Sensor fusion with deep neural networks for 3D object detection for autonomous vehicles

Nguyen Anh Minh Mai

Université Fédérale
Toulouse Midi-Pyrénées

THÈSE

En vue de l'obtention du
DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE
Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Nguyen Anh Minh MAI

Le 26 janvier 2023

Fusion de capteurs par réseaux de neurones profonds pour la détection d'objets 3D dans l'environnement des véhicules autonomes

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Informatique et Télécommunications**

Unité de recherche :
IRIT : Institut de Recherche en Informatique de Toulouse

Thèse dirigée par
Denis KOUAMÉ et Louahdi KHOUDOUR

Jury

M. Fadi Dornaika, Rapporteur
M. Denis Hamad, Rapporteur
M. Frédéric LERASLE, Examinateur
Mme Samia Bouchafa-Bruneau, Examinatrice
M. Louahdi KHOUDOUR, Co-directeur de thèse

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *le 26 janvier 2023* **par :**

Nguyen Anh Minh MAI

Fusion de capteurs par réseaux de neurones profonds pour la détection d'objets 3D dans l'environnement des véhicules autonomes

Sensor fusion with deep neural networks for 3D object detection for autonomous vehicles

JURY

Denis KOUAMÉ	Professeur, Université Toulouse III - Paul Sabatier	Directeur de Thèse
Louahdi KHOUDOUR	Directeur de Recherche, Cerema	Co-Directeur de Thèse
Alain CROUZIL	Maître de Conférences, Université Toulouse III – Paul Sabatier	Encadrant
Pierre DUTHON	Chargé de Recherche, Cerema	Encadrant
Pascal HOUSAM SALMANE	Chargé de Recherche, Cerema	Encadrant
Fadi DORNAIKA	Professeur, Université du Pays Basque UPV/EHU	Rapporteur
Denis HAMAD	Professeur, Université du Littoral Côte d'Opale	Rapporteur
Samia BOUCHAFA-BRUNEAU	Professeure, Université d'Evry-Val-d'Essone - Université Paris-Saclay	Examinatrice
Frédéric LERASLE	Professeur, Université Toulouse III – Paul Sabatier	Examinateur
Sergio A VELASTIN	Visiting Professeur, Université Queen Mary of London	Invité

École doctorale et spécialité :
MITT : Image, Information, Hypermédia

Unité de Recherche :
Institut de Recherche en Informatique de Toulouse (UMR 5505)

Directeur(s) de Thèse :
Denis KOUAMÉ et *Louahdi KHOUDOUR*

Rapporteurs :
Fadi DORNAIKA et *Denis HAMAD*

UNIVERSITÉ TOULOUSE III – PAUL SABATIER

Résumé

Doctorat

Fusion de capteurs par réseaux de neurones profonds pour la détection d'objets 3D dans l'environnement des véhicules autonomes

par Nguyen Anh Minh MAI

L'objectif principal de cette thèse est de détecter des objets 3D dans des scènes routières en présence de conditions climatiques défavorables comme le brouillard, avec un nombre d'objets multiple. Un écart de performance entre les méthodes basées sur le LiDAR et celles sur les caméras ou la fusion est observé. Les méthodes basées sur la fusion caméra+LiDAR doivent gérer simultanément plusieurs sources de données. Nous examinons en détail les techniques avancées de la littérature sur la détection d'objets 3D pour les véhicules autonomes. Nous proposons ensuite une nouvelle approche basée sur la fusion pour la détection de ces objets. Un premier problème est de savoir comment fusionner efficacement des images et des données sous forme de nuages de points dans une architecture unique qui sera capable d'apprendre des représentations de haut niveau à partir d'un réseau de neurones profond et d'améliorer les capacités de détection. Une deuxième question est de savoir comment les conditions météorologiques défavorables affectent les capteurs et les performances du modèle de détection, et quelles données doivent être utilisées dans le modèle en fonction de ces conditions défavorables ? Cela a abouti à l'introduction d'une nouvelle technique de détection d'objets 3D appelée SLS-Fusion (Sparse LiDAR and Stereo Fusion), qui utilise une caméra stéréo et un LiDAR pour prédire une carte de profondeurs. Cette dernière est ensuite convertie en pseudo nuage de points à des fins de traitements. Afin d'obtenir des boîtes englobantes 3D, le pseudo nuage de points peut être utilisé avec n'importe quelle méthode actuelle de détection d'objets basée sur le LiDAR. Notre architecture peut améliorer à la fois l'estimation de la profondeur et la précision de la détection d'objets 3D. Les résultats expérimentaux sur des ensembles de données publiques (KITTI) montrent que l'approche proposée surpasse l'état de l'art actuel. La détection d'objets 3D par temps de brouillard a aussi été traité. Un jeu de données artificiel a été crée (fogification de la base KITTI) avec des distances de visibilité variables (Base Multifog KITTI est la résultante allant de 10m à 80m de visibilité). Comme pour KITTI, 7481 images d'apprentissage et 7518 images de test ont été utilisées. Les meilleurs résultats obtenus sont ceux qui utilisent des données d'apprentissage avec du brouillard quand on cherche à détecter des obstacles en présence de brouillard. Nous avons également analysé d'autres aspects : l'apport des deux types de capteurs aussi bien par temps favorable que par temps de brouillard, lorsqu'ils sont fusionnés et lorsqu'ils sont utilisés séparément. Le résultat principal est que l'utilisation du LiDAR par temps de brouillard conduit à une performance de détection d'objets assez mauvaise (surtout avec un LiDAR 4 nappes).

Les résultats basés sur la caméra stéréo sont prometteurs par temps de brouillard, quel que soit le niveau de visibilité. Dans une étude d'ablation, la contribution d'une caméra stéréo et de différentes versions de LiDAR (4 à 64 nappes) aux performances du modèle SLS-Fusion dans la détection d'obstacles 3D est analysée. Les meilleurs résultats obtenus sont ceux issus de la fusion caméra+LiDAR. Les résultats quantitatifs ont montré que les performances de détection chutent raisonnablement lorsqu'on diminue le nombre de nappes des LiDAR dans le processus de fusion avec la caméra. Ces résultats ouvrent de nouvelles directions de recherche pour la détection d'objets 3D pour la conduite autonome en combinant des images de caméra stéréo avec des nuages de points LiDAR. De plus, nous générons et introduisons dans cette thèse le jeu de données Multifog KITTI, une nouvelle base de données sur les conditions de brouillard qui contient à la fois des images et des nuages de points. Cette base pourra être utile à la communauté des chercheurs du domaine à des fins de comparaison.

Mots clés : véhicule autonome, apprentissage profond, détection d'objets 3D, fusion de capteurs, nuage de points, conditions météorologiques défavorables.

UNIVERSITÉ TOULOUSE III – PAUL SABATIER

Abstract

Doctor of Philosophy

Sensor fusion with deep neural networks for 3D object detection for autonomous vehicles

by Nguyen Anh Minh MAI

This thesis deals with 3D object detection for autonomous driving, using various data sources. The main objective is to detect 3D objects in driving scenes. There are a number of factors that challenge this task, including the variability of conditions as well as the number of objects, lighting, and weather factors. A performance gap exists between methods based on LiDAR and those based on cameras or fusion. In contrast to camera-based methods, which are troubled by the lack of depth information, fusion-based methods have the problem of multiple data sources (such as camera, RADAR, and LiDAR). To address these challenges, we review and evaluate the most prominent state-of-the-art techniques to assess the current state of 3D object detection in autonomous vehicles. We then propose a new fusion based approach for 3D object detection. Two key questions have been addressed. The first concern is how efficiently fuse images and point cloud data in a single architecture that will be able to learn high-level representatives from the deep neural networks and result in improved detection abilities. A second question is how adverse weather conditions affect sensors, how does it affect the performance of the detection model, as well as what data should be used in the model based on these adverse conditions? This resulted in the introduction of a new 3D object detection technique called SLS-Fusion (Sparse LiDAR and Stereo Fusion), which uses a stereo camera and LiDAR to predict a depth map. This depth map is then converted into a pseudo point cloud by using camera-LiDAR extrinsic parameters. Finally, this pseudo point cloud can be used with any current state-of-the-art LiDAR-based object detection method to obtain 3D bounding boxes. Our architecture can improve both depth estimation and 3D object detection accuracy. Experimental results on public datasets (KITTI) show that the proposed approach outperforms the current state-of-the-art. We also conducted research on the problem of detection in foggy weather conditions. To do this, we have created a foggy dataset called Multifog KITTI. This dataset is augmented on the KITTI dataset. Like KITTI, It includes 7,481 frames for training and 7,518 frames for testing with fog intensity (from 20 m to 80 m visibility) applied. In these conditions, the model's performance drop, however shows a marked improvement when training with additional foggy data. We have also analyzed several aspects: the contribution of the two types of sensors both in favorable weather and in foggy weather conditions, when they are fused and when they are used separately. The main result

is that using LiDAR in foggy weather leads to a slightly bad object detection performance (even worse when the LiDAR is a 4-beam laser sensor). On the other hand, results based on stereo camera are promising in foggy weather, regardless of the level of visibility. In an ablation study, the contribution of a stereo camera and different versions of LiDAR (4 to 64 beams) to the performance of the SLS-fusion model in detecting 3D obstacles is analyzed. Based on our ablation analysis and the different measurements used to evaluate our detection algorithms, we have shown that sensors should always be unseparated for better performance. Quantitative results have shown that detection performance drops reasonably with each component disabled (stereo camera or LiDAR) or by modifying the number of LiDAR beams.

These findings open new research directions for 3D object detection for autonomous driving by combining stereo camera images with LiDAR point clouds. Additionally, we generate and introduce in this thesis the Multifog KITTI dataset, a new foggy weather conditions dataset that contains both images and point clouds.

Keywords: autonomous vehicle, deep learning, 3D object detection, sensor fusion, point cloud, adverse weather conditions.

Remerciements

Je saisis cette occasion pour exprimer ma profonde gratitude à toutes les personnes qui m'ont aidé et soutenu durant mon parcours doctoral.

Tout d'abord, je voudrais adresser mes sincères remerciements à mes superviseurs, Louahdi Khoudour, Alain Crouzil, Pierre Duthon et Pascal Housam Salmane, pour leur accompagnement indéfectible, leur soutien et leur encouragement tout au long de mes recherches. Leur expertise, leur savoir et leur perspicacité ont joué un rôle clé dans la conception de mon travail et ma méthodologie de recherche. Je suis également reconnaissant pour leur disponibilité et leur volonté de fournir des commentaires constructifs et des conseils à tout moment.

Je suis également profondément redevable à mon membre de comité, Sergio A. Velastin, pour ses précieuses suggestions, ses commentaires critiques sur ma thèse et ses réflexions. Ses commentaires m'ont aidé à affiner mes questions de recherche, mes méthodes et mes conclusions. Je suis reconnaissant de son temps et de ses efforts.

Je tiens également à reconnaître le soutien financier que j'ai reçu de CEREMA. Leur soutien m'a permis de me concentrer sur mes recherches et a contribué à la réussite de ma thèse de doctorat.

Mes recherches n'auraient pas été possibles sans la participation des nombreuses personnes qui ont généreusement consacré leur temps et leur savoir. Je suis reconnaissant à chacun des participants pour leurs contributions précieuses et leurs idées, qui m'ont permis d'approfondir ma compréhension du sujet et de ses nuances.

Je voudrais également exprimer ma sincère reconnaissance à ma famille et mes amis pour leur amour, leur soutien et leur encouragement indéfectibles tout au long de ce parcours. Leur patience et leur compréhension ont été essentielles pour me maintenir motivé et concentré durant les moments difficiles. Je suis reconnaissant de leur soutien et encouragement constants.

Enfin, je tiens à remercier toutes les personnes qui m'ont aidé de près ou de loin. Qu'il s'agisse de me fournir un accès aux données ou aux ressources, de me soutenir moralement ou de me donner des conseils, je suis profondément reconnaissant de leur contribution et de leur soutien.

L'achèvement de cette thèse de doctorat a été un parcours long et difficile, mais avec l'aide de ces personnes bienveillantes, j'ai pu atteindre mon objectif. Leurs contributions ont été inestimables, et je suis reconnaissant pour les nombreuses façons dont ils m'ont aidé tout au long du chemin.

En conclusion, je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont joué un rôle dans mon parcours pour m'aider à atteindre ce point. Votre soutien,

vos conseils et votre encouragement ont été cruciaux pour ma réussite, et je suis vraiment reconnaissant de tout ce que vous avez fait. Merci beaucoup.

Toulouse, France, le 26 Janvier 2022

Nguyen Anh Minh MAI

Acknowledgements

I would like to take this opportunity to express my profound gratitude to all those who have helped and supported me during my doctoral journey.

Firstly, I extend my heartfelt appreciation to my supervisors, Louahdi Khoudour, Alain Crouzil, Pierre Duthon and Pascal Housam Salmane, for their unwavering guidance, support, and encouragement throughout my research. Their expertise, knowledge, and insights have been instrumental in shaping my work and research approach. I am also grateful for their availability and willingness to provide constructive feedback and advice at all times.

I am also deeply indebted to my committee member, Sergio A. Velastin, for his invaluable insights, suggestions, and critical comments on my thesis. His feedback has helped me refine my research questions, methods, and findings, and I am grateful for his time and effort.

I would also like to acknowledge the financial support I received from CEREMA. Their support made it possible for me to focus on my research and contributed to the successful completion of my PhD.

My research would not have been possible without the participation of the many individuals who generously volunteered their time and knowledge. I am grateful to each and every participant for their valuable contributions and insights, which helped me gain a deeper understanding of the topic and its nuances.

I would also like to express my sincere appreciation to my family and friends for their unwavering love, support, and encouragement throughout this journey. Their patience and understanding have been instrumental in keeping me motivated and focused during challenging times. I am grateful for their unwavering support and encouragement.

Finally, I would like to extend my thanks to all those who have helped me in ways big and small. From providing me with access to data or resources to offering moral support and advice, I am deeply grateful for your contributions and support.

Completing this PhD has been a long and challenging journey, but with the help of these supportive individuals, I was able to achieve my goal. Their contributions have been invaluable, and I am grateful for the many ways in which they have helped me along the way.

In conclusion, I want to express my deepest gratitude to everyone who has played a role in helping me reach this point. Your support, guidance, and encouragement have been critical to my success, and I am truly grateful for everything you have done. Thank you.

Toulouse, France, January 26, 2022

À mes parents . . .

Author's Publications

I hereby declare that the following publications have been produced during the course of this thesis:

- Pascal Housam Salmane, Josué Manuel Rivera Velázquez, Louahdi Khoudour, **Nguyen Anh Minh Mai**, Pierre Duthon, Alain Crouzil, Guillaume Saint Pierre, Sergio A. Velastin. "3D Object Detection with SLS-Fusion Network in Foggy Weather Conditions". In: SENSORS 23.6 (2023). DOI: 10.3390/s23063223.
- **Nguyen Anh Minh Mai**, Pierre Duthon, Pascal Housam Salmane, Louahdi Khoudour, Alain Crouzil, Sergio A. Velastin. "Camera and LiDAR analysis for 3D object detection in foggy weather conditions". In: Proceedings of the International Conference on Pattern Recognition Systems (ICPRS). 2022, pp. 1–7. DOI: 10.1109/ICPRS54038.2022.9854073. HAL: hal-03820137.
- **Nguyen Anh Minh Mai**, Pierre Duthon, Louahdi Khoudour, Alain Crouzil, Sergio A. Velastin 2021. "3D Object Detection with SLS-Fusion Network in Foggy Weather Conditions". In: SENSORS 21.20 (2021). DOI: 10.3390/s21206711.
- **Nguyen Anh Minh Mai**, Pierre Duthon, Louahdi Khoudour, Alain Crouzil, Sergio A. Velastin 2021. "Détection d'obstacles par vision et LiDAR par temps de brouillard pour les véhicules autonomes", In: Proceedings of journées francophones des jeunes chercheurs en vision par ordinateur (ORASIS). 2021. HAL: hal-03339637.
- **Nguyen Anh Minh Mai**, Pierre Duthon, Louahdi Khoudour, Alain Crouzil, Sergio A. Velastin. "Sparse LiDAR and Stereo Fusion (SLS-Fusion) for Depth Estimation and 3D Object Detection"In: Proceedings of the International Conference of Pattern Recognition Systems (ICPRS). Vol. 2021. 2021, pp. 150–156. DOI: 10.1049/icp.2021.1442. arXiv: 2103.03977.

Author's Google Scholar profile:

https://scholar.google.com/citations?user=lxTC9IIAAAAJ&hl

Contents

List of Figures

List of Tables

List of Abbreviations

AP	**A**verage **P**recision
BEV	**B**ird's **E**ye **V**iew
CIE	**I**nternational **C**ommission on **I**llumination
CNN	**C**onvolutional **N**eural **N**etwork
DeCV	**De**pth **C**ost **V**olume
DiCV	**Di**sparity **C**ost **V**olume
FC	**F**ully **C**onnected
FV	**F**ront **V**iew
GAN	**G**enerative **A**dversarial **N**etwork
GDC	**G**raph-based **D**epth **C**orrection
HOG	**H**istogram of **O**riented **G**radients
iMAE	**M**ean **A**bsolute **E**rror of the **i**nverse depth
iRMSE	**R**oot **M**ean **S**quared **E**rror of the **i**nverse depth
IoU	**I**ntersection **o**ver **U**nion
KNN	**K**-**N**earest **N**eighbor
KOD	**K**ITTI **O**bject **D**etection
LiDAR	**Li**ght **D**etection **a**nd **R**anging
MAE	**M**ean **A**bsolute **E**rror
MLP	**M**ulti-**l**ayer **P**erceptron
MOR	**M**eteorological **O**ptical **R**ange
NMS	**N**on-**M**aximum **S**uppression
PL	**P**seudo-**L**iDAR
RADAR	**Ra**dio **D**etection **a**nd **R**anging
RMSE	**R**oot **M**ean **S**quared **E**rror
RNN	**R**ecurrent **N**eural **N**etwork
RoI	**R**egion **o**f **I**nterest
RPN	**R**egion **P**roposal **N**etwork
RV	**R**ange **V**iew
SDN	**S**tereo **D**epth **N**etwork
SLS-Fusion	**S**parse **L**iDAR and **S**tereo **F**usion
SOTA	**S**tate-**o**f-**t**he-**a**rt
SV	**S**pherical **V**iew
SVM	**S**upport **V**ector **M**achine
VFE	**V**oxel **F**eature **E**ncoding
VOC	**V**isual **O**bject **C**lasses

Chapter 1

Introduction

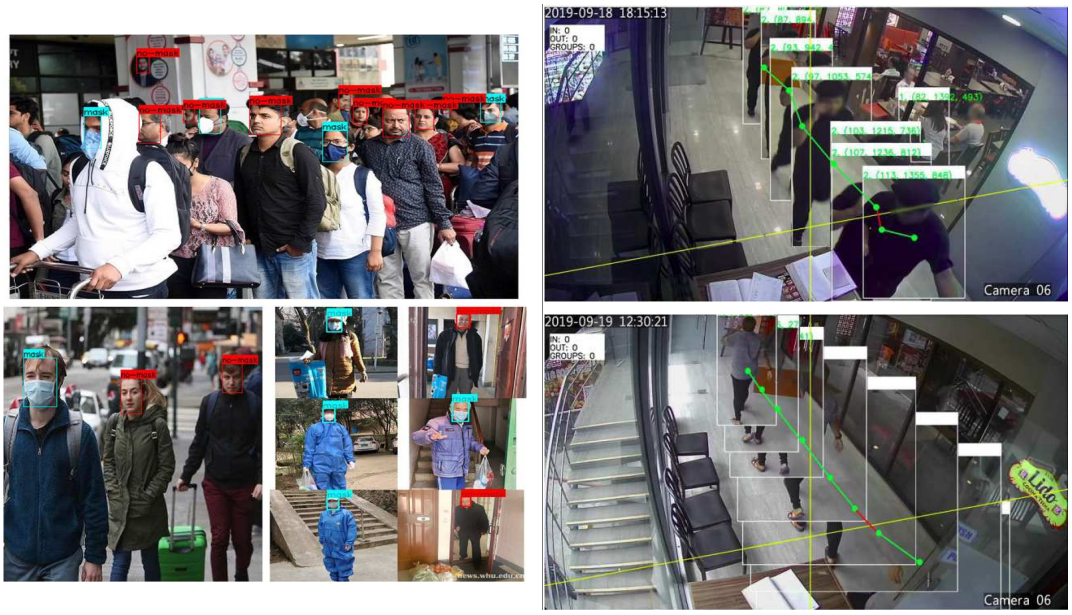
Contents

Chapter overview:

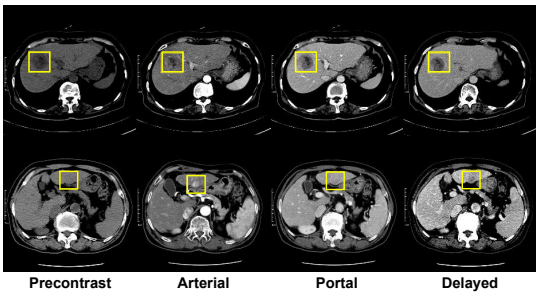
Self-driving cars rely on automated tasks that a human driver would typically perform, such as detecting surrounding objects, avoiding lanes, and recognizing traffic signs. Among these tasks, 3D object detection is the most crucial for safe and efficient self-driving. However, there is limited research on the best methods for fusing sensors, such as cameras and LiDAR, to detect and localize objects effectively, especially in adverse weather conditions where these sensors may be significantly impacted. This research aims to design and analyze deep learning-based methods that fuse sensors in the same model for 3D object detection in self-driving cars. As the field of knowledge is continuously evolving and expanding, the proposed methods will be updated and improved to address new challenges and emerging technologies.

