markierungstest und Bayes'scher Modellierung.
ISBN 978-3-86644-630-4

Band 017 Pink, Oliver
Bildbasierte Selbstlokalisierung von Straßenfahrzeugen.
ISBN 978-3-86644-708-0

Band 018 Hensel, Stefan
Wirbelstromsensorbasierte Lokalisierung von Schienenfahrzeugen in topologischen Karten.
ISBN 978-3-86644-749-3

Band 019 Carsten Hasberg
Simultane Lokalisierung und Kartierung spurgeführter Systeme.
ISBN 978-3-86644-831-5

Band 020 Pitzer, Benjamin
Automatic Reconstruction of Textured 3D Models.
ISBN 978-3-86644-805-6

Band 021 Roser, Martin
Modellbasierte und positionsgenaue Erkennung von Regentropfen in Bildfolgen zur Verbesserung von videobasierten Fahrerassistenzfunktionen.
ISBN 978-3-86644-926-8

Band 022 Loose, Heidi
Dreidimensionale Straßenmodelle für Fahrerassistenzsysteme auf Landstraßen.
ISBN 978-3-86644-942-8

Band 023 Rapp, Holger
Reconstruction of Specular Reflective Surfaces using Auto-Calibrating Deflectometry.
ISBN 978-3-86644-966-4

Band 024 Moosmann, Frank
Interlacing Self-Localization, Moving Object Tracking and Mapping for 3D Range Sensors.
ISBN 978-3-86644-977-0

Band 025 Geiger, Andreas
Probabilistic Models for 3D Urban Scene Understanding from Movable Platforms.
ISBN 978-3-7315-0081-0

Band 026 Hörter, Marko
Entwicklung und vergleichende Bewertung einer bildbasierten Markierungslichtsteuerung für Kraftfahrzeuge.
ISBN 978-3-7315-0091-9

Band 027 Kitt, Bernd
Effiziente Schätzung dichter Bewegungsvektorfelder unter Berücksichtigung der Epipolargeometrie zwischen unterschiedlichen Ansichten einer Szene.
ISBN 978-3-7315-0105-3

Band 028 Lategahn, Henning
Mapping and Localization in Urban Environments Using Cameras.
ISBN 978-3-7315-0135-0

Band 029 Tischler, Karin
Informationsfusion für die kooperative Umfeldwahrnehmung vernetzter Fahrzeuge.
ISBN 978-3-7315-0166-4

Band 030 Schmidt, Christian
Fahrstrategien zur Unfallvermeidung im Straßenverkehr für Einzel- und Mehrobjektszenarien.
ISBN 978-3-7315-0198-5

Band 031 Firl, Jonas
Probabilistic Maneuver Recognition in Traffic Scenarios.
ISBN 978-3-7315-0287-6

Band 032 Schönbein, Miriam
Omnidirectional Stereo Vision for Autonomous Vehicles.
ISBN 978-3-7315-0357-6

Band 033 Nicht erschienen

Band 034 Liebner, Martin
Fahrerabsichtserkennung und Risikobewertung für warnende Fahrerassistenzsysteme.
ISBN 978-3-7315-0508-2

Band 035 Ziegler, Julius
Optimale Trajektorienplanung für Automobile.
ISBN 978-3-7315-0553-2

Band 036 Harms, Hannes
Genauigkeitsuntersuchung von binokularen Normalenvektoren für die Umfeldwahrnehmung.
ISBN 978-3-7315-0628-7

Band 037 Ruhhammer, Christian
Inferenz von Kreuzungsinformationen aus Flottendaten.
ISBN 978-3-7315-0721-5

Band 038 Stein, Denis
Mobile laser scanning based determination of railway network topology and branching direction on turnouts.
ISBN 978-3-7315-0743-7

Band 039 Yi, Boliang
Integrated Planning and Control for Collision Avoidance Systems.
ISBN 978-3-7315-0785-7

Band 040 Schwarze, Tobias
Compact Environment Modelling from Unconstrained Camera Platforms.
ISBN 978-3-7315-0801-4

Band 041 Knorr, Moritz
Self-Calibration of Multi-Camera Systems for Vehicle Surround Sensing.
ISBN 978-3-7315-0765-9

Band 042 Rabe, Johannes
Lane-Precise Localization with Production Vehicle Sensors and Application to Augmented Reality Navigation.
ISBN 978-3-7315-0854-0

Band 043 Weiser, Andreas
Probabilistische Vorhersage von Fahrstreifenwechseln für hochautomatisiertes Fahren auf Autobahnen.
ISBN 978-3-7315-0794-9

Band 044 Tian, Wei
Novel Aggregated Solutions for Robust Visual Tracking in Traffic Scenarios.
ISBN 978-3-7315-0915-8

Band 045 Gräter, Johannes
Monokulare Visuelle Odometrie auf Multisensorplattformen für autonome Fahrzeuge.
ISBN 978-3-7315-0935-6

Band 046 Spindler, Max
Ferromagnetische Inhomogenitäten zur berührungslosen Bestimmung der Geschwindigkeit und gleisselektiven Position von Schienenfahrzeugen.
ISBN 978-3-7315-1000-0

Band 047 Sons, Marc
Automatische Erzeugung langzeitverfügbarer Punktmerkmalskarten zur robusten Lokalisierung mit Multi-Kamera-Systemen für automatisierte Fahrzeuge.
ISBN 978-3-7315-1029-1

Band 048 Hubmann, Constantin
Belief State Planning for Autonomous Driving: Planning with Interaction, Uncertain Prediction and Uncertain Perception.
ISBN 978-3-7315-1039-0

Band 049 Naumann, Maximilian
Probabilistic Motion Planning for Automated Vehicles.
ISBN 978-3-7315-1070-3

Band 050 Tas, Ömer Sahin
Motion Planning for Autonomous Vehicles in Partially Observable Environments.
ISBN 978-3-7315-1299-8

Band 051 Orzechowski, Piotr Franciszek
Verhaltensentscheidung für automatisierte Fahrzeuge mittels Arbitrationsgraphen.
ISBN 978-3-7315-1312-4

Band 052 Schneider, Nick
Deep Fusion of Camera and LIDAR.
ISBN 978-3-7315-1361-2

Schriftenreihe
INSTITUT FÜR MESS- UND REGELUNGSTECHNIK
KARLSRUHER INSTITUT FÜR TECHNOLOGIE (KIT)

052

Fusing camera and LIDAR data in the context of autonomous driving poses significant challenges. First, a convenient yet accurate calibration is required. Second, data representations are critical as camera and LIDAR differ significantly. Finally, as autonomous driving is heavily based on machine learning, a large amount of training and evaluation data is required. In this book, those challenges are addressed by proposing three different methods, each accompanied by associated experiments. First, a method for LIDAR-to-camera calibration is presented which is the first to successfully apply a deep neural network for calibration and with that replaces all common calibration steps. Furthermore, two different depth completion approaches are presented, which allow processing depth measurements in the same space as the RGB image. Both methods perform well on various domains. Finally, a large-scale depth dataset is introduced which consists of 93k RGB and depth images and can be used to train and evaluate camera and LIDAR-based algorithms.

ISSN 1613-4214
ISBN 978-3-7315-1361-2

Gedruckt auf FSC-zertifiziertem Papier