1.1 Motivation and Context

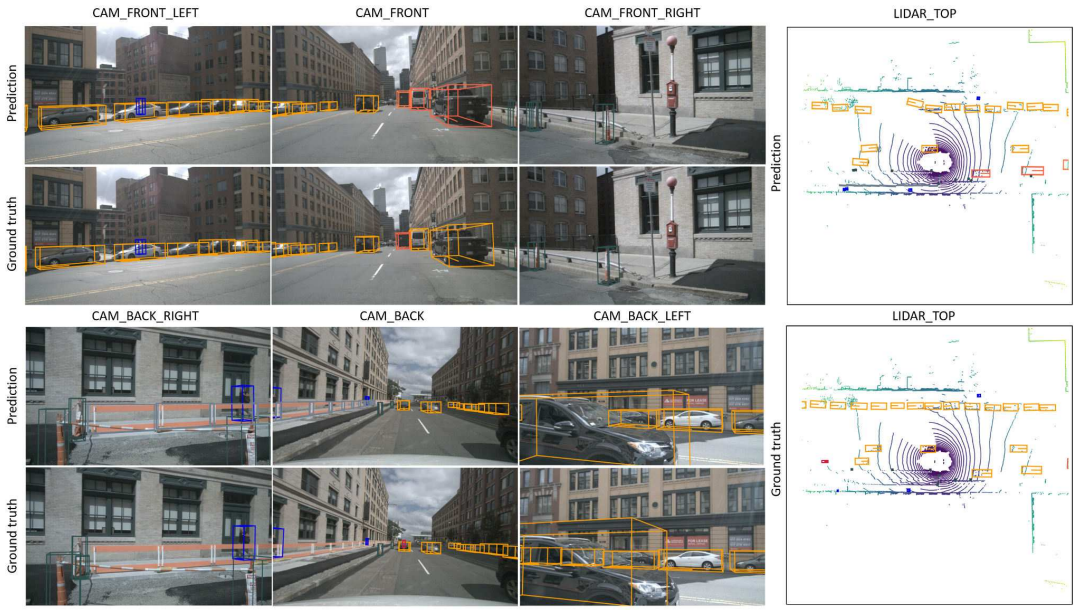
There is currently a growing trend towards integrating automation into various applications to reduce costs, alleviate human workload, and improve efficiency. Some examples of these applications are shown in Figure 1.1, including face recognition [1], people counting [2], liver segmentation [3], 3D object detection [4], and sensor fusion [5]. Unlike humans, who perceive the world through their senses of sight, hearing, smell, and touch, industrial applications collect data using sensors such as cameras, RADAR (Radio Detection and Ranging), Kinect, LiDAR (Light Detection and Ranging), and IMU (Inertial Measurement Unit). The gathered data is then processed by complex algorithms.



(a) (b)



(c)



(d)

FIGURE 1.1: **Some relevant applications of object detection.** (a) face mask detection [1, Fig. 6]; (b) people counting in retail [2, Fig. 3]; (c) liver lesion detection [3, Fig. 1]; (d) autonomous driving [4, Fig. 5].

In the context of autonomous vehicles, the ability to perceive the environment in which the vehicle is moving is crucial, as there is no human driver present. Ensuring the safety of the vehicle as it travels from point A to point B is a top priority. To monitor the environment, autonomous vehicles typically use a combination of sensors, including cameras and LiDAR.

Before discussing the specific contributions of this PhD thesis, let's provide an overview of autonomous vehicles. In the context of autonomous vehicles, perceiving the environment in which the vehicle is operating is crucial, as there is no human driver present to respond to unexpected situations. Ensuring the safety of the vehicle as it travels from point A to point B is a top priority. To monitor the environment, autonomous vehicles typically use a combination of sensors, including cameras and LiDAR.

Before discussing the specific contributions of this PhD thesis, let us provide an overview of autonomous vehicles.

What is an autonomous vehicle?

An autonomous vehicle is a car equipped with an automatic control system that enables it to drive without human intervention in real-world traffic conditions. In simpler terms, an autonomous vehicle is able to sense its environment and operate without human input. Automation refers to the use of mechanical, pneumatic, hydraulic, or electric systems to perform tasks that would normally be carried out by humans.

To achieve this understanding, an autonomous vehicle must have a precise understanding of its environment. This is accomplished through the use of sensors, highly accurate and constantly updated maps, communication systems, and artificial intelligence. Each of these components plays a crucial role in the operation of the autonomous vehicle. Sensors such as cameras, radars, and LiDAR provide a detailed view of the vehicle's surroundings. Communication systems allow for the exchange of information about the environment and other vehicles or individuals encountered during the journey. Artificial intelligence collects and analyzes all of this information, enabling the vehicle to make decisions and take actions similar to those of a human driver. Each of these components is crucial for the vehicle to be able to operate safely and effectively in its environment.

Autonomous vehicles are safer than traditional vehicles. The primary goal of autonomous vehicles is to provide passengers with a higher level of safety than if there were a human driver. Sensors and electronics can react faster than human drivers, allowing for quicker reaction times. In addition, technology can eliminate certain human-specific factors that may affect safety, such as drowsiness or distraction. As sensors and electronics become more advanced, they will provide a more comprehensive view of the environment and allow for even faster and more accurate responses, further improving safety on the roads.

A brief history of autonomous vehicles: It is useful to provide a chronological overview of the history of automation in the automotive manufacturing industry. While automation has been present in our lives for many years, it may not always be immediately apparent. This thesis will focus specifically on the automation of land vehicles, including autonomous driving technology. The history of automation in public transportation began with the introduction of People Moovers, small trains running on rails, at the New York World's Fair in 1964 (as shown in Figure 1.2). These vehicles were significant because they were one of the first examples of automated transportation available to the general public. A few years later, the Paris Metro Line 1 and the London Victoria Line revolutionized transportation with automated metros, with sensors on the rails regulating their speeds and trajectories. However, in the early days, these metros always had a driver present who could take control in case of emergencies. The first driverless automated metro was inaugurated in Kobe, Japan in 1981, which reduced labor costs and increased efficiency. This practice quickly spread to other cities such as Lille in 1983 and Paris (Orlyval) in 1991. As of 2016, cities around the world have nearly 55 automated metro lines, with some regions leading in the adoption of automation.

FIGURE 1.2: **People Moovers in New York in 1964.**

The car case. Now let us look at the cars themselves. The very first self-driving car was presented in 1921 in the United States. However, this vehicle cannot be considered the first autonomous car, but rather the first remote-controlled car. Indeed, a truck, located at the rear of the car, controlled the entire vehicle, from its speed to its trajectory through the steering wheel. We were able to start really talking about vehicle automation in the 1939s, with the appearance of the automatic transmission, cruise control, and braking assistance.

In 1977, Japanese researchers piloted an autonomous vehicle on a suitable circuit using trajectory tracking with lane recognition technology. It was operating at a maximum speed of 30 km/h. A few years later, in 1984, the Mercedes-Benz group managed to circulate an automatic van, called Vamors, with a speed of 100 km/h without traffic, thanks to the use of camera sensor. The experiment was carried out under the supervision of Ernest Dickmanns at the University of the Bundeswehr in Munich.

In 1987, the automobile industry took a major turn thanks to the establishment of the European Commission of the European program Prometheus, (Program for European Traffic of Highest Efficiency and Unprecedented Safety). The aim of this program is to develop concepts and solutions capable of making road traffic safer, more efficient, more cost-effective and less polluting. This program lasted until 1995 and made it possible to learn more about the vehicle's range and to define multiple telematics applications supported by the intelligent vehicle concept.

All of these discoveries were remarkable at the time, but technology, went through one of its biggest turning points in the 2000s with the arrival of the widely used navigation system and the most famous, GPS (Global Positioning System).

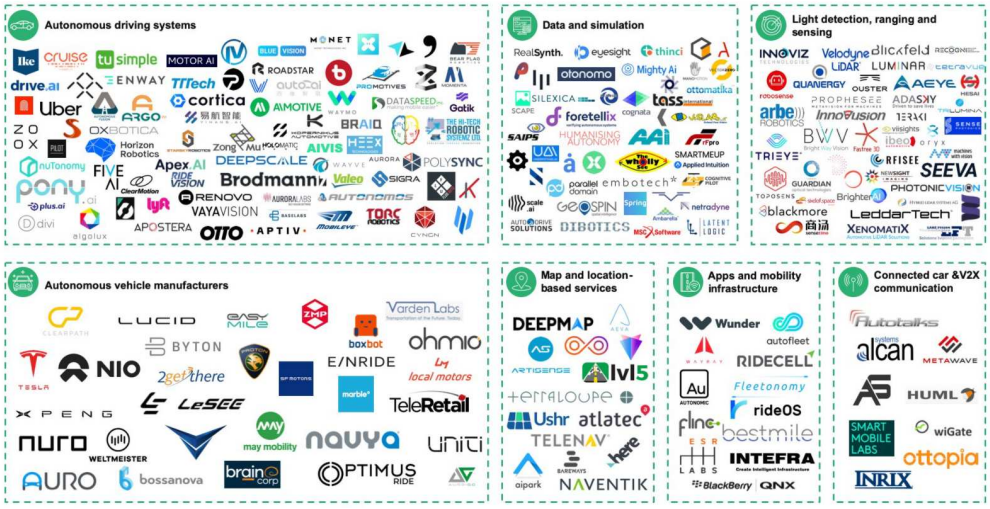


FIGURE 1.3: **Autonomous vehicle market [6].**

Autonomous vehicles have been put into tests or even commercial uses with varying degrees of automation in big tech companies such as Waymo, Uber, Lyft and Tesla, as shown in Figure 1.3. Several projects in recent years are shown in Figure 1.4. However, these cars have not yet reached the level of performance in all circumstances and all weather conditions.

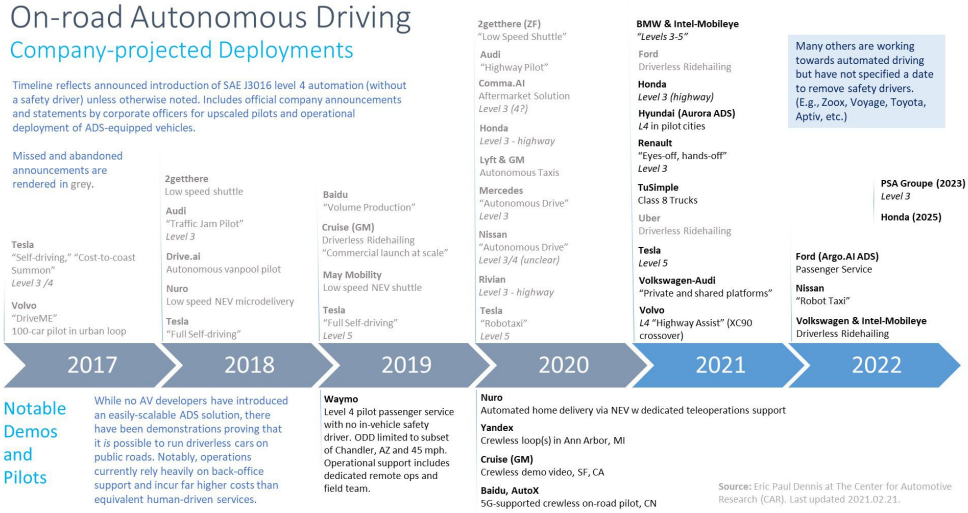


FIGURE 1.4: **On-road autonomous driving companies projected deployments [7].**

The different automation levels of an autonomous vehicle. We often talk about autonomous vehicles in their entirety, but different levels of autonomy exist. These different levels of autonomy were determined by the organization SAE International in 2021 committee [8], as shown in Figure 1.5. Level 0 is currently the best known, which is the traditional vehicle. The human, the driver, is solely responsible and alone performs all the maneuvers intended for driving the vehicle. From level 1, the responsibility is shared between the driver and the system.

Level 1 is represented by driver assistance. The latter is responsible for the vehicle and its driving while having driving aids, such as ABS, anti-lock wheel system, ESP, electronic trajectory corrector or even cruise control. In summary, the driver takes care of steering the vehicle, accelerating, braking, monitoring the road and is ready to intervene in the event of a problem. The system then assists the driver using the aids mentioned above.

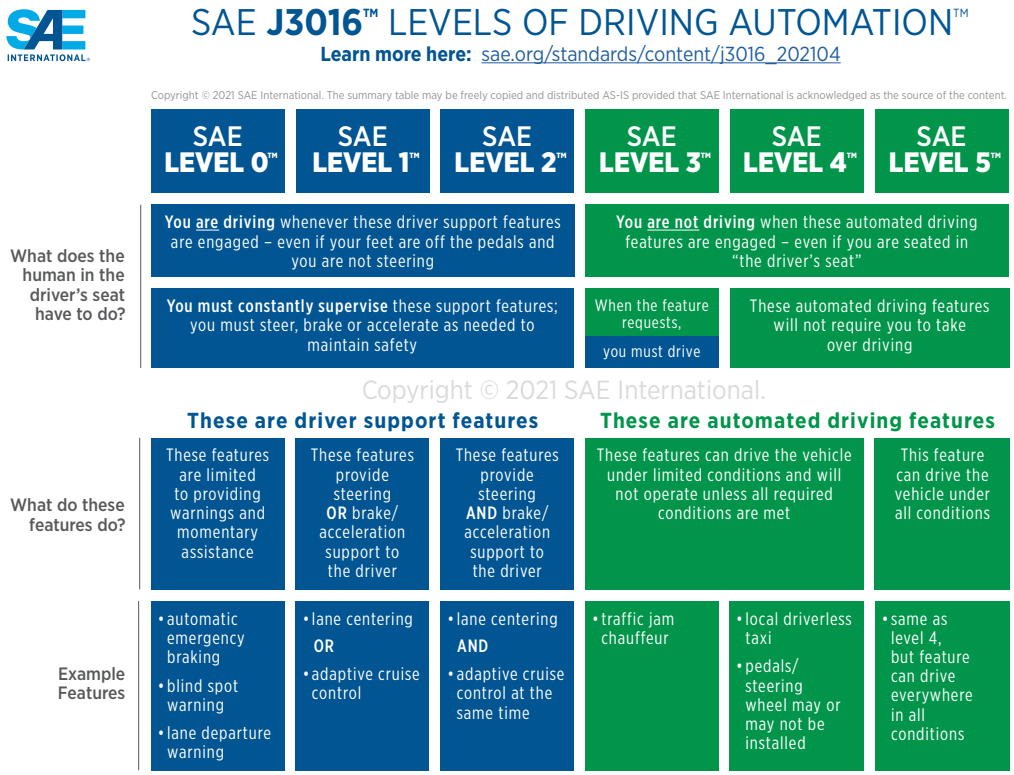


FIGURE 1.5: **The five levels of driving automation.** The visual chart was taken from SAE International in 2021 committee [8].

Level 2 takes over the driving aids of level 1 by adding an assistance to control the steering, but also the brake and the accelerator by the system. Indeed, the latter is able to redirect the trajectory of the driver and can also help to move from one line to another once the driver has activated the indicator. The latter therefore only has to manage the steering wheel, without any action on the pedals. In summary, the system takes care of accelerating, braking and steering the vehicle, the driver must monitor the road and regain control of the vehicle if necessary. Level 2 is the maximum level authorized to date by European legislation to drive on the road network.

As for level 2, level 3 takes up all the aids and rules of the previous level by adding certain specificities. The driver has the possibility of delegating total driving to the vehicle in certain situations, when traffic is dense or on the motorway for example. However, the driver must always remain attentive to his surroundings and be able to regain control of the vehicle at any time. In summary, the driver should only take control of the car in the event of a problem, while the system takes care of the pedals, steering, and road surveillance in other cases. This level is more commonly referred to as "Eyes off — Hands off", which means that the driver does not have to use their eyes or hands all the time.

Once level 4 has been reached, there is already a high degree of autonomy. Indeed, the driver can indulge in other activities such as reading a book or watching a film, leaving full control of his vehicle to move from point A to point B. Driving tests of this level of autonomy will first be tested on the highway in order to be better understood by, moreover, mild weather. In short, the driver does nothing and the system looks after everything (mind off).

Finally, the last level, level 5 represents the total autonomy of the vehicle (human off). The driver is non-existent, the human will be a passenger and can engage in any other activity, and the vehicle will circulate completely autonomously on the entire road network and in any weather. It is even very likely that the level 5 autonomous vehicles will be able to do without a steering wheel and pedals.

Autonomous vehicle sensors. To fully understand what a self-driving vehicle is, it is important to perform a technical analysis to understand how it actually works. Various elements of technology are brought into play here.

For an autonomous vehicle to be able to circulate completely safely, it is necessary to provide it with elements of technology that guide it. In order to recognize the environment, the vehicle is equipped with various sensors. These sensors collect and generate data in bulk. These useful data and perception algorithms will make it possible to determine how dangerous a situation is, for example, the presence of an obstacle.

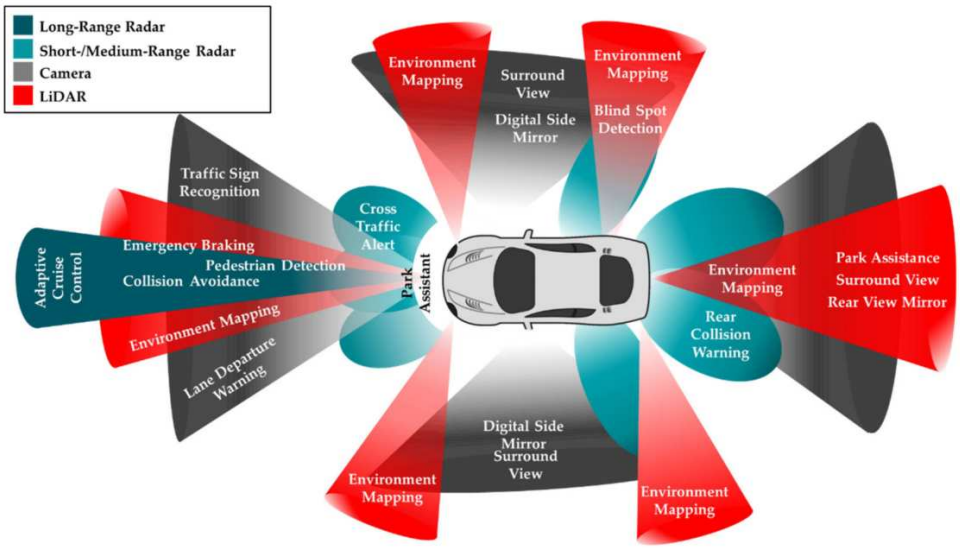


FIGURE 1.6: **Autonomous vehicle sensors [9, Fig. 3].**

So, the first category covered here is hardware. In general, the technological equipment will make it possible to send all the necessary information, whether it concerns

the vehicle itself, its environment, or even the other actors present in the vehicle. This will allow anticipating any possible risks that may be present, the vehicle will therefore be able to act accordingly and maintain the highest possible level of safety. To make all this possible, various sensors come into play, as shown in Figure 1.6, such as stereo cameras, infrared cameras, RADAR, SONAR, LiDAR, electronic stability controls or even GPS, speedometers and odometers.

In the framework of this research, and for the hardware part of the system to develop, we will focus mainly on cameras and LiDAR because these are the most commonly used sensors with regard to autonomous vehicles. From the literature, we can see that the combination of fusion-based 3D object detection has been extensively investigated due to its significance for many real-world applications. These choices and justifications are coming from a deep analysis of the state-of-the-art (SOTA) (see Chapter 2).

The stereo camera represents the driver's eyes on a human scale. This camera is made up of several lenses that allow the vehicle to represent its environment in the smallest details, and therefore to know how and where to move. With two sensors, it is possible to constitute a 3D environment in front of the vehicle to detect possible obstacles and predict their distance from the car. The LiDAR will complete the vision and the information provided by the camera, by adding notions of size and distance to the different objects that make up the environment. Figure 1.7 provides a representation of data output from camera and LiDAR sensors.

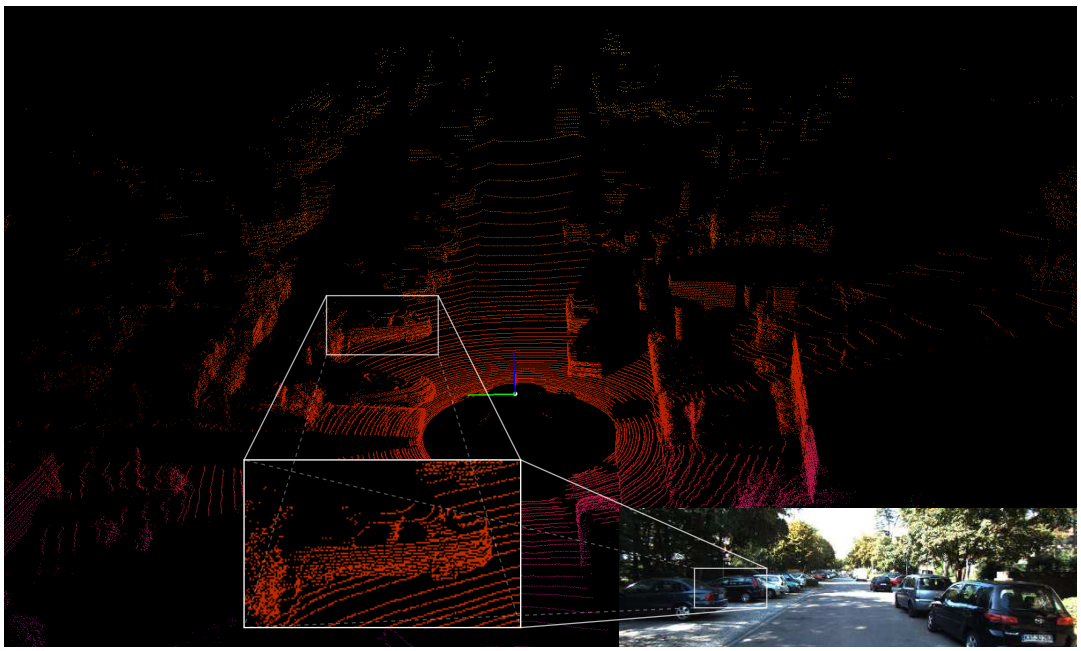


FIGURE 1.7: **Example of camera and LiDAR data.**

Hardware (hardware components) is not the only thing being investigated. It is also important to consider the software (algorithm). Input from the sensors is then used as input to the algorithm, which takes any necessary action, such as predicting, accelerating, and turning.

Environmental awareness is crucial to the development of software. This allows the vehicle to identify the entities surrounding it, analyze them, and decide whether there is a risk to the vehicle. As closely as possible, these sensors, and software, will reproduce human actions without intervention.

1.2 Problem Statement and Scope of Study

In recent years, 3D object detection has gained increasing attention in both academia and industry due to its numerous applications in fields such as autonomous driving. Accurate detection and localization of objects are crucial for the perception of self-driving cars. With the advancement of deep learning and convolutional neural networks (CNNs), the 2D object detection technique has made significant progress in efficient network architectures. However, determining the distance from the ego-vehicle to objects is the most challenging aspect of 3D object detection. Expensive technology, such as LiDAR, can provide precise and accurate depth information, leading to a performance gap between LiDAR-based methods and camera-based methods. Although many authors have investigated how to fuse LiDAR with RGB cameras, limited studies have explored fusing LiDAR and stereo in a deep neural network for 3D object detection. In this thesis, we propose a new approach to fuse data from a stereo camera and 4-beam LiDAR via a neural network for depth estimation, achieving better dense depth maps and improving 3D object detection performance. This approach is also classified as a low-cost sensors-based method since 4-beam LiDAR is cheaper than the well-known 64-beam LiDAR.

Several factors can affect the perception ability of self-driving cars, potentially causing serious consequences for other road users. Environmental influences can distort the data collected from these sensors, affecting the awareness of self-driving cars. While sensors perform well in controlled lighting or indoor environments, outdoor applications face numerous challenges, particularly in extreme lighting conditions such as sunburns, low light, and nighttime conditions. Self-driving cars also face significant challenges under adverse weather conditions such as fog, rain, and snow, severely affecting both the camera and LiDAR. Therefore, this work focuses on 3D object detection with camera and LiDAR sensors operating in foggy weather conditions.

This thesis includes the following objectives:

- To identify the current state of deep learning-based approaches for 3D object detection, including the most commonly used deep architectures for learning

3D object features. We will discuss their advantages and limitations and how they can address 3D object detection challenges for autonomous vehicles.

- To investigate and propose new 3D deep learning architectures for 3D object detection based on stereo camera and LiDAR. The proposed approach should be able to detect objects and ensure high accuracy.
- To collect or produce a dataset for evaluating our proposed 3D object detection approach under adverse weather conditions such as fog.
- To investigate and propose improvements for 3D object detection for self-driving cars in foggy weather conditions.
- To analyze the role of each sensor in the model.

The following research questions will guide the study:

- How can we effectively fuse image and point cloud into a single model?
- How are cameras and LiDAR affected by fog?
- Regarding the affected sensors, how does the model's performance change?
- Which data is worth using in the model?

1.3 Main Contributions

With the research objectives mentioned above, our contributions in this thesis can be summarized as follows:

- We propose a novel deep learning-based architecture called Sparse LiDAR and Stereo Fusion (SLS-Fusion) for 3D object detection. This network takes both stereo camera and LiDAR inputs.
- We introduce a novel dataset called Multifog KITTI, which includes both stereo images and point cloud data. This dataset is augmented for foggy scenes based on the KITTI dataset.
- We analyze the contribution of each sensor, camera, and LiDAR to the proposed SLS-Fusion model under favorable and foggy weather conditions.
- We conduct experiments on the KITTI dataset and demonstrate that SLS-Fusion outperforms low-cost-based car detection methods.
- We perform experiments on the Multifog KITTI dataset, propose improvements, and demonstrate that SLS-Fusion can achieve better performance in foggy scenes.
- We analyze different combinations of cameras and LiDAR types to demonstrate the trade-off between cost and performance of 3D object detection for autonomous vehicles.

1.4 Thesis Outline

In this chapter, we provide the context for this study, including the identification of research objectives and questions, and the argument for the value of this research. The remaining chapters in this thesis are outlined as follows:

Chapter 2 presents a review of the state-of-the-art in object detection for autonomous vehicles, considering both favorable and foggy weather conditions. The chapter discusses the methods found in the literature, including the combination of sensors and processing algorithms used. This discussion helped inform our choices for our proposed system.

Chapter 3 describes the dataset used in this thesis and the process of augmenting the original KITTI dataset with fog to produce the Multifog KITTI dataset. As our objective is to develop algorithms for detecting 3D objects in foggy weather conditions, this chapter provides important context for our proposed approach.

Chapter 4 presents our proposed approach, SLS-Fusion, which uses a neural network to fuse data from a 4-beam LiDAR and a stereo camera for depth estimation, to improve 3D object detection performance. This approach is also classified as a low-cost sensors-based method, as 4-beam LiDAR is less expensive than the more commonly used 64-beam LiDAR.

Chapter 5 analyzes the effects of fog on our model's performance, based on experiments conducted on the Multifog KITTI dataset presented in Chapter 3. This chapter also proposes methods for improving model performance under foggy conditions.

Chapter 6 provides an ablation study that examines the roles of cameras and LiDAR in our object detection model. We test different combinations of cameras and several types of LiDAR with varying numbers of beams (e.g., 4 beams, 8 beams, 16 beams, 64 beams) to analyze the complementarity between these two sensor types. Additionally, a brief economic analysis is carried out.

Chapter 7 concludes the thesis by reviewing the results and highlighting the main findings, lessons learned, and short-term perspectives for future research.

Chapter 2

Background and Related Work

Contents

Chapter overview: Object detection is a crucial element of computer vision that enables us to detect and classify objects in digital images. In recent years, there has been a surge of interest in 3D object detection for autonomous driving. Detecting and localizing objects in 3D space is more challenging than in 2D. Additionally, adverse weather conditions pose a significant challenge for self-driving cars. Sensor installation is an essential aspect of collecting necessary meteorological data, which is then included in the object detection model. This chapter offers a comprehensive review of 2D object detection and the current state-of-the-art (SOTA) of deep learning-based 3D object detection for self-driving cars in Section 2.3. We classify the methods according to the type of input data they take as input to the model: camera-based, LiDAR-based, and fusion-based 3D object detection. Finally, we provide some observations and conclusions in Section 2.5.

2.1 Introduction

The automation of driving is based on many aspects, such as perception, positioning, scenario analysis, decision-making, command control. The subject of this thesis is about the perception part only. More specifically, it aims at taking into account

data fusion in order to improve detection levels, especially in degraded weather conditions. The data come from perception sensors on board of vehicles. LiDARs, radars and cameras are commonly used. We will focus in the framework of the thesis on the joint use of cameras and LiDARs, because these are the sensors that best allow the detection of vulnerable road users and that these are the sensors most impacted by degraded weather conditions. It is therefore on this combination of sensors that our contribution can be relevant. A SOTA on 2D detection methods will be presented in the Section 2.2. Then in Section 2.3, more recent methods of detection and localization methods (3D detection) will be presented. Finally, in Section 2.4 we will focus on the problem of degraded weather conditions, and we will show that these are still too little addressed in the literature.

2.2 Traditional-based and Deep Learning-based 2D Object Detection Methods

A 2D object detector produces, for each object of interest in an image, a 2D bounding box and a class label. In two dimensions, the bounding box is defined by an axis-aligned rectangle, which should be as small as possible while still containing all parts of the associated object in the image, as shown in FIGURE 2.1. There are many ways to encode a bounding box. For example, in the PASCAL Visual Object Classes (VOC) Challenge [10], a 2D bounding box is parameterized as $(x_{tl}, y_{tl}, x_{br}, y_{br})$, where (x_{tl}, y_{tl}) represents the pixels in the top-left corner and (x_{br}, y_{br}) represents the bottom-right corner of the object in the image. On the other hand, in the challenging COCO benchmark [11], it is encoded as (x, y, w, h), where (x, y) represents the pixels in the top-left corner and (w, h) represents the size of the object in the image.

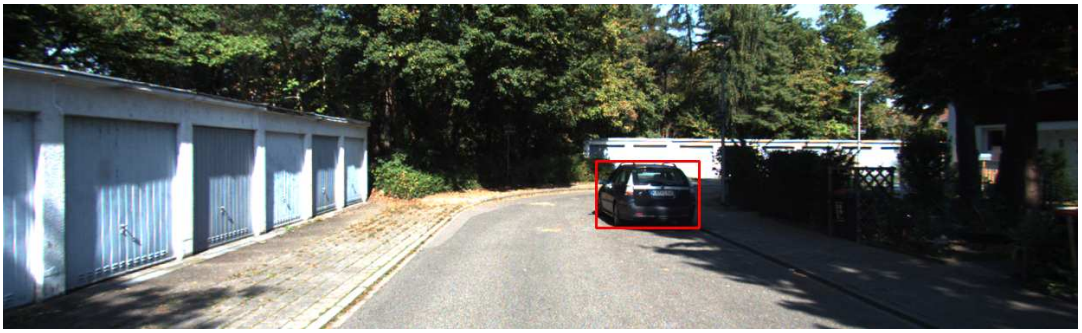


FIGURE 2.1: **Ground truth annotated 2D bounding box from a scene in KITTI dataset [12].** The red bounding box corresponds to the car object class.

Over the past 20 years, there has been a significant evolution in object detection technologies. In the past, researchers used classical methods for detecting objects [13, 14] before deep learning became widely known. Due to this, 2D object detection can be divided into two categories: traditional-based and deep learning-based methods.

Traditional-based methods usually use hand-crafted features serving for object recognition/ detection. For this reason, these methods often get stuck in complex scenarios. In the domain of face detection, by using sliding windows Viola and Jones [13] looks at all possible positions and scales in an image to determine if any location contains a human face. Haar-like features are extracted and then fed into the classifier to determine if it is a face or not. Dalal and Triggs [14] solve the pedestrian detection problem by adopting the Histogram of Oriented Gradients (HOG) descriptor to extract features and then using a linear SVM to determine if it is a person. In this thesis, our works are not based on this family of methods because these methods are now overtaken by the methods based on deep learning which are presented below.

In contrast to the traditional-based method, recent deep learning-based architectures [15–26] leverage GPU power and massive amounts of data to learn and then extract these learned feature representations which is usually optimized for a particular task, leading to a significant improvement in the model performance. However, it requires a large number of annotated images (for supervised learning) and GPU infrastructure, which is expensive. Deep learning-based object detection methods can generally be classified into two main categories: two-stage object detectors [21–26] where a region proposal network (RPN) first predicts the proposed objects called the region of interests (RoI) and these ROIs are refined later to obtain the final bounding boxes, and one-stage object detectors [15–20, 27] in which the objects are predicted directly from the image. Girshick et al. [21] proposed R-CNN to investigate how to apply Convolutional Neural Network (CNN) from classification to object detection problem. As shown in Figure 2.2, this method employs selective search proposed by Uijlings et al. [28] to find about 2000 region proposals: category-independent regions for each input image, leading to an extremely slow detection speed. Each region is affinely warped into a fixed input size that can be fed into a CNN to extract useful features from these regions. Support Vector Machine (SVM) is then used for each category to classify each region.

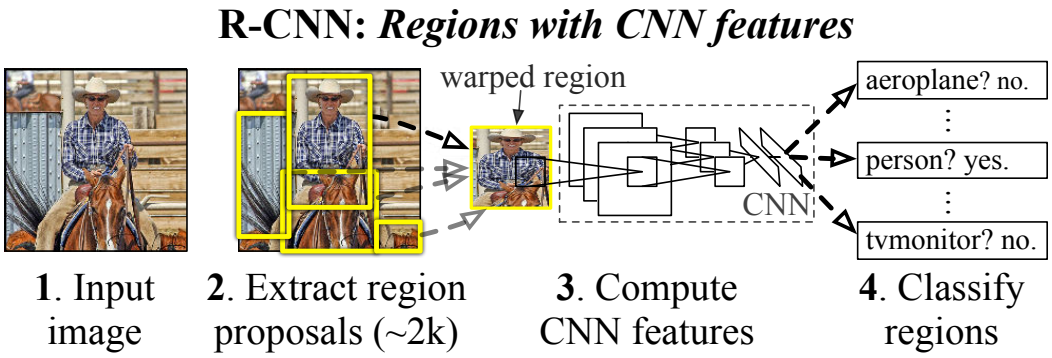


FIGURE 2.2: **Architecture of R-CNN method proposed by Girshick et al. [21, Fig. 1].**

Girshick [23] proposed Fast R-CNN which simplifies the R-CNN architecture while improving the performance and accelerating training/ inference time. Its architecture is shown in Figure 2.3. Computation sharing between regions speeds up training. The procedure results in end-to-end training with the ability to fine-tune all layers. It extracts a feature map from an image of any size. For each region, a ROI pooling layer pulls a set length feature from the feature map (suggested by selective search [28]). Each feature is fed through a sequence of Fully Connected (FC) layers, which branch into two output layers: one for softmax probability (classification) and the other one for the four bounding box regressor parameters. Despite the fact that Girshick [23] successfully integrates the benefits of the work of Girshick et al. [21] and He et al. [22], its detection speed is still restricted by proposal detection.

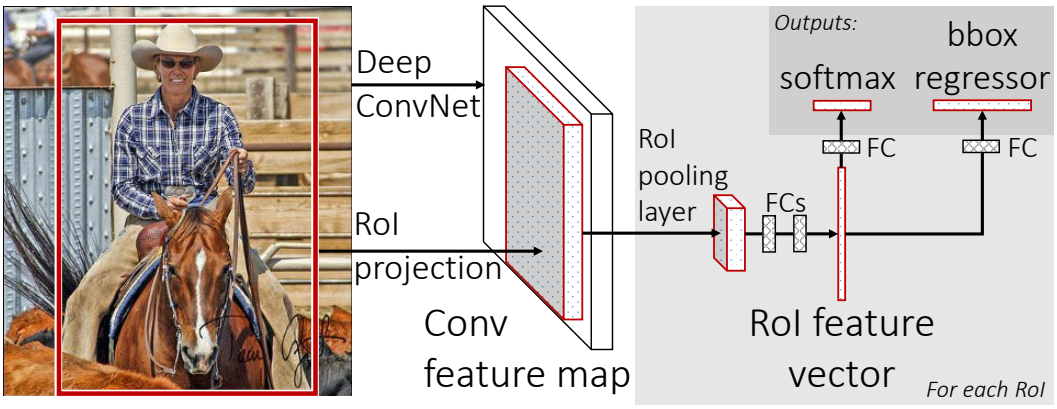


FIGURE 2.3: **Architecture of Fast R-CNN method proposed by Girshick [23, Fig. 1].**

Region proposal became a real-time detection bottleneck (Uijlings et al. [28] takes almost the same time as detection). Shortly after the work of Girshick [23] and Ren et al. [24] proposed the Faster R-CNN detector in 2015, shown in Figure 2.4. It is the first end-to-end deep learning object detector, as well as the first deep learning object detector that works in near-real time. The RPN has been trained to handle region proposals. Fast R-CNN slightly improves average accuracy (mAP) compared to R-CNN. It significantly improves training (8.75 hours and 84 hours) and detection time of R-CNN. Even though Faster R-CNN overcomes the speed bottleneck of Fast R-CNN, there is still computation redundancy at the detection stage.

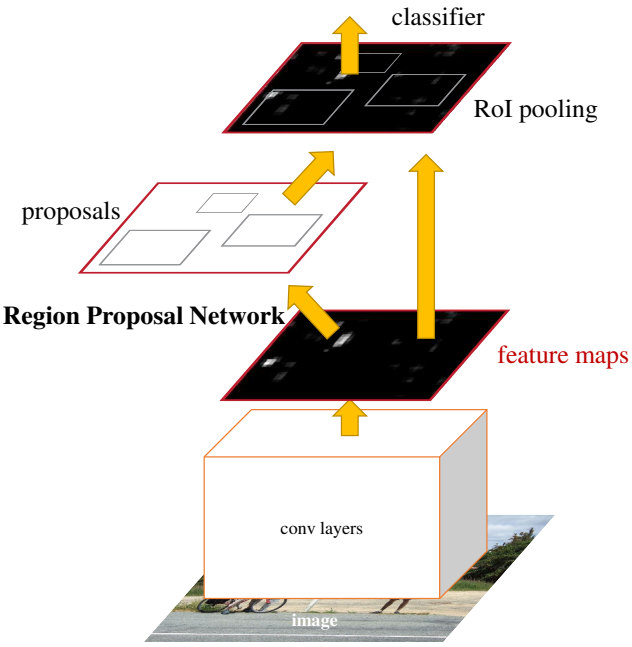


FIGURE 2.4: **Architecture of Faster R-CNN method proposed by Ren et al. [24, Fig. 2].**

To localize the object within the image, all the prior object detection techniques used regions detectors before classification (two-stage detectors). With YOLO proposed by [15], the network no longer examines the entire image. Rather, it examines portions of it that have a high likelihood of holding the object. Object detection is then treated as an end-to-end regression problem. In YOLO method, a single CNN predicts multiple bounding boxes and class probabilities for those boxes. It also predicts all bounding boxes for an image across all classes at the same time. It generates a S $\times$ S grid from the input image. If an object's center falls inside a grid cell, that grid cell is in charge of detecting that object. For each grid cell, B bounding boxes, the corresponding confidence score, and C class probabilities are predicted. These

confidence scores represent the model confidence that the box includes an object, as well as how accurate it estimates the box it predicted is.

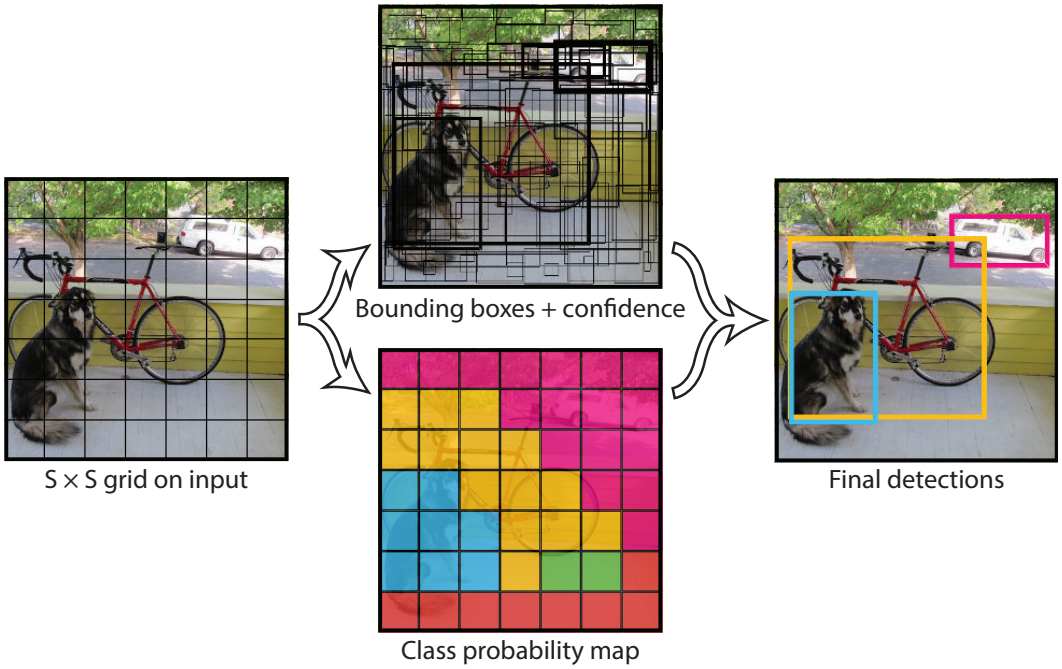


FIGURE 2.5: **Architecture of YOLO method proposed by Redmon et al. [15, Fig. 2].**

In YOLO [15], each grid cell only predicts one object, as shown in Figure 2.5. As a result, the method significant weakness is that it fails when the center of multiple objects falls into the same cell. The problem was solved using anchor boxes in [16, 17, 24] and YOLO variants [18, 19, 27].Instead of directly regressing a bounding box, these methods predict off-sets from a pre-determined set of boxes, called anchor boxes, with particular height-width ratios. These anchor boxes is predefined by clustering bounding boxes on ground truth labels. Instead of directly predicting bounding boxes, regressing the offset to anchor boxes allows the network to learn easier. However, it makes the model highly dependent on the training dataset because it uses a k-means clustering to find the optimal size of anchor boxes. Besides, anchor-based methods often increase the number of false positive predictions. So it is normally equipped during inference step with a post-processing step, which is Non-Maximum Suppression (NMS) to retain only one box for each object. Moreover, rather than treating the same regardless of how large or small the object is, later methods [16–19] have a multiscale design which helps the model to make better predictions when objects of different sizes appear in the image. Using anchors includes a lot of hyperparameters, such as the number of anchors and anchor size, all of which have an impact on the end outcome, even if the changes are minor.

Thanks to the explosion of deep learning, 2D object detection is an extensive and very attractive research topic in the field of computer vision. However, the previously discussed 2D object detectors can be difficult to adapt to detect 3D objects. Only 2D bounding boxes can be displayed, and depth information is not presented, which is crucial for safe driving.

2.3 3D Object Detection Methods

In the autonomous driving field, 3D object detection is more relevant than 2D object detection since it provides more spatial information, such as location, direction, and size, as shown in FIGURE 2.6. According to Qi et al. [29], a 3D bounding box is encoded as a set of seven parameters $(x, y, z, h, w, l, \theta)$, including the coordinate of the object center (x, y, z), the size of the object (height, width, and length) and its heading angle (θ). Besides the development of deep learning, hardware technology has also made strides, with the appearance of LiDAR in autonomous driving (The 2005 DARPA Grand Challenge [30]) next to cameras and RADAR. It makes this task more developed and competitive through its impressive results by using point cloud data.

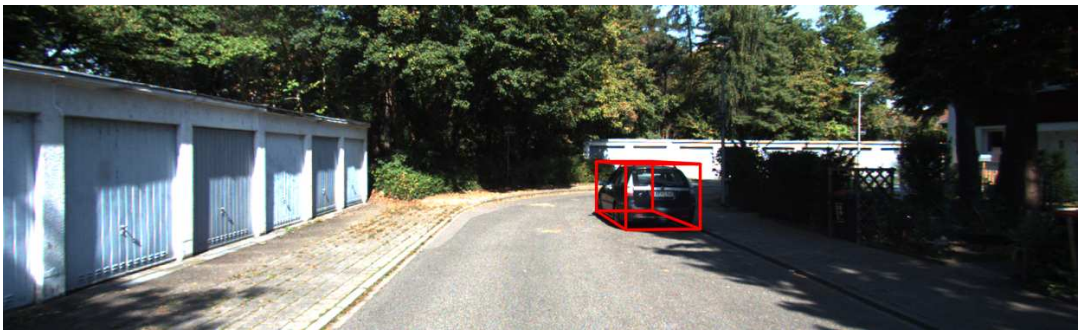
This thesis focuses on the fusion of camera and LiDAR. Based on these sensors, 3D object detection has typically approached in the following three directions: camera-based (including monocular or binocular) (presented in Section 2.3.1), LiDAR-based (presented in Section 2.3.2) and fusion-based (presented in Section 2.3.3).

2.3.1 Camera-based 3D Object Detection Methods

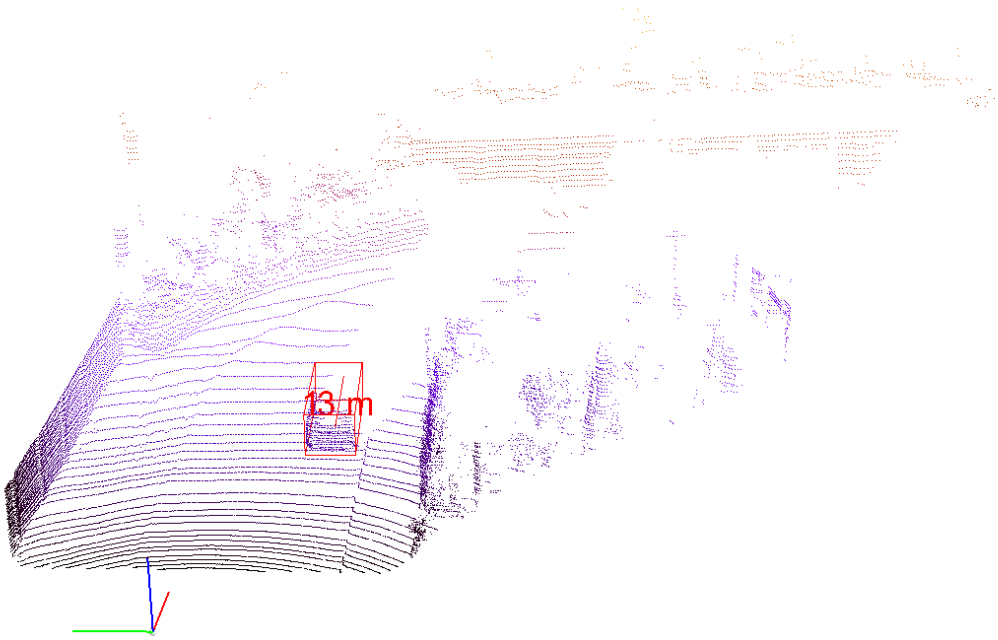
3D object detection based on a 2D image is a challenging task. This task is often subdivided into: monocular-based and binocular-based methods. Although the image provides rich shape and texture details, it lacks depth information, which is critical for estimating the size and location of objects (depth information from the stereo camera is not accurate enough). This information is the mainstay of 3D object detection for autonomous driving. So instead, camera-based methods usually try to detect objects by adding inductive biases into their model such as the 2D/ 3D constraints as well as predict keypoints/ shapes of objects on image to form 3D bounding boxes. More modernly, by first predicting the depth map and then using this predicted depth information, the 2D image space is converted to 3D points space to take advantage of the efficiency of LiDAR-based methods.

Keypoints/ Shapes-based methods – [31–36]

In 2017, Chabot et al. [31] presented Deep MANTA, one of the first works on monocular 3D object detection. Its architecture is shown in Figure 2.7. It recognizes 2D keypoints and uses 2D/ 3D matching to find the best position. The procedure is divided into two steps. The method firstly uses a cascaded RCNN architecture to



(a)



(b)

FIGURE 2.6: **Ground truth annotated 3D bounding box from a scene in KITTI dataset [12]** respectively showed on (a) the RGB image (camera); (b) point cloud (LiDAR). The red bounding box corresponds to the car object class.

detect and refine 2D bounding boxes, corresponding classification, 2D keypoints (36 vehicle part pixels coordinates including visible, occluded and self-occluded parts), and template similarity. This data is then passed to a second stage, which uses the predicted template similarity to pick the best matching 3D CAD model in the 3D CAD vehicle dataset (103 CAD models with a weakly annotated process for 36 keypoints) in order to recover the 3D location and orientation. For 3D localization, a 3D bounding box is deemed correct if the distance between its center and the center of the ground truth bounding box is less than 1 meter.

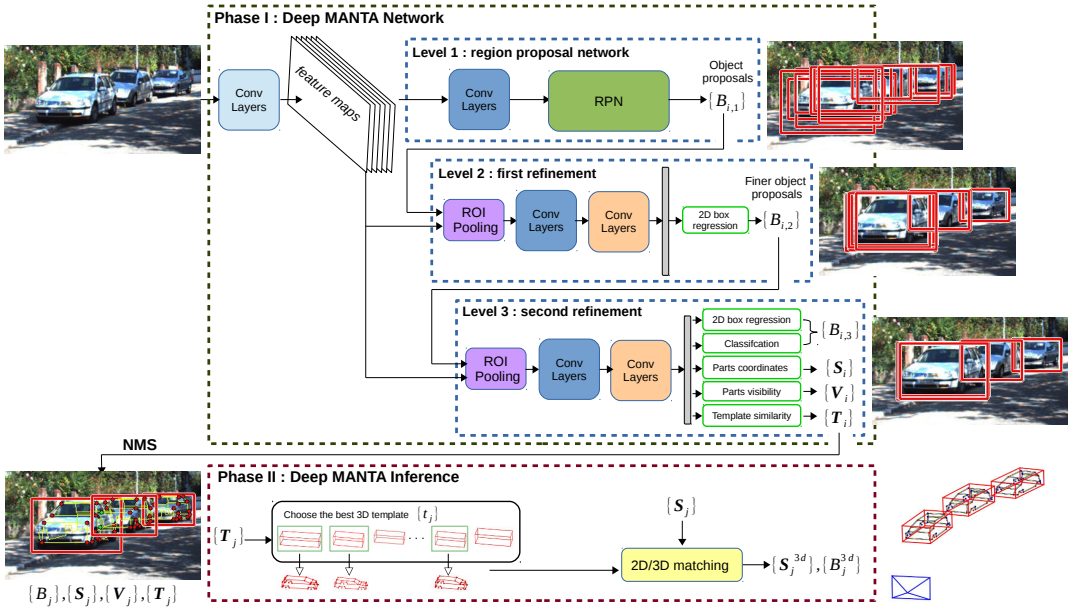
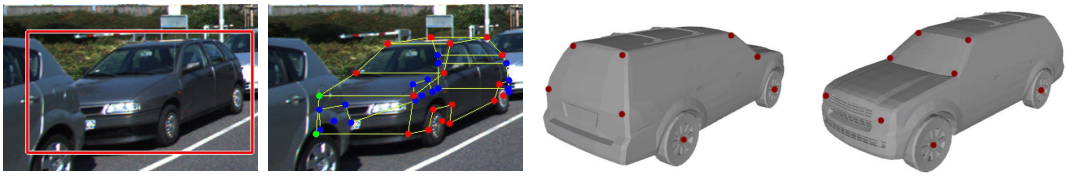


FIGURE 2.7: **Architecture of Deep MANTA method proposed by Chabot et al. [31, Fig. 2].**

The 2D/3D matching is time-consuming. As a result, reducing the complex quantity of points while keeping competitive performance is worth considering. Barabanau et al. [36] greatly reduces this by using only 5 CAD models and 14 keypoints. Figure 2.8 shows (a) 36 keypoints and (b) 14 keypoints defined from Deep MANTA [31] and MonoGRNet 2 [36].



(a) 36 keypoints (b) 14 keypoints

FIGURE 2.8: **3D keypoints defined** from (a) Deep MANTA proposed by Chabot et al. [31, Fig. 4(a, b)]; (b) MonoGRNet 2 proposed by Barabanau et al. [36, Fig. 3].

RTM3D method proposed by Li et al. [33] presented a novel approach for detecting scale-invariant keypoints in point-wise space called the Keypoint Feature Pyramid Network (KFPN). Its architecture is shown in Figure 2.9. The method detection head is inspired by Duan et al. [37]. The method predicts 2D projection keypoints of all 8 cuboid vertices and cuboid center. It also directly regresses the 3D dimension, orientation, distance, and 2D size of objects. As the predicted keypoints are relatively

noisy, rather than directly forming a cuboid with them, these values are used as initial estimates for optimization using perspective projection geometric constraints to create a 3D bounding box.

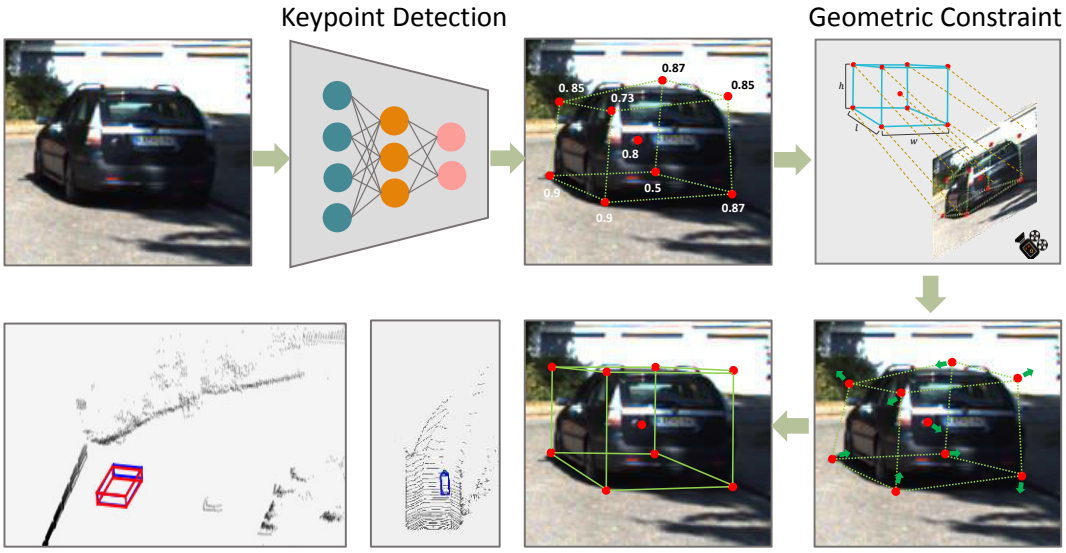


FIGURE 2.9: **Architecture of RTM3D method proposed by Li et al. [33, Fig. 1].**

2D/ 3D consistency constraints-based methods – [38–41]

Mousavian et al. [38] proposed a simple monocular image-based architecture. A SOTA of 2D object detector is employed in this approach to generate 2D region proposals, from which related 3D proposals are calculated, and then the local point cloud of dynamic objects is reconstructed. Each region proposal that meets a specific level of confidence is input into a CNN, which generates estimates for the corresponding 3D bounding box dimensions (h, w, l) and heading angle θ. For the heading estimation, a novel classification-regression hybrid is used, in which the angle is divided into discrete bins and residuals for each angle bin center are regressed. The center coordinates (x, y, z) are calculated using the estimated dimensions and heading angle, as well as the constraint that a projected 3D bounding box should fit closely into the corresponding 2D bounding box. An optimization-based method can then be used to estimate the entire 3D bounding box.

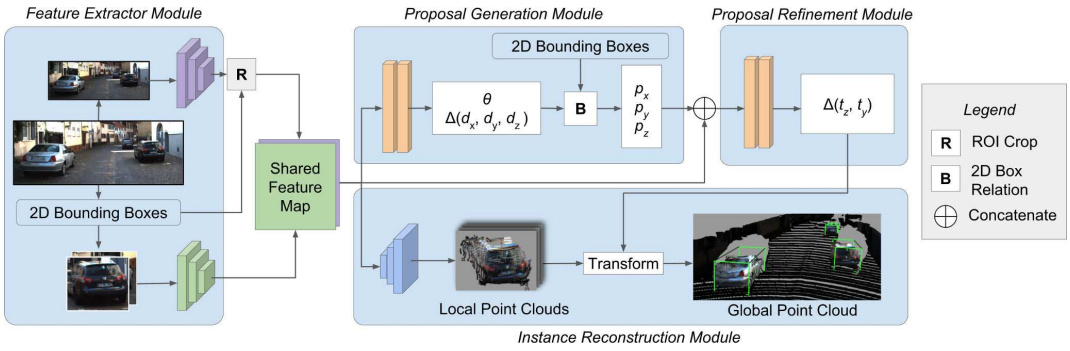


FIGURE 2.10: **Architecture of MonoPSR method proposed by Ku, Pon, and Waslander [41, Fig. 2].**

Pseudo point cloud-based methods – These methods typically have data conversion from 2D to 3D by using extrinsic calibration information between the camera and LiDAR.

In MLF method presented by Xu and Chen [42], a depth map is first predicted from monocular RGB and concatenated as RGB-D to form a tensor of six channels (RGB image, Z-depth, height, distance), and then it is utilized to regress the 3D bounding boxes. This is one of the first papers to suggest that the estimated depth information be raised to a 3D point cloud. The term "Point Cloud", as seen in Figure 2.11, is converted from the estimated depth but is only used to visualize the detection results. An interesting paper titled Pseudo-LiDAR presented by Wang et al. [43] was inspired by this article, based on this idea of this data representation transformation.

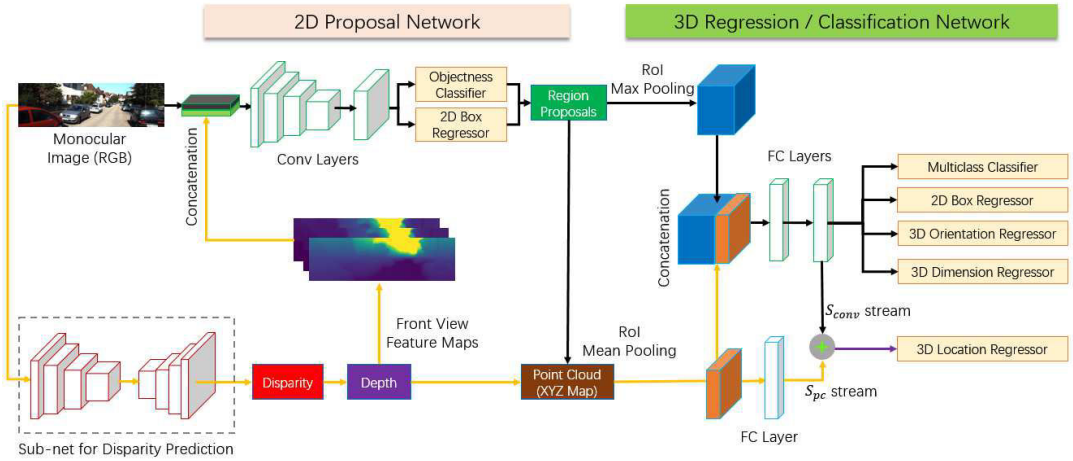


FIGURE 2.11: **Architecture of MLF method proposed by Xu and Chen [42, Fig. 1].**

Wang et al. [43] pioneered a pseudo-LiDAR approach for camera-based object detection. Its architecture is shown in Figure 2.12. It emphasizes the inefficiency of current camera-based object detection methods. This method try to bridge the gap between camera-based and LiDAR-based methods by representing the data as point clouds. It estimates the depth map from an RGB image using a depth estimation neural network. It then converts the predicted depth map into a pseudo 3D point cloud by projecting all pixels with depth information into LiDAR coordinates. And finally, they can process the pseudo point cloud as real point cloud to leverage of the efficiency of LiDAR-based object detection methods. This paper employs AVOD [44] and F-PointNet [29] for its 3D object detection part. The ability to reconstruct 3D point clouds (normally collected from expensive LiDAR) from less expensive monocular or stereo cameras is a valuable feature of this approach. It can take advantage of point cloud representations such as distance invariance, no ambiguity, and occluded objects. However, pseudo point clouds have some limitations compared to real point clouds. For example, although it has a much higher resolution than a LiDAR point cloud, its quality is determined by the depth estimation algorithm and it is computationally more expensive to process.

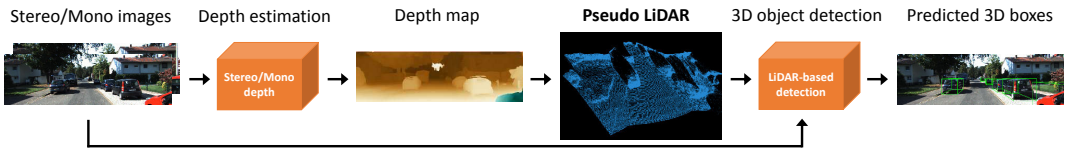
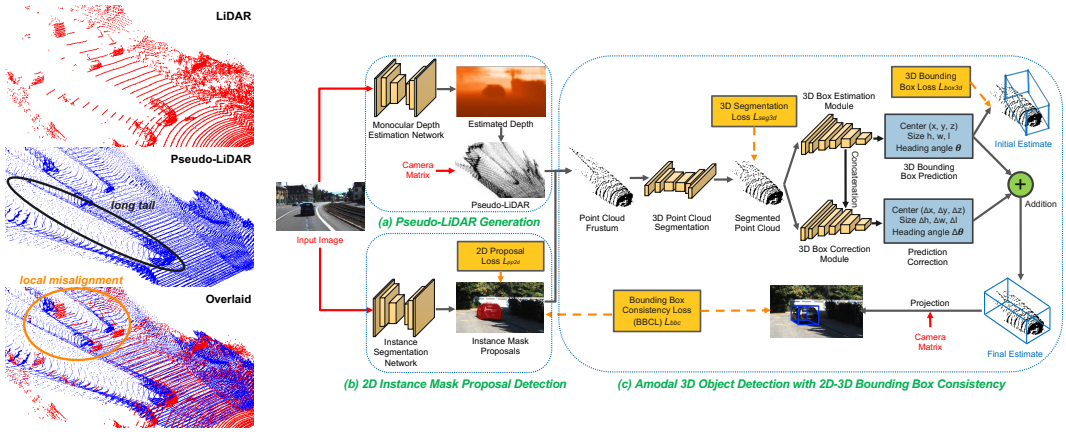


FIGURE 2.12: **Architecture of Pseudo-LiDAR method proposed by Wang et al. [43, Fig. 2].**

As demonstrated in Figure 2.13, Weng and Kitani [45] presented an end-to-end pseudo-LiDAR training routine. Pseudo-LiDAR [43] and this work have a similar concept. However, instead of training two separate networks (depth estimation and object detection as in [43], Weng and Kitani [45] designed the architecture in an end-to-end training fashion to better train the 3D detection task. The major issue with pseudo point clouds is noise such as depth errors, long tails in the 3D point cloud reprojected from the predicted depth map caused by fuzzy boundaries, as shown in Figure 2.13 (a). To attack this issue, Weng and Kitani [45] used the consistency of 2D and 3D bounding boxes and instance masks to improve the accuracy of the depth map.

Based on the work of Pseudo-LiDAR [43], the same authors proposed Pseudo-LiDAR++ [46] which suggests stereo depth estimation neural network, called SDN. In this network, it proposed a Depth Cost Volume to directly predict the depth map instead of predicting the disparity map (Disparity Cost Volume) as proposed by



(a) (b)

FIGURE 2.13: **Architecture of Pseudo-LiDAR end-to-end method proposed by Weng and Kitani [45, Fig. 2, 4].**

Chang and Chen [47] and this helps to boost the predicted depth map accuracy. In addition, to improve the accuracy of the predicted depth map, it proposes a depth correction phase by using a simulated 4-beam LiDAR to regularize the predicted depth map and since then the pseudo point cloud looks more real (see Figure 2.14). This belongs to the family of pseudo-LiDAR methods so we we mention it as camera-based method. However, it can also be classified in fusion-based methods 2.3.3 because it uses stereo camera as main and LiDAR as an option.

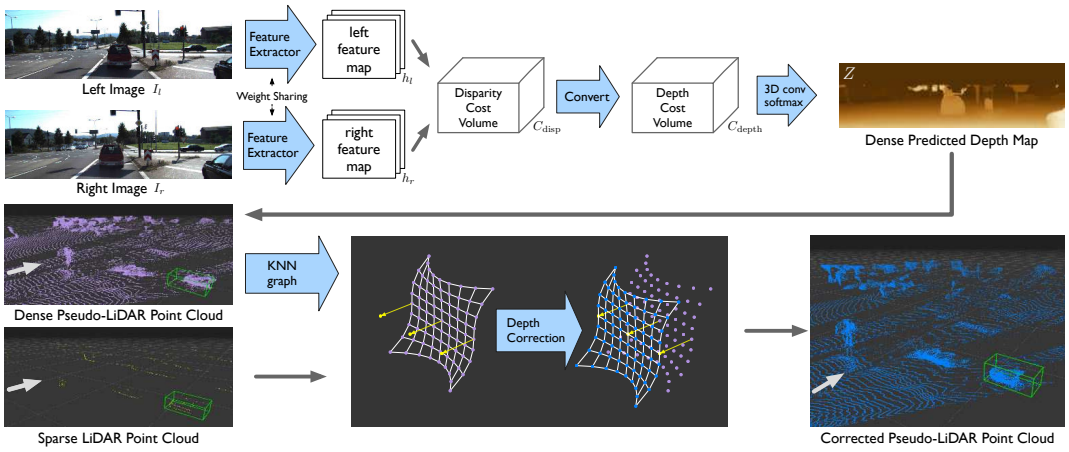


FIGURE 2.14: **Architecture of Pseudo-LiDAR++ method proposed by [46, Fig. 5].**

AM3D proposed by Ma et al. [48] improves on the concept of Pseudo-LiDAR by combining pseudo point cloud with RGB image information. Its architecture is shown in Figure 2.15. As a simple concatenation (x, y, z, r, g, b) is not effective, the

paper proposed an attention-based method for guiding message transmission between spatial and RGB features. The RGB fusion module is also effective for real point cloud. The method also proposed a simple point cloud segmentation design based on Qi et al. [29] work to improve the long-tail problem existing in pseudo point cloud.

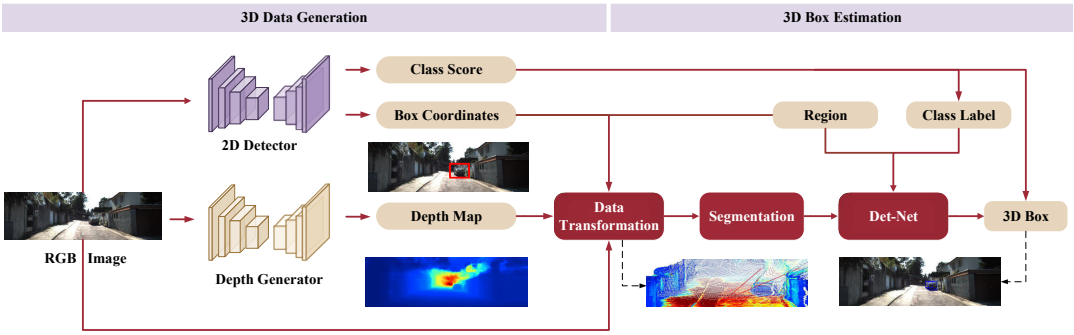
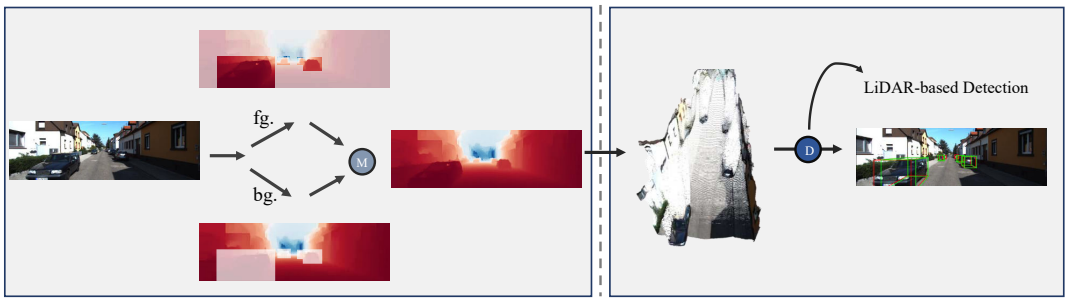


FIGURE 2.15: **Architecture of AM3D method proposed by Ma et al. [48, Fig. 2].**

Pseudo point cloud depends on the quality of the predicted depth map, which is normally inaccuracies and has blurry edges leading to edge bleeding. Wang et al. [49] proposed to train a depth estimation neural network in two separate parts: foreground and background. It focuses more on the foreground part by giving it a larger weight during training time. By the way it can focus on extracting information from the focused objects. Its architecture is shown in Figure 2.16.



(a) ForeSeE (b) 3D Object Detection

FIGURE 2.16: **Architecture of ForeSeE method proposed by Wang et al. [49, Fig. 4].**

Concluding this section, we see that image-based object detection is a traditional research area for a long time and moreover, cameras are cheap and provide a lot of texture information about objects based on color, edges, etc. However, the image

lacks depth information, which is extremely important for 3D tasks. Even when we use a stereo camera it still doesn't leave a sufficiently accurate depth map. From here, the research is gradually pushing to a sensor such as LiDAR that brings better information accuracy to better match the required task. This will therefore be presented in the following section.

2.3.2 LiDAR-based 3D Object Detection Methods

The point cloud reconstructs the scene in three dimensions and contains a wealth of information about geometry, shape, and size. As a result, relevant features that improve detection performance can be extracted. However, due to their nature and ability to process, point clouds encounter major obstacles such as sparse objects especially at a distance, difficulty in glass surfaces, lack of color information.The most effective deep learning approaches for object detection require data to be organized in a structured tensor (e.g., images, videos), which is not the case for point clouds. Due to the irregular, unstructured, and unordered nature of point clouds [50], they are often handled in one of three ways: projecting point clouds to generate a regular pseudo image, subsampling point cloud cells called voxels, or encoding raw point clouds with sequence of multi-layer perceptron (MLP) proposed in [51, 52]. Existing research has looked into the following three ways depending on the representation of the point cloud: view-based, voxel-based, point-based, and hybrid point-voxel-based detection.

View-based detection – Methods in this category project a point cloud onto a 2D image space to obtain a regular structure as an initial stage. Generally, CNN is then used to take advantage of this information. The most common types of projection are bird's eye view (BEV) [53], front view (FV) [53], range view (RV) [54, 55], and spherical view (SV) [56].

Among these view-based methods, methods are most often converted to BEV form [57–60] as it can preserve depth, maintain object size consistency over range which is a very important information of the 3D point cloud. BirdNet proposed by Beltrán et al. [58] firstly projects point cloud into BEV. This paper detected 2D objects on BEV based on the work of Ren et al. [24]. And then it proposed some post-processing step to generate 3D bounding boxes. Its architecture is shown in Figure 2.17.

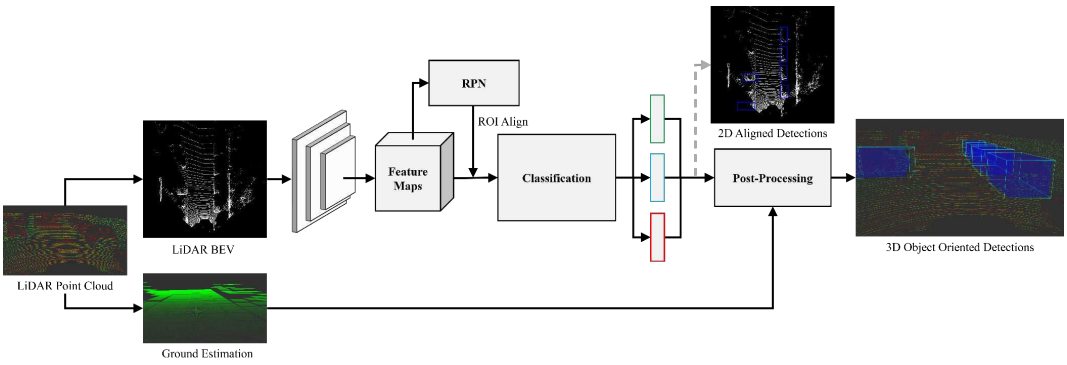


FIGURE 2.17: **Architecture of BirdNet method proposed by Beltrán et al. [58, Fig. 1].**

On another hand, several methods projects point cloud into range view (RV) image [54, 55]. To obtain a range view image, point cloud is roughly projected and discretized into a 2D point map. VeloFCN is the first study based on range image detection proposed by [54], which projects point clouds to 2D and uses 2D convolutions to densely forecast 3D bounding boxes. LaserNet [55] proposed a real-time probabilistic LiDAR that models aleatoric uncertainty. Its architecture is shown in Figure 2.18. As you can see from the RV view, point cloud is dense and is a native representation of LiDAR. Alternatively, if projected to 3D space or BEV, sparse representation occurs and more computation.

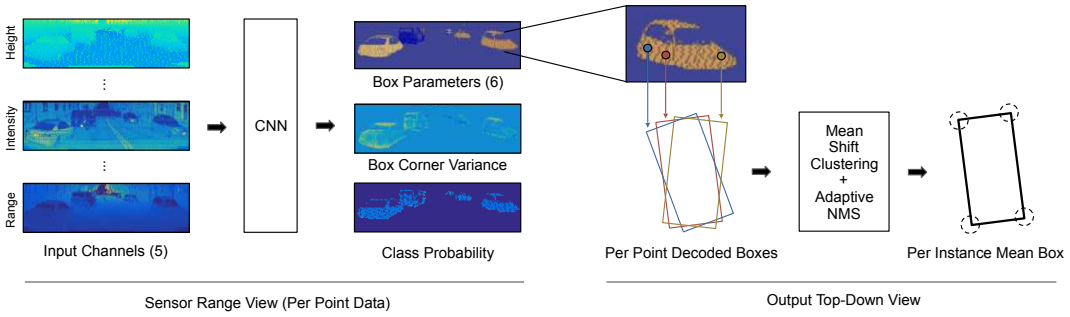


FIGURE 2.18: **Architecture of LaserNet method proposed by Meyer et al. [55, Fig. 2].**

Voxel-based methods – Methods in this category voxelize a point cloud into 3D grids (normally called voxels) as an initial stage.

In 2017, Vote3deep, a LiDAR-based detector, was presented by Engelcke et al. [61]. LiDAR point clouds are discretized into a sparse 3D grid using a hand-crafted feature vector calculated from the statistics of each cell based on the points within that cell. In order to process the discretized point cloud quickly, this sparse grid can be applied with sparse 3D convolutions, which limits filtering to the grid cells that do not contain empty arrays. Items are detected using a sliding-window search with a fixed-size window with N different orientations. On each window, a CNN performs binary classification, anticipating whether an object is included or not.

A similar approach was described in [62] for the detection of vehicles. LiDAR point clouds are discretized into three-dimensional grids and then processed through a fully convolutional network using 3D convolutions. In essence, this network consists of a 3D RPN that produces object confidence scores along with 3D bounding box residuals. This model outperforms the Vote3deep method [61].

In 2019, LiDAR points are divided into voxels that are evenly spaced, and they are grouped by the voxel they belong to. The proposed Voxel Feature Encoding (VFE) layer is then applied to each non-empty voxel point, resulting in a fixed-size voxel-wise feature vector. It is essentially a tiny PointNet [51]. It produces a sparse 4D array, which corresponds to a 3D grid in which only some voxels have learned feature vectors. A series of 3D convolutional layers are applied to the 4D array, which is used to create a 3D feature map. This feature map is passed into an RPN, which produces 3D anchor box residuals and object confidence scores.

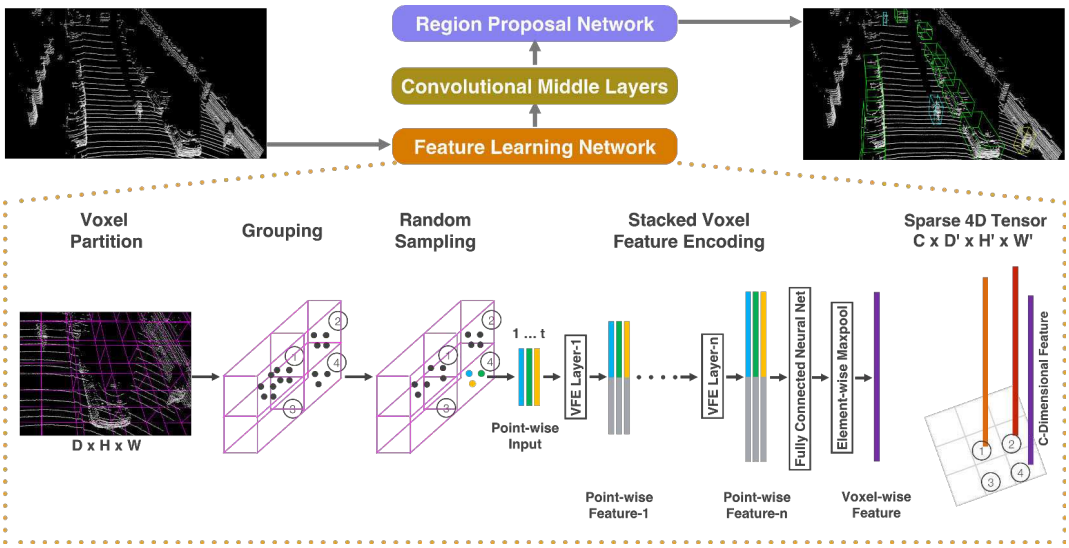


FIGURE 2.19: **Architecture of VoxelNet method proposed by Zhou and Tuzel [63, Fig. 2].**

Figure 2.19 shows the architecture architecture of VoxelNet [63]. In this work, Zhou

and Tuzel [63] completely removes the hand-crafted feature engineering for 3D point clouds such as BEV and proposed an end-to-end single stage 3D object detection. By encoding voxel feature encoding (VFE), VoxelNet divides a point cloud into evenly spaced 3D voxels, which are then transformed into unified feature representations. This encodes the point cloud as a descriptive volumetric representation, which then generates detections via a RPN.

Point-based methods – These methods usually deal with the raw point cloud directly [64–72] instead of converting the point cloud to a regular structure.

PointNet, a pioneering study on deep learning-based architectures for consuming point cloud raw data for classification and semantic segmentation, was introduced by Qi et al. [51]. PointNet, a pioneering study on deep learning-based architectures for consuming point cloud raw data for classification and semantic segmentation, was introduced by Qi et al. [51]. They point out that as the point cloud is unordered, the architecture should be permutation-invariant for all points. The PointNet, as shown in Figure 2.20, receives a point cloud P with n number of points as input and generates classification scores for k classes. The bottom branch handles segmentation and generates point-wise classification scores for m classes. The input point cloud is first passed through a spatial transformation network called T-net to generate an affine transformation matrix. The affine transformation matrix is used to align the features from different point cloud data so that the model is invariant to feature space variance from different point clouds. A shared Multi-layer perceptron (MLP) with ReLU activation and batch normalization is then used to extract point-wise features from the input. Note that both the T-net and MLP operations operate on one point independent of all others making the operation agnostic to point ordering. After two spatial transformation networks and two MLPs, 1024-dimensional features are extracted for each point ($n \times 1024$). The point-wise features are aggregated together using a max pooling operation to create a global feature of size 1×1024 representing the whole point cloud. Similar to the T-net and MLP operations, the global max pooling operation is also invariant to the order of the input points. This layer is simple to use, effective and produces reliable results when dealing with outliers and missing data. Finally, the global feature vector is passed through a fully connected network to predict classification scores for k predefined classes, as shown in the top branch of Figure 2.20. For tasks that need local structure information like part segmentation (in the bottom branch), local and global feature vectors are concatenated together, resulting in a vector size (n$\times$1088). This allows the prediction of the point-wise classification score for m predefined classes while considering both local and global information at the same time.

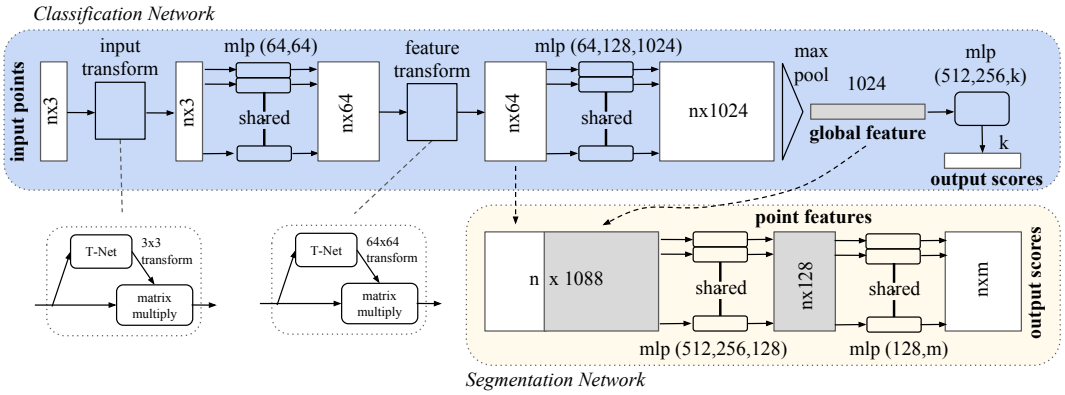


FIGURE 2.20: **Architecture of PointNet method proposed by Qi et al. [51, Fig. 2].**

PointNet++, proposed by Qi et al. [52], is a multi-stage and hierarchical extension of PointNet [51]. They point out that the original PointNet does not capture local features because the features of each point are extracted and aggregated directly into the global signature. It is more generalizable to unseen cases when local patterns can be abstracted along the hierarchy. Thus, PointNet++ applies the PointNet architecture to a hierarchical partitioning of the input data. Its architecture is shown in Figure 2.21. Using the distance metric of the underlying space, the set of points is divided into overlapping local regions. Using PointNet, each local region is processed to extract local features. The farthest point sampling (FPS) is used to partition the data. Input data and the underlying Euclidean space determine the receptive field. Another key feature is its ability to combine features from multiple scales to address the issue of varying sampling densities, resulting in improved robustness and detail capture.

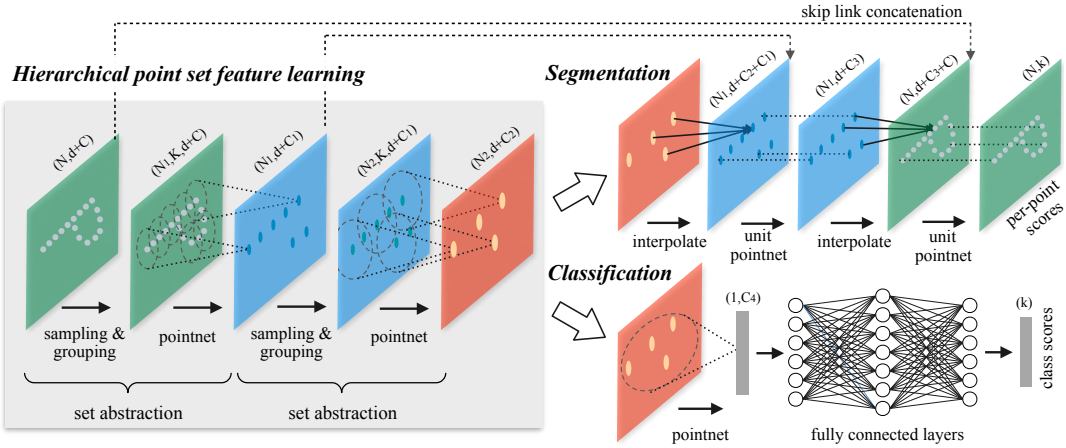


FIGURE 2.21: **Architecture of PointNet++ method proposed by Qi et al. [52, Fig. 2].**

Shi et al. [67] proposed the part-aware and aggregation 3D object detection neural network. Its architecture is shown in Figure 2.22. The framework consists of two stages: part-awareness and part-aggregation. By first learning from free-of-charge supervisions derived from 3D ground-truth boxes, the part-aware stage predicts coarse 3D proposals and precise intra-object part locations simultaneously. They have developed a new RoI-aware point cloud pooling module that groups predicted intra-object part locations within the same proposals, creating an effective way to encode features within 3D proposals. After the part-aggregation stage, the box location is refined based on the pooled part locations.

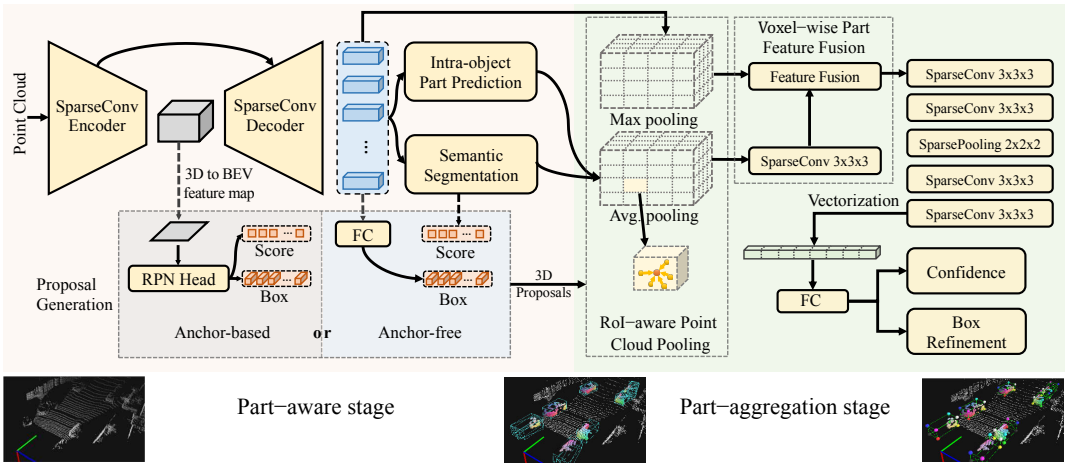


FIGURE 2.22: **Architecture of Part-A^2 Net method proposed by Shi et al. [67, Fig. 2].**

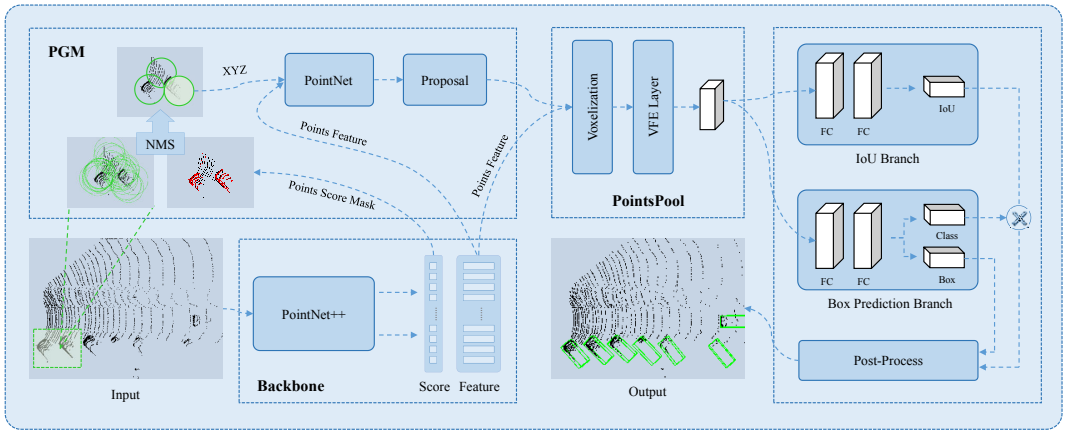


FIGURE 2.23: **Architecture of STD method proposed by Yang et al. [69, Fig. 1].**

Yang et al. [69] proposed a two-stage 3D object detection. Its architecture is shown in Figure 2.23. Initially, spherical anchors are seeded onto raw point clouds to generate accurate proposals using a bottom-up proposal generation network. By eliminating computation from prior works, this technique achieves a higher recall rate. To generate proposal features, PointsPool is applied by transforming interior point features from sparse expressions into compact representations, which saves even more computation. During the second stage of the prediction process, they implemented an intersection-over-union branch to improve localization accuracy.

Here, we can see the power of LiDAR data. Although it does not give the color information like the image, it outputs the point cloud by an accurate distance measurement (laser). LiDAR-based methods show a very high performance compared to camera-based methods, especially in terms of depth information. However, there are still limitations such as obscure information about object category. For example, in some cases, it is difficult to distinguish whether it is a car or a bush based on point cloud data alone while this can be handled more easily looking at the image data. This is why methods based on data fusion have been developed presenting the advantages of LiDAR and the camera.

2.3.3 Multimodal Fusion-based 3D Object Detection Methods

LiDAR and camera sensor fusion is one of the major concerns in this thesis. In the literature, there are three main fusion methods: early fusion-based [29, 44, 53, 73–78] where the raw data is fused in data-level or feature-level to form a tensor data of numerous channels; late fusion-based [79] where the fusion takes place in the decision-level; and deep fusion-based [74, 80–83] where the fusion is carefully constructed to combine the advantages of both early and late fusion systems.

In 2018, Qi et al. [29] presented the Frustum-PointNet architecture. As shown in Figure 2.24 , the method is composed of three phases: 3D frustum proposal, 3D instance segmentation, and 3D bounding box estimation. The first phase of this procedure is to produce 2D region proposals. By extruding the matching 2D region proposal under 3D projection, a 3D frustum proposal is generated, which contains all points in the LiDAR point cloud that lie inside the 2D region. The instance segmentation stage feeds the frustum proposal point cloud to the PointNet segmentation network [51], which classifies each point and determines if it is linked with the discovered item. In the last stage, all positively classified points are loaded into a new PointNet that estimates 3D bounding box parameters. The network regresses residuals relative to the segmented point cloud centroid for the bounding box center estimation. A classification regression hybrid is used to calculate bounding box size and heading angle, as inspired by Mousavian et al. [38].

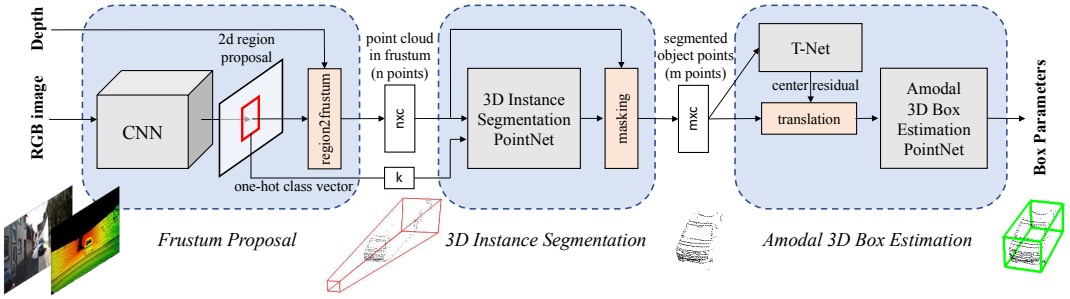


FIGURE 2.24: **Architecture of Frustum-PointNet method proposed by Qi et al. [29, Fig. 2].**

In 2017, MV3D [53] introduced an architecture utilizing both monocular image and LiDAR information. Its architecture is shown in Figure 2.25. The LiDAR point cloud is projected onto both a 2D top-view and a 2D front-view, from which feature maps are extracted using separate CNNs. In the feature extraction stage, a feature map is also extracted from the monocular image. The LiDAR top-view feature map is passed to an RPN to output proposal 3D bounding boxes. Each of these 3D proposals is projected onto the feature maps of all three views, and a fixed-size feature vector is extracted for each view by using pooling. The three feature vectors are then fused in a region-based fusion network, which finally outputs class scores and regresses 3D bounding box residuals. The feature vectors are fused by combining the three vectors by element-wise mean, feeding the combined vector through three separate fully-connected layers, and then once again combining the resulting vectors by element-wise mean.

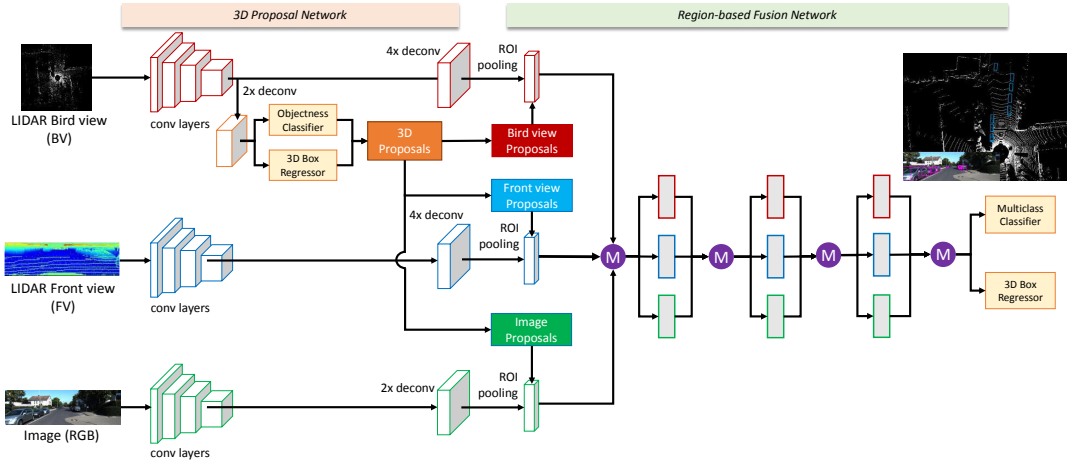


FIGURE 2.25: **Architecture of MV3D method proposed by Chen et al. [53, Fig. 1].**

A quite similar approach, also utilizing the PointNet architecture [51], was independently presented by Xu, Anguelov, and Jain [73]. In this work, the image-and-LiDAR architecture PointFusion was introduced in Figure 2.26. Just as in Frustum-PointNet [29], a SOTA 2D object detector is used to extract 2D region proposals (ResNet) which are extruded to the corresponding frustum point cloud. Each frustum is fed to a PointNet, extracting both point-wise feature vectors and a global LiDAR feature vector. Each 2D image region is also fed to a CNN that extracts an image feature vector. For each point in the frustum, its point-wise feature vector is concatenated with both the global LiDAR feature vector and the image feature vector. This concatenated vector is finally fed to a shared MLP, outputting 8×3 values for each point. The output corresponds to predicted (x, y, z) offsets relative to the point for each of the eight 3D bounding box corners. The points in the frustum are thus used as dense spatial anchors. The MLP also outputs a confidence score for each point, and in inference the bounding box corresponding to the highest-scoring point is chosen as the final prediction.

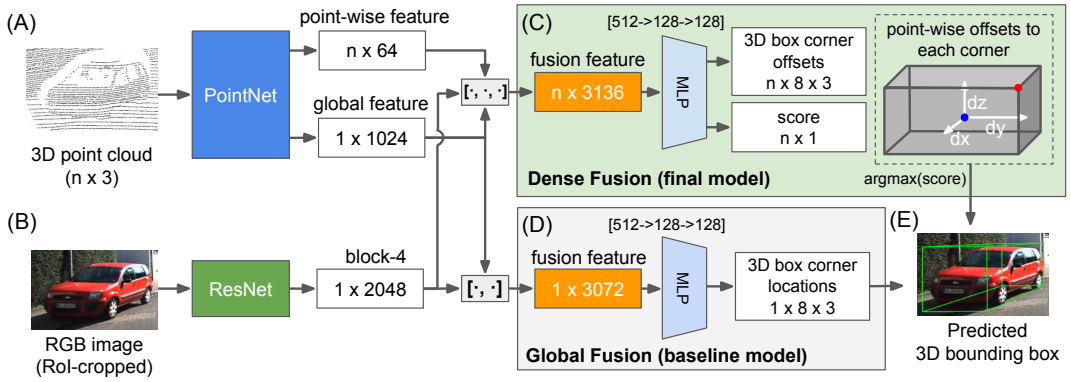


FIGURE 2.26: **Architecture of PointFusion method proposed by Xu, Anguelov, and Jain [73, Fig. 2].**

Another fusion architecture named AVOD [44] was introduced in 2018. Its architecture is shown in Figure 2.27. The LiDAR point cloud is projected onto a 2D top-view, from which a feature map is extracted by a CNN. A second CNN is used to extract a feature map also from the input monocular image. The two feature maps are shared by two subnetworks: an RPN and a second stage detection network. The architecture is thus similar to that in [53], the key difference being that AVOD uses both image and LiDAR features also in the RPN. The reported 3D detection performance is a slight improvement compared to [53] and is comparable to that of [63] for cars, but somewhat lower for pedestrians and cyclists. They also found that utilizing both image and LiDAR features in the RPN, as compared to only using LiDAR features, has virtually no effect on the performance for cars, but a significant positive effect for pedestrians and cyclists.

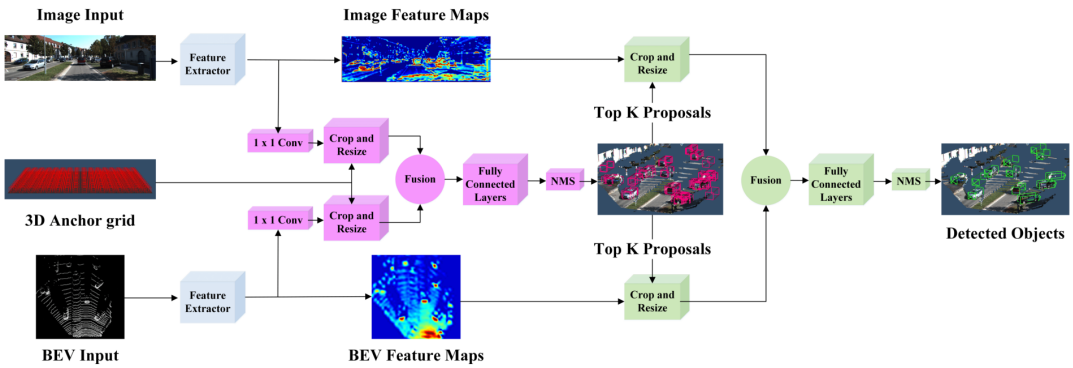


FIGURE 2.27: **Architecture of AVOD method proposed by Ku et al. [44, Fig. 2].**

In general, fusion-based methods are still difficult to effectively combine the two

data sources. The SOTA methods for 3D object detection are still based on LiDAR only. However, even if the results of fusion are not good, it is still necessary to study it, especially when either or both sensors are affected by the environment, such as in foggy weather conditions. So in the next section, we will briefly introduce some methods to improve the model in extreme weather conditions.

For a more complete survey on recent developments in object detection for self-driving cars, the reader is referred to [84–87].

2.4 Object Detection Methods in Foggy Weather Conditions

Self-driving cars are initially operated and tested in sunny locations during the day. However, to build a level 5 self-driving car, the vehicle must be able to move in all circumstances or all weather conditions. While many studies as mentioned in Section 2.2 and 2.3 have been proposed and achieved with certain successes in favorable weather conditions, the perception in adverse weather conditions is still limited and there are still many challenges to be solved. Rain, snow, and fog may all obstruct a self-driving car vision, just as they can affect human drivers. We focus on foggy weather conditions in this thesis, as this phenomenon is common and has a major impact on artificial vision systems [88–92].

Concerning artificial vision under foggy conditions, there is a substantial corpus of research on fog detection [93–96]. The classification of scenes into fog and fog-free conditions has also been addressed by Pavlic, Rigoll, and Ilic [97]. However, the works that take into account fog in 2D or 3D object detection methods are still limited. Previous studies have demonstrated how performance drops in foggy scenes for segmentation [98–102], 2D object detection [83, 98, 102] and depth estimation [102] tasks.

The lack of work for object detection applied to the autonomous vehicle taking into consideration fog comes from the fact that it is difficult to obtain datasets with fog. In the general case, the availability of huge, labeled datasets has contributed significantly to the advancement of computer vision in recent years [103, 104]. On the other hand, it is inefficient to collect and annotate such dataset for each new problem (e.g., fog, rain). For this, two approaches are currently being explored:

- A small-scale fog chamber prototype created by Colomb et al. [105] can produce steady sight levels and uniform fog to evaluate the reactions of drivers or algorithms under degraded conditions.
- The addition of digitally simulated degraded weather conditions on images initially acquired in clear weather Tarel et al. [106] created synthetic fog and segments hazy photos into free-space and vertical objects. In [83], numerous sensor inputs are combined to improve car detection in fog.

Learning synthetic data is gaining popularity because this method is generally efficient. There are some noteworthy examples. To train dense optical flow regression networks, Dosovitskiy et al. [107] used visualizations of a floating chair. In order to learn an end-to-end text identification system, Gupta, Vedaldi, and Zisserman [108] imposed text onto natural photos. Vázquez et al. [109] used virtual data to train pedestrian detectors. Using video game engines, methods [110–112] created images with dense semantic annotations, which are then combined with real data to improve the semantic segmentation performance of recent CNN architectures. According to Johnson-Roberson et al. [113], training CNN models on synthetic images is more effective than training them on vast, real-world datasets such as Cityscapes for vehicle recognition.

With these two foggy data collection solutions, some studies on 3D object detection have shown that the performance can be improved in three ways: by learning with synthetic data [98–100, 102], dehazing/ defogging [98] or using late-based fusion [83, 101].

In this thesis, we propose to use synthetic data in order to optimize the learning phase and to obtain better results in fog conditions. For a more complete survey on recent developments autonomous driving in adverse weather conditions, the reader is referred to [114].

2.5 Conclusion

In this chapter, we saw how deep learning-based algorithms have demonstrated outstanding performance and significant potential in analyzing and detecting objects in three dimensions in recent years. Our goal in doing this SOTA and analysis is to gain a better understanding of deep learning models used in detecting 3D objects for autonomous vehicles. Based on more than 100 linked publications, a detailed evaluation of several deep learning architectures and their applications in 2D and 3D object detection has been mentioned. By comparing several deep learning techniques for object detection, we can find the best SOTA deep architecture. LiDAR-based methods are found to achieve outstanding performance. Meanwhile, Pseudo-LiDAR opens up new avenues for camera-based, where Pseudo-LiDAR++ is the best solution in terms of accuracy for camera-based methods. Fusion-based methods are still far from over all because of the difficulty of combining two disparate data sources. Furthermore, the properties of the most prominent deep learning architectures for 3D object detection were examined in order to identify current trends and open difficulties for future research in this field. We have therefore used these elements to guide the new architecture that we present in Chapters 4. In addition, we also briefly introduced the effects of weather, especially fog, on the vision-based models. Our work on object detection in foggy weather conditions is presented in Chapter 5.

Besides the well-designed architectures, another indispensable and extremely important part of deep learning-based solutions is the dataset. In the next chapter, we introduce the existing datasets for self-driving cars and the datasets used in this thesis for both the favorable and foggy weather conditions datasets.

Chapter 3

Dataset

Contents

Chapter overview: This chapter provides an overview of the datasets used in this thesis. We begin by analyzing and exploring the existing datasets for autonomous driving. The KITTI dataset was selected as the primary dataset for this thesis due to its widespread use in research and collection under favorable weather conditions. To reduce costs, we extracted 4-beam point clouds from the original 64-beam LiDAR (KITTI dataset) for our experiments. We also discuss the limited availability of existing datasets for adverse weather conditions and introduce our proposed Multifog KITTI dataset, which is an augmentation of the KITTI dataset with foggy weather conditions, for use in our experiments.

3.1 State-of-the-art of Datasets for Autonomous Driving

The availability of large-scale datasets is critical for data-driven deep learning-based algorithms to succeed. Several existing datasets for autonomous driving scenarios

have been released [12, 103, 115–141] and are summarized in Table 3.1. Among these datasets, KITTI [12], Citysapes [103], Ford [120], Waymo [126], and nuScenes [123] are the most widely utilized and contribute significantly to the advancement of the perception in general and object detection for autonomous driving in particular. KITTI, Cityscapes and Ford are acquired under favourable weather conditions whereas Waymo and nuScenes contain all kind of weather conditions. The major details regarding these benchmarks, including dataset size, variety, and extra data, are shown below.

TABLE 3.1: **Autonomous driving datasets.** C and L denote camera and LiDAR, respectively. Size of the dataset is shown non-uniformly in terms of units (e.g., hours, kms, frames) as it is not easy to collect information.

Datasets	Sensors		Size	Release date	Citation
	C	L			
KITTI [12]	✓	✓	6h	2012	9403
Cityscapes [103]	✓		25k frames	2016	7729
nuScenes [123]	✓	✓	242 km	2019	1684
Oxford RobotCar [124]	✓	✓	1000km	2016	1007
Mapillary [134]	✓		1.6M	2021	819
Waymo open perception [126]	✓	✓	10.83h	2019	816
BDDV [142]	✓		10k h	2016	758
GTSDB [132]	✓		N/A	2012	703
BDD100K [131]	✓		1k h	2020	624
SEMANTIC3D [128]		✓	N/A	2016	507
H3D [121]		✓	27.7k frames	2019	406
Ford [120]	✓	✓	100 GB	2011	334
STC [115]		✓	14k	2011	220
Paris Lille 3D [127]		✓	1.9 km	2018	179
ApolloScape [125]	✓	✓	100h	2018	166
TorontoCity [130]	✓	✓	8439 km	2016	162
A2d2 [143]	✓	✓	12k frames	2020	159
KAIST [116]	✓	✓	191k km	2017	149
Lyft Level5 perception[122]	✓	✓	2.5 h	2019	132
TerraMobilita [129]		✓	N/A	2014	119
LiVi-Set [119]	✓	✓	100km	2018	100
ApolloCar3D [117]	✓		5.2k frames	2019	94
A*3D [138]	✓	✓	55h	2019	60
EU Long-term [136]	✓	✓	63.4 km	2021	43
BLVD [118]	✓	✓	120k frames	2019	32
RELLIS-3D [139]	✓	✓	6k frames	2021	32
HUAWEI ONCE [141]	✓	✓	114h	2021	29

Continued on next page

Table 3.1 – *Continued from previous page*

Datasets	Sensors		Size	Release date	Citation
	C	L			
Brno Urban [137]	✓	✓	375.7 km	2020	18
PandaSet [135]	✓	✓	0.23h	2020	11
Cirrus [140]	✓	✓	6.2k frames	2021	5

In the Chapter 4, we present our deep learning-based 3D object detection algorithm taking a stereo camera and 4-beam LiDAR (low-cost sensors) as inputs. Our method involves training on Scene Flow dataset [144] and finetuning on KITTI dataset [12] for depth estimation and 3D object detection on the KITTI dataset for 3D object detection as follows Wang et al. [43] and You et al. [46]. Then, comparing our method to other methods in the literature is straightforward. Furthermore, KITTI dataset is quite well-prepared: clean (no corrupted data), preprocessed (data in a format suitable for training implementations), annotated/ labeling available, and also well done in calibration, synchronizing camera and LIDAR. In the next section, we introduce the details of the Scene Flow dataset and KITTI dataset as well as how to generate a 4-beam point cloud for our experiments from 64-beam point cloud.

3.2 Favorable Weather Conditions Datasets

3.2.1 Scene Flow Dataset Overview

Our proposed architecture in the next chapter is based on the Pseudo-LiDAR pipeline [43] in which the training procedure contains two parts: depth prediction and 3D object detection. The corresponding dataset for each part is introduced as follows. The Scene Flow dataset [144], a large-scale synthetic dataset, is used first for training our depth estimation part. The dataset contains over 39,000 stereo frames with, 960×540 for training including dense ground truth for optical flow, disparity, and disparity change, as well as segmented objects, and 4,370 images for testing. It is rendered from synthetic sequences.

The Scene Flow dataset (including with RGB images for FlyingThings3D, Monkaa, and Driving) is used here as a pre-trained dataset for our algorithm. Figure 3.1 shows examples of Scene Flow dataset.

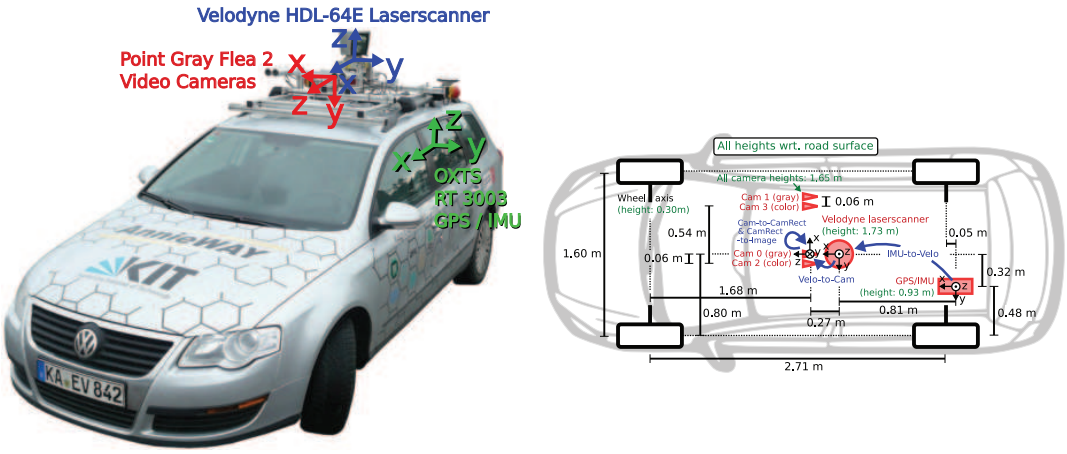
3.2.2 KITTI Dataset Overview

We present the main dataset for this thesis, the KITTI dataset [12]. This dataset is used for both fine-tuning the depth estimation part and for training the 3D object detection part of our algorithm. To avoid confusion, from now it is referred to as Clear KITTI dataset.

FIGURE 3.1: **Examples from the Scene Flow dataset [144, Fig. 7]**. This figure illustrates the diversity of this dataset. From left to right are the left color synthetic image and the corresponding disparity map, respectively.

The KITTI Vision Benchmark Suite offers an excellent set of datasets for automotive applications such as car and pedestrian recognition [64, 145]. These datasets include optical flow, stereo vision, visual odometry, SLAM, and object detection. KITTI object detection dataset is one of the most common dataset for driving scenes collected in the daytime and under favorable weather conditions. This dataset contains 7,481 training samples and 7,518 testing samples for both images (with a resolution of 1242 $\times$ 375) and point cloud. Like most other studies [53, 146], the training dataset is divided into a training part (3,712 samples) and a validation part (3,769 samples). In addition to the easy-to-use dataset formats, Geiger, Lenz, and Urtasun [12] provided a vast amount of information on every labeled object and provided high-quality images. They used the annotation files to apply a filter based on distance, car type, and orientation to retrieve a total of 1500 rear-view car images.

As shown in Figure 3.2 (a), the camera and the LiDAR utilize two distinct coordinate systems (red for the camera and blue for the LiDAR). The directions (X, Y, Z) are set as (rightward, downward, forward) and (forward, leftward, upward) from the camera and LiDAR, respectively. All ground-truth data is delivered in camera coordinates, but it is not difficult to convert from camera to LiDAR coordinates and vice versa by using the calibration information. As can be seen in Figure 3.2 (b), the sensors (red) are positioned with respect to the vehicle body according to their dimensions. The measurements are carried out according to the road surface and the height above the ground is indicated in green. The blue indicates transformations



(a) KITTI recording platform (b) KITTI sensor setup

FIGURE 3.2: **KITTI vehicle setup [147, Fig. 1, 3].** (a) fully equipped vehicle (Volkswagen Passat B6); (b) setup bird's eye view (BEV).

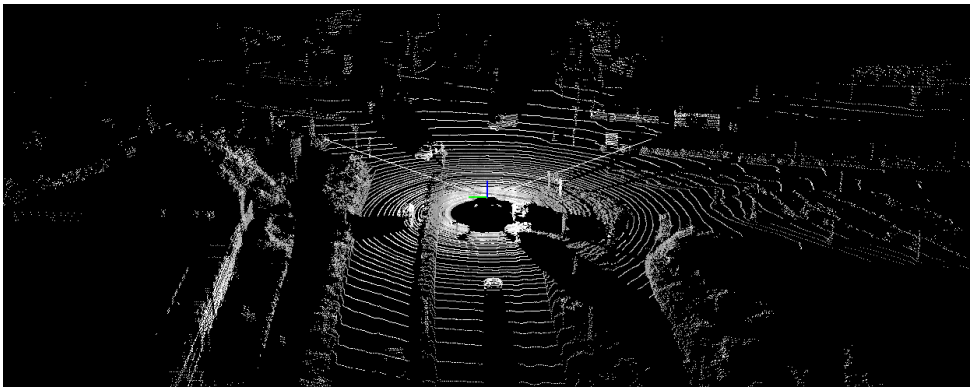
among the sensors.

An example from KITTI dataset is shown in Figure 3.3. Figure 3.3 (a) is the RGB image of size 1242×375. Figure 3.3 (b) shows the point cloud corresponding to the RGB image presented in Figure 3.3 (a). Points and objects that are not in range of the front camera are filtered out following the BEV range (side range of [-40 m, 40 m] and forward range of [0 m, 70.4 m]), as shown in Figure 3.3 (c). We only focus on objects that appear in the image (left camera). As a result, it is necessary to perform filtering on the point cloud to keep those in the camera's field of view. Figure 3.4 shows some examples of KITTI dataset, including the left color image and the corresponding point cloud from 64-beam LiDAR after filtering.

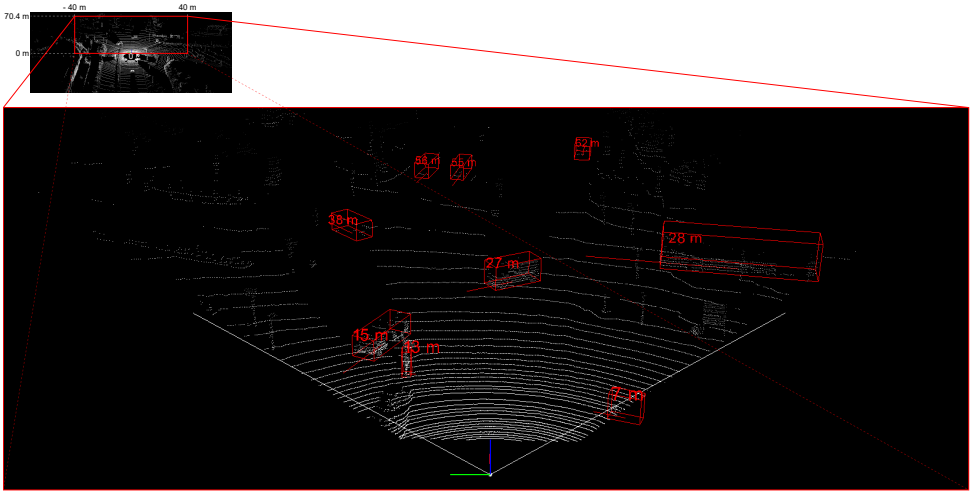
Once the data filtering is finished, we also prepare data for the depth estimation phase. For this, it is imperative to have a dense point cloud, unlike the classic point cloud of the 64-beam LiDAR. We use the accumulation of the 11 frames point clouds in order to obtain more density depth map as represented by Wang et al. [49]. 11 frames point cloud means that KITTI dataset keeps ± 5 frames for each current point cloud. Figure 3.5 shows the 64-beam LiDAR in Figure 3.5 (a) and the accumulation of 11 frames 64-beam LiDAR in Figure 3.5 (b). As we can see the accumulation of 11 frames 64-beam LiDAR gives more points, therefore, we can have a denser depth map. We use it as the ground truth of our depth estimation phase.

The KITTI object detection benchmark has also delivered label files in text file format. Each line in a label file describes a single ground-truth object that contains the information listed in Table 3.2. KITTI divides the world into eight categories: car, van, truck, pedestrian, person sitting, cyclist, tram, and misc. Our experiments concentrate only on the Car data category.

(a) KITTI image example



(b) KITTI point cloud example



(c) KITTI point cloud after filtering

FIGURE 3.3: **KITTI dataset example.** (a), (b) and (c) show the left color image, the point cloud from 64-beam LiDAR and the filtered point cloud. Ground truth 3D bounding box objects are shown in red in the LiDAR coordinate system.

TABLE 3.2: **KITTI ground truth annotations for object detection**.

Attribute	Values	Description
type	1	Describes the type of object: 'Car', 'Van', 'Truck', 'Pedestrian', 'Person_sitting', 'Cyclist', 'Tram', 'Misc' or 'DontCare'.

Continued on next page

Table 3.2 – *Continued from previous page*

Attribute	Values	Description
truncated	1	Float from 0 (non-truncated) to 1 (truncated), where truncated refers to the object leaving image boundaries.
occluded	1	Integer (0, 1, 2, 3) indicating occlusion state: 0 = fully visible, 1 = partly occluded 2 = largely occluded, 3 = unknown.
alpha	1	Observation angle of object, ranging $[-\pi, \pi]$.
bbox	4	2D bounding box of object in the image (0-based index): contains left, top, right, bottom pixel coordinates.
dimensions	3	3D object dimensions: height, width, length (in meters).
location	3	3D object location x, y, z in camera coordinates (in meters).
rotation_y	1	Rotation r_y around $Y - axis$ in camera coordinates $[-\pi, \pi]$.
score	1	Only for results: Float, indicating confidence in detection, needed for p/r curves, higher is better.

The difficulty of detecting objects varies depending on the situation. Some items, such as occluded objects, long-range objects, and so on, are more difficult to detect. The farther away an object is from the vehicle, the smaller it appears in the RGB image and the fewer points it reflects back into the point cloud. As shown in Table 3.3, the KITTI object detection benchmark has three difficulty levels to evaluate the object detection system. The object size in the image, occlusion, and truncation distinguish these three levels (Easy, Moderate and Hard).

TABLE 3.3: **KITTI 3D object detection benchmark's object attribute parameters for the three difficulty levels.** Difficulty of an object is defined by its object size, occlusion ratio, truncation ratio.

	Easy	Moderate	Hard
min. 2D bbox height	40 pixels	25 pixels	25 pixels
max. occlusion level	fully visible	partly occluded	difficult to see
max. truncation	15%	30%	50%

Figure 3.6 shows examples of how KITTI defines the difficulty for each object. Easy, Moderate and Hard objects are shown in yellow, blue and pink 3D bounding boxes, respectively.

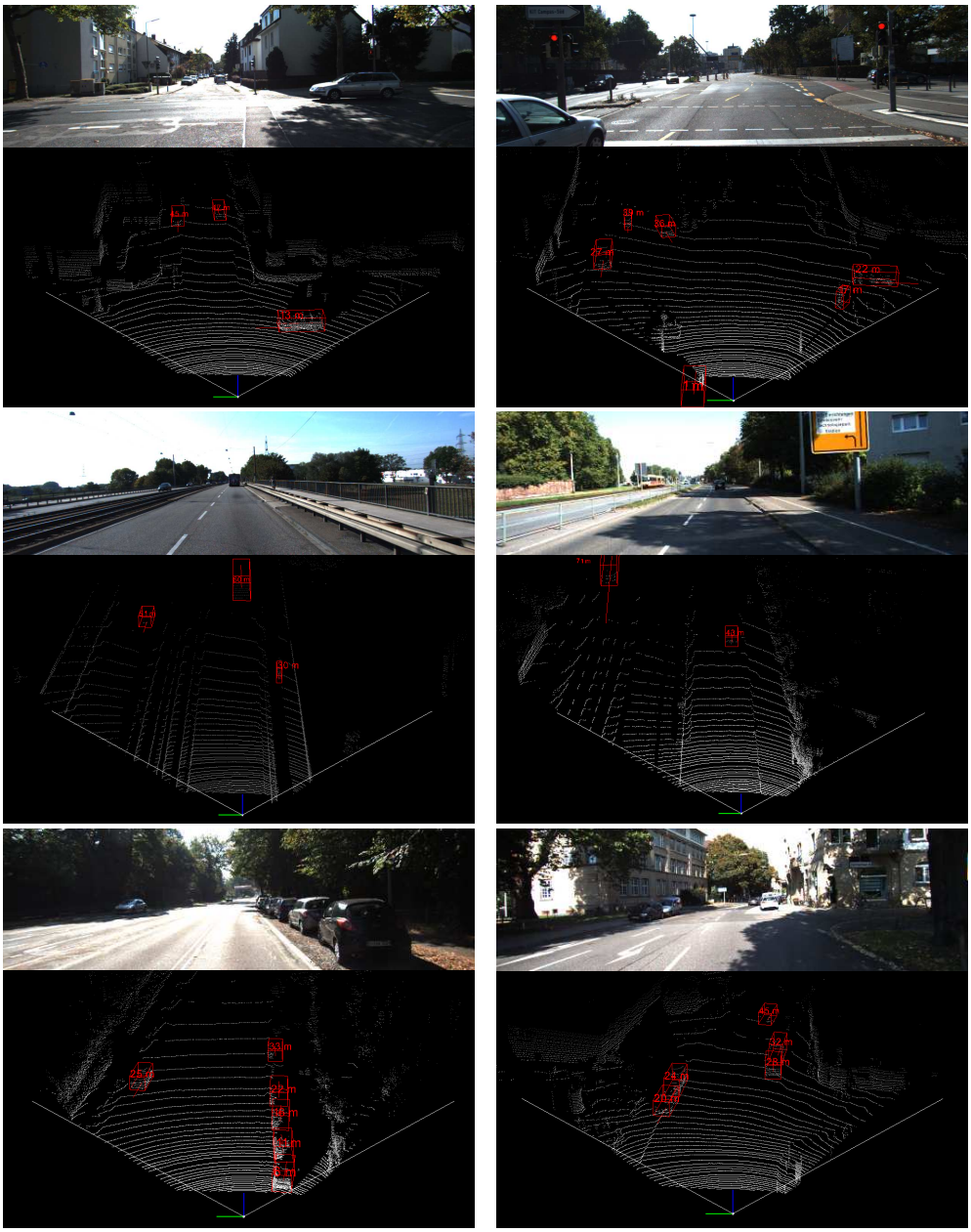


FIGURE 3.4: **Examples from the KITTI dataset.** This figure shows some examples from KITTI dataset including the left color image, point cloud and ground truth bounding boxes (red) in LiDAR coordinate.

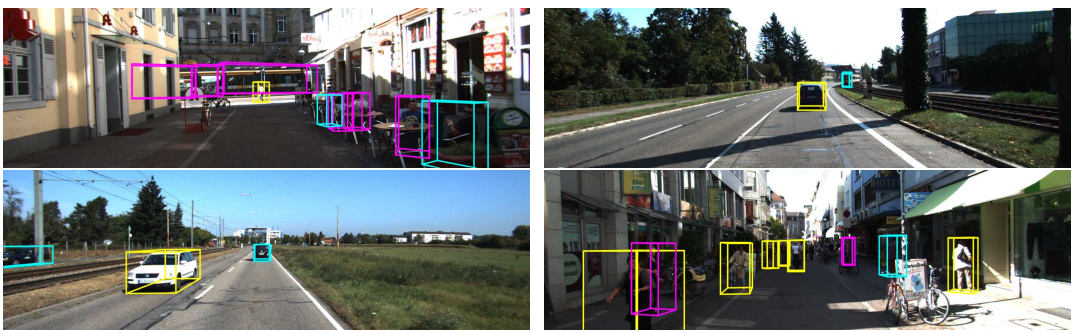
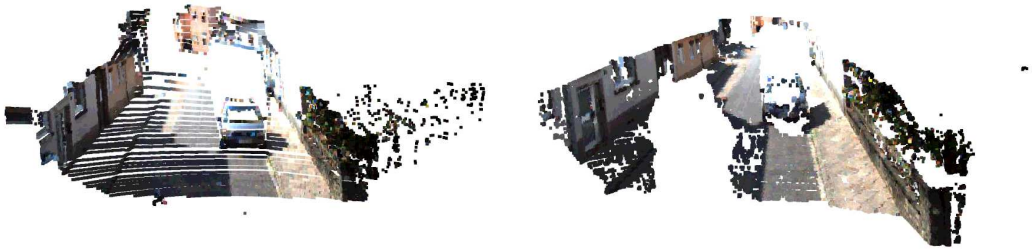


FIGURE 3.6: **Example of the difficulty level of the objects in the KITTI dataset** corresponding to the definition in Table 3.3. Easy, Moderate and Hard objects are shown in yellow, blue and pink 3D bounding boxes.



(a) 64-beam point cloud (b) Depth map ground truth

FIGURE 3.5: **Comparison between sparse LiDAR and depth map ground truth.** (a) 64-beam LiDAR (sparse); (b) depth map ground truth (dense): accumulation of 11 frames 64-beam LiDAR.

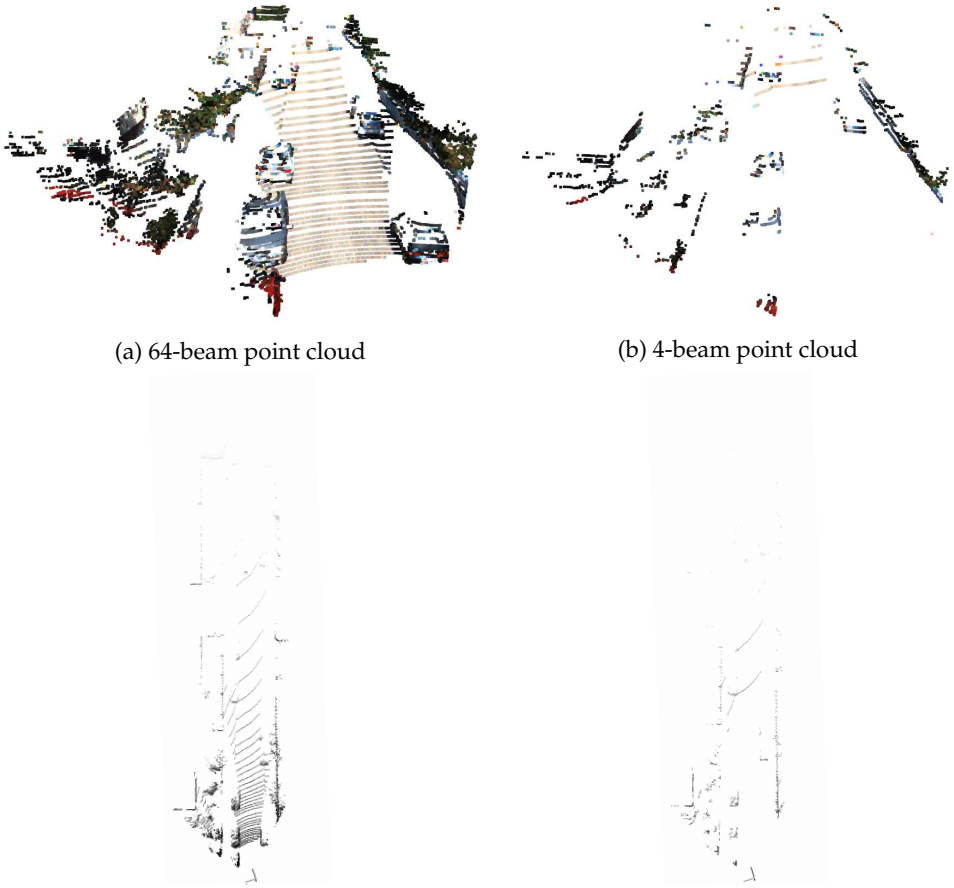
KITTI provides a C++ program that can be used to calculate 3D AP (AP_{3D}) and BEV AP (AP_{BEV}) for any frame from the training set for each difficulty level. This is used to evaluate the performance of our models.

3.2.3 Downsampling 64-beam Point Cloud

This thesis focuses on low-cost sensor technologies. Our method uses a 4-beam LiDAR point cloud, which is much cheaper than 64-beam LiDAR. For simplicity and fair comparison, we generated a 4-beam point cloud from the original 64-beam point cloud (64-beam Velodyne) taken from the KITTI dataset instead of using real data from 4-beam LiDAR sensors (like SCALA). This extracted point cloud is inspired by You et al. [46]. Given a point cloud as a set of 3D points $\{P_i | i = 1, ..., n\}$, where each point P_i is a vector of its (x, y, z) coordinate, they firstly compute the elevation angle θ_i to the LiDAR sensor for each point (x_i, y_i, z_i) of the point cloud in one scene (in LiDAR coordinate system ((x, y, z) corresponding (front, left, up), and $(0, 0, 0)$ is the location of the LiDAR sensor)) as follows:

$$\theta_i = \arccos\left(\frac{\sqrt{x_i^2 + y_i^2}}{\sqrt{x_i^2 + y_i^2 + z_i^2}}\right) \tag{3.1}$$

By grouping the points according to their elevation angles, the point cloud is sliced into different lines by step 0.4° starting from -23.6° (close to the Velodyne 64-beam LiDAR specifications). A 4-beam LiDAR consists of points whose elevation angles are between four intervals $[-2.4^\circ, -2.0^\circ] \cup [-1.6^\circ, -1.2^\circ] \cup [-0.8^\circ, -0.4^\circ] \cup [0.0^\circ, 0.4^\circ]$ corresponding to each beam. The choice of 4-beam LiDAR SCALA was made based on its specifications so that each consecutive line had an interval of 0.8°. Figure 3.7 depicts the 64-beam and extracted 4-beam LiDAR point cloud.



(a) 64-beam point cloud (b) 4-beam point cloud

(c) 64-beam point cloud in BEV (d) 4-beam point cloud in BEV

FIGURE 3.7: **Visualization of 64-beam LiDAR and the extracted 4-beam LIDAR.** (a) and (c) show point cloud colored according to RGB image and point cloud in BEV, respectively, based on point cloud from 64-beam LiDAR. (b) and (d) show the same thing but for point cloud from the extracted 4-beam LiDAR. The point clouds (a), (b) are colored according to RGB image.

Finally, we have the dataset consisting of images and 64-beam LiDAR point cloud, as well as a 4-beam point cloud extracted for our experiments. This dataset was collected under favorable weather conditions. To simulate adverse weather conditions, we generate a foggy weather conditions dataset, called Multifog KITTI, which is augmented on KITTI dataset. Detailed information about this dataset can be found in the following section.

3.3 Fog Augmented KITTI – Multifog KITTI Dataset

3.3.1 Adverse Weather Conditions Datasets

Most common existing datasets have been collected under favorable conditions such as KITTI [12], Cityscape [103], or in different lighting conditions such as BDD100K [131], Waymo [126], NuScenes [123]. In recent years, attention has been drawn to

how self-driving cars perceive in adverse weather conditions, since such conditions negatively impact camera and LiDAR sensing quality, resulting in lower performance. Consequently, some datasets have been collected in fog (including Foggy Driving [98], Foggy Zurich [99], SeeingThroughFog [83], nuScenes [123], BDD100k [131], Oxford [124, 148], [88], [149]), rain [149, 150] or snow [149, 151, 152] conditions. Most common foggy weather conditions datasets are summarized in Table 3.4.

TABLE 3.4: **Foggy weather conditions datasets for autonomous driving.** C and L denote camera and LiDAR, respectively. Size of the dataset is shown non-uniformly in terms of units (e.g., hours, kms, frames) as it is not easy to collect information.

Datasets	Sensors		Size	Release date	Citation
	C	L			
Foggy Cityscapes-DBF [103, 153]	✓		N/A	2016	7776
Foggy Cityscapes [98]	✓		20.5k frames	2018	473
Foggy Driving [98]	✓		101 frames	2018	473
Foggy Zurich [99]	✓		3.8k frames	2018	107
DENSE [83]	✓	✓	13.5k frames	2020	99
Weather augmented [154]	✓		7480+2995 frames	2019	63
ACDC [155]	✓		N/A	2021	52
Foggy Synscapes [100]	✓		498	2019	32
Cerema database [88]	✓		62k frames	2016	10

Data gathered in such conditions is not easy. Additionally, it has some problems such as the absence of a labeled object detection dataset under adverse weather conditions as well as the difficulty of controlling fog density (it cannot control all fog levels, or it is difficult to collect all fog levels). Synthetic datasets, on the other hand, are increasingly similar to real data, and we can avoid these problems by using them. Synthetic datasets can be divided into two categories: physics-based such as Foggy Cityscapes [98], Foggy Cityscapes [99], Rain augmented [100, 154] and generative adversarial network-based (GAN-based) such as [154].

Despite the usefulness of these datasets above, KITTI dataset [12] is commonly used in the literature, and it is easy to work on. We decide to use this dataset as a base dataset for further fog rendering on it. While most synthetic datasets focus only on images [98–100, 154], our work aims to take into account fog for both image and point cloud starting from a favorable weather dataset [12]. We use the physics-based procedure proposed by Bijelic et al. [83] for generating the Multifog KITTI dataset to retain physical properties like real fog. We represent how generating this dataset as follows.

3.3.2 Fog Rendering

3.3.2.1 Fog Definition

Before we explain how the fog is rendered, it is important to understand what fog is. The fog is physically characterized from a microscopic point of view (distribution of the size of the drops) and from a macroscopic point of view (visibility through a thickness of fog). Only this second aspect is important for the study of images acquired from cameras and LiDAR of the road context. Indeed, with the use of a standard camera and LiDAR, the majority of the acquired wavelengths are part of the visible domain and near infrared domain. However, for wavelengths in the visible range/ NIR, the type of fog (size of the drops) has little impact on perception [156].

3.3.2.2 Camera through Fog

The physical quantity that characterizes fog is meteorological optical range (MOR), which is expressed in meters. In a road context, this size is characterized from the contrast. Based on the law of Koschmieder [157] in 1924, Sakaridis, Dai, and Van Gool [98] formulated the equation to obtain an observed foggy image $I_{foggy}(u,v)$ at pixel (u,v) as follows:

$$I_{foggy}(u,v) = t(u,v)I_{clear}(u,v) + (1 - t(u,v))L, \tag{3.2}$$

where $I_{clear}(u,v)$ denotes a latent clear image, L is the atmospheric light which is assumed to be globally constant (generally valid only for daytime images), and in case of a homogeneous medium, the transmission coefficient is:

$$t(u,v) = \exp(-\beta D(u,v)), \tag{3.3}$$

where β is the fog density (or attenuation) coefficient, and $D(u,v)$ is the scene depth at pixel (u,v). Fog thickness is controlled by fog density coefficient β. Using Koschmieder's law [157], visibility V can be described by the equation presented in [158, pp. I–9.4]:

$$C_T = \exp(-\beta V), \tag{3.4}$$

or,

$$V = -\frac{\ln(C_T)}{\beta}, \tag{3.5}$$

or,

$$\beta = -\frac{\ln(C_T)}{V}, \tag{3.6}$$

where C_T is the contrast threshold. $C_T = 0.05$ is considered to be the minimum recognizable contrast for humans by Prokes [159]. From equations (3.3) and (3.6),

we can also express $t(u,v)$ as a dependency on visibility V and the depth map D as follows:

$$t(u,v) = \exp(\frac{\ln(C_T)D(u,v)}{V}). \tag{3.7}$$

3.3.2.3 LiDAR through Fog

Bijelic et al. [83] assumes that beam divergence is not affected by fog. In this model, a returned pulse echo is always registered as long as the received laser intensity is larger than the effective noise floor. However, severe back-scatter from fog may lead to direct back-scatter from points within the scattering fog volume, which is quantified by the transmissivity $t(u,v)$. Then, the emitted laser beam intensity can be modeled using the following equation presented as follows:

$$L_{foggy}(u,v) = t(u,v)L_{clear}(u,v), \tag{3.8}$$

where $L_{clear}(u,v)$ and $L_{foggy}(u,v)$ are the intensity of the light pulses emitted from LiDAR and the received signal intensity, respectively.

Equation (3.8) assumes that the fog has no effect on the beam divergence. As long as the received laser intensity is greater than the effective noise floor, a returned pulse echo is always recorded in this model. Severe fog backscatter, on the other hand, might result in direct backscatter from points within the scattering fog volume, which is measured by the transmissivity $t(u,v)$ from Equation (3.2). Modern scanning LiDAR systems implement adaptive laser gain g to increase the signal for a given noise floor, as shown in [160], resulting in the maximum distance:

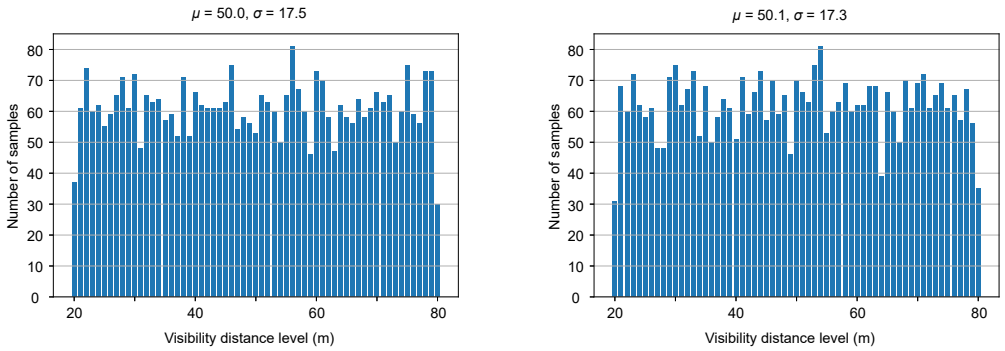
$$d_{max} = \frac{1}{2\beta} \ln\left(\frac{n}{L_{clear} + g}\right), \tag{3.9}$$

where n is the detectable noise floor. With the sum of the reciprocal of the received laser intensity from Equation (3.8) and gain, the detectable distance drops logarithmically as shown in Equation (3.9). As a result, in fog, LiDAR measurements suffer not only from peak intensity loss but also from back-scattering, which causes a peak-shift inside the fog volume, obliterating all information on the target scene point. To mimic fog-distorted LiDAR observations, the algorithm is proposed by Bijelic et al. [83]. The proposed algorithm is based on Equations (3.8) and (3.9), as well as additional fog chamber results that validate the hyperparameters used. Note that beam divergence in fog has been ignored, and we assume that the LiDAR depth measuring process is accurately described by a constant additive gain g and noise-floor n. They also assume that the detected object intensities drop exponentially according to the attenuation model in Equation (3.8).

Fog can be added to images and LiDAR data using the method we presented. As a result of applying this method to KITTI dataset, we are able to generate Multifog KITTI dataset.

3.3.3 Multifog KITTI dataset

The proposed Multifog KITTI dataset is generated by using the equations given above for different visibility levels from 20 to 80 meters (corresponding to heavy fog – medium fog range). As the depth map D is required in Equation (3.7) to calculate the transmission coefficient $t(u,v)$ and then $I_{foggy}(u,v)$ in Equation (3.2) and $L_{foggy}(u,v)$ in Equation (3.8) for each frame, the algorithm proposed by Park et al. [161] is employed to generate the predicted depth map D corresponding to each image on the left side I_l and the same for the right image I_r. This method takes an RGB image and a sparse depth image as input and results in an image where the value of each pixel is the depth information. The default configuration proposed by Bijelic et al. [83] is used, with $g = 0.35$ and $n = 0.05$ for the Velodyne HDL64 S2 LiDAR used in the KITTI dataset.



(a) Training set (b) Validation set

FIGURE 3.8: **Distribution of MOR of different parts in the Multifog KITTI dataset [162, Fig. 4]** (a) for training set with mean μ = 50.0 and σ = 17.5; (b) for validation set with mean μ = 50.1 and σ = 17.3.

Figure 3.8 shows the number of samples for each visibility level for the training and validation sets of the Multifog KITTI dataset. The distribution of fog visibility is uniform between 20 m and 80 m. This dataset is used similarly to the Clear KITTI dataset for both depth estimation and 3D object detection parts. The proposed Multifog KITTI dataset contains 7,481 training samples and 7,518 testing samples for stereo images, 64-beam LiDAR, and 4-beam LiDAR.

Figures 3.9, 3.10 show examples of the favorable weather conditions dataset (KITTI) including the left color image, the point cloud from 64-beam LiDAR, the extracted 4-beam point cloud based on 64-beam LiDAR and the corresponding one in foggy weather conditions. Figure 3.11 shows examples of our proposed Multifog KITTI

dataset. For each frame, it shows the left color on top, the 64-beam point cloud in the middle and the 4-beam point cloud at the end and the corresponding visibility V. In Figures 3.10 (b) and 3.11 (c), we see that the fog is not covered well in the sky, and it is not like reality. This is explained by the way that fog generation is based on depth information, where sky depth in this case is predicted to be 0 instead of infinity. Therefore, fog quality is highly dependent on the depth map where 64-beam LiDAR is very accurate, but sparse and the predicted depth map is dense but less accurate.

(a) Left color image (b) Foggy augmented on left color image

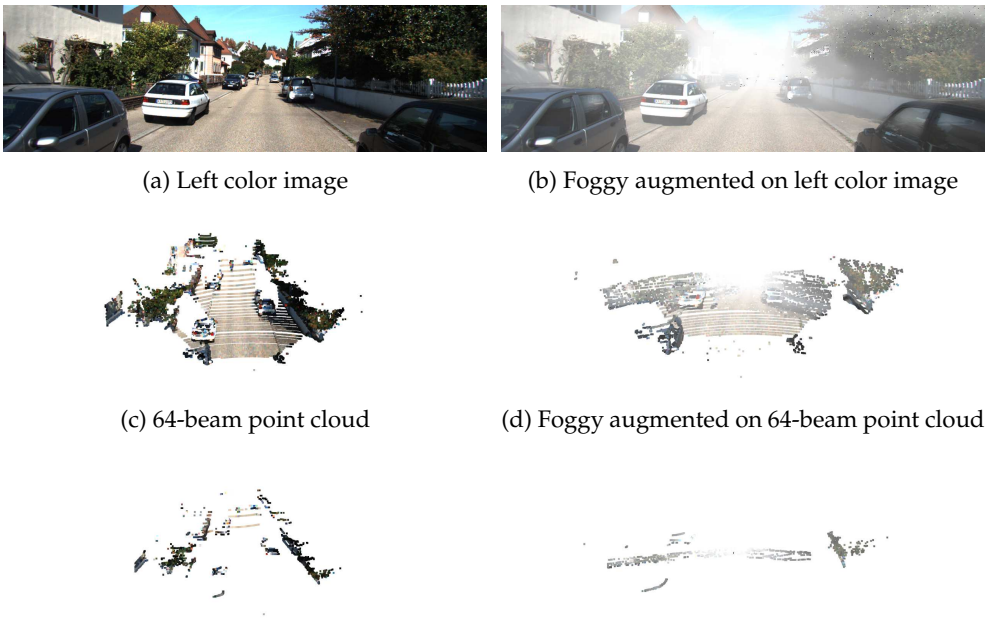
(c) 64-beam point cloud (d) Foggy augmented on 64-beam point cloud

(e) 4-beam point cloud (f) Foggy augmented on 4-beam point cloud

FIGURE 3.9: **From KITTI to Multifog KITTI dataset – example 1.** An example of KITTI dataset includes the left color image, the 64-beam point cloud, the extracted 4-beam point cloud and the corresponding one with the foggy augmentation (visibility V = 59 m). The point cloud is colored according to RGB image.

3.4 Conclusion

In this chapter, we introduced the datasets that we use for our object detection experiments in next chapters. The main dataset we work with is KITTI dataset. This dataset includes images and point cloud from 64-beam LiDAR. Besides, we wanted to experiment on low-cost sensors, so we extracted the 4-beam point cloud based on 64-beam LiDAR data. In the next chapter, we will introduce our 3D object detection method, and its experiments on this dataset. Besides, further, we want to perform experiments not only in favorable weather conditions, but also in extreme weather conditions, specifically here foggy weather conditions. Therefore, we have created a dataset, called Multifog KITTI, including both images and LiDAR, which are heavily



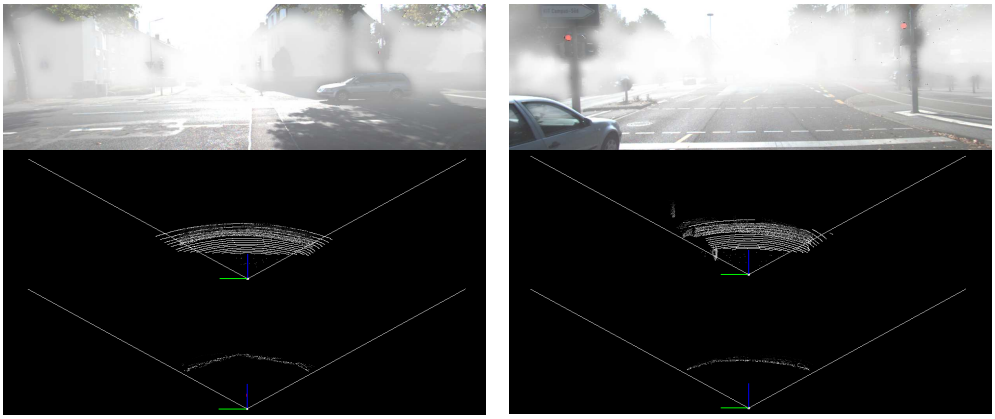
(a) Left color image (b) Foggy augmented on left color image

(c) 64-beam point cloud (d) Foggy augmented on 64-beam point cloud

(e) 4-beam point cloud (f) Foggy augmented on 4-beam point cloud

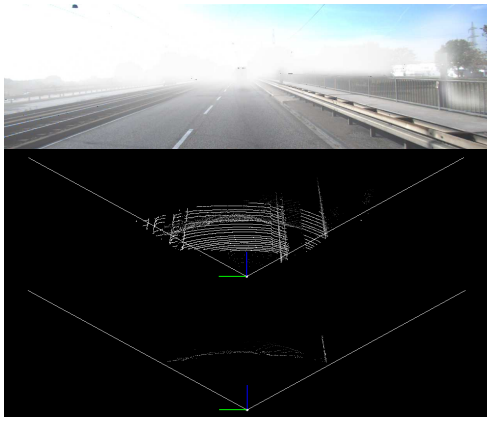
FIGURE 3.10: **From KITTI to Multifog KITTI dataset – example 2.** An example of KITTI dataset includes the left color image, the 64-beam point cloud, the extracted 4-beam point cloud and the corresponding one with the foggy augmentation (visibility V = 45 m). The point cloud is colored according to RGB image.

distorted in foggy weather conditions. This dataset will be used in experiments in Chapter 5.

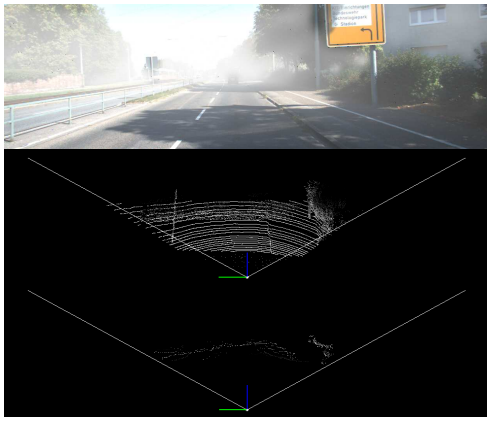


(a) Visibility V = 45 m

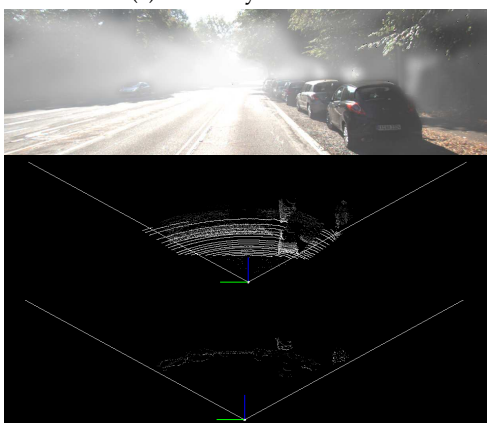
(b) Visibility V = 43 m



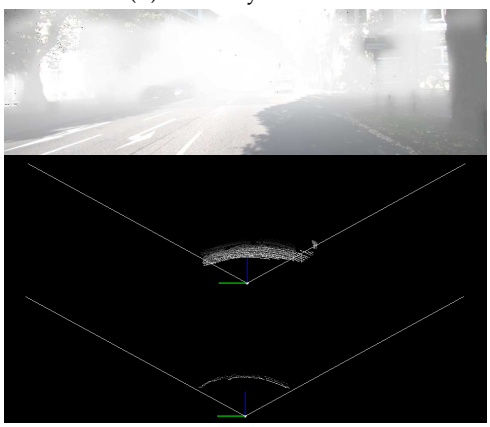
(c) Visibility V = 60 m



(d) Visibility V = 75 m



(e) Visibility V = 65 m



(f) Visibility V = 30 m

FIGURE 3.11: **Examples from our proposed Multifog KITTI dataset.** This figure shows some examples from our proposed Multifog KITTI dataset including the left color image, 64-beam point cloud and extracted 4-beam point cloud with foggy augmentation based on the KITTI dataset. Each example has its own visibility V.

Chapter 4

Sensor Fusion for 3D Object Detection

Contents

Chapter overview: Detecting objects in 3D and determining their distance from the ego-vehicle is a complex task in 2D and 3D object detection. Many studies have highlighted the performance gap between camera-based and LiDAR-based 3D object detection methods due to the latter's high cost but ability to offer accurate and precise depth information. Despite the extensive research on fusing a single RGB camera and LiDAR, no study has investigated the fusion of stereo cameras and LiDAR in a deep neural network for 3D object detection. To improve depth estimation and enhance the performance of 3D object detection, we introduce a novel method called SLS-Fusion, which fuses data from a stereo camera and a low-cost

4-beam LiDAR through a neural network. Since the 4-beam LiDAR is less expensive than the commonly used 64-beam LiDAR, this approach is categorized as a low-cost sensors-based method. The evaluation on the KITTI benchmark demonstrates that our proposed method significantly improves depth estimation performance compared to the baseline method. Furthermore, it advances the state-of-the-art in low-cost sensors-based methods when applied to 3D object detection.

4.1 Introduction

In this chapter, we mainly describe the proposed fusion-based 3D object detection method. We present SLS-Fusion architecture [163], a novel approach fusing LiDAR and stereo camera together in a deep neural network with the final objective of 3D object detection. Based on DeepLiDAR [164] as our backbone network and the pseudo-LiDAR pipeline [43], we present SLS-Fusion architecture [163] which explores several novel ideas to improve 3D object detection performance. The first part of our method, SLS-Fusion depth estimation network, estimates the depth maps from a stereo camera and the projected LiDAR depth maps. This step aims to enrich the feature maps and thus leads to get a better predicted depth map. Then in the second part, the predicted depth map is converted into a pseudo point cloud by using the calibration information between the LiDAR and the camera. In order to improve the pseudo point cloud accuracy, the point cloud is corrected by the real 4-beam point cloud. The last part of SLS-Fusion architecture is LiDAR-based 3D object detection. From the obtained pseudo point cloud, we can apply any LiDAR-based 3D object detection method. In this work, we use PointRCNN [64] for the 3D object detection task. We evaluate our method on the KITTI dataset, a 3D object detection benchmark. Our results show that SLS-Fusion has superior performance to state-of-the-art methods that fuse stereo camera and low-cost 4-beam LiDAR.

4.2 Contribution

Accurately detecting obstacles in their surroundings is key to safe driving of an autonomous vehicle. The current top-performance methods in 3D object detection [64, 66, 165] are mainly based on LiDAR technology. Alternatively, systems based only on camera sensors also received much attention because of their low cost and wide range of use [31, 43, 46, 166–169]. Even though much research has focused on camera-based methods, the gap in performance between monocular or stereo-based methods and LiDAR-based methods is still significant, meriting more research.

The work reported here aims at designing a 3D object detection architecture from low-cost sensors because we are interested in the practical applications of using

them in self-driving vehicles like an autonomous bus. In recent years, a new interesting and promising branch of research proposed by You et al. [46], namely Pseudo-LiDAR, generates the pseudo point cloud based on the estimated depth map from images using the pinhole camera model. Then, they can be treated as a LiDAR signal input for any LiDAR-based 3D object detector. This method has shown a significant improvement in performance compared to previous camera-based object detection methods with this simple data conversion to leverage the performance of the state-of-the-art LiDAR-based method.

Real LiDAR sensors can provide depth information with high accuracy in the form of 3D points. However, these point clouds are sparse compared to continuous surfaces in 3D. Hence, point density is normally uneven, and it causes difficulties in detecting obstacles. Furthermore, in denser point clouds obtained by the Pseudo-LiDAR pipeline [43], distant or small objects such as pedestrians and cyclists, for which LiDAR sensors usually generate fewer points, can get more points and so be more easily detected and localized if the predicted depth map from monocular or stereo-based method is good enough. Finally, another advantage of Pseudo-LiDAR is that it can combine the state of the art of depth estimation and of 3D object detection. Therefore, the performance for 3D object detection of this kind of methods based on this idea strongly depends on the estimated depth map.

Instead of using expensive 64 laser beams which cost about $75000, You et al. [46] simulate the 125 times cheaper 4-beam LiDAR Scala by sparsifying the original 64-beam signals and use these points to correct their estimated depth map, obtained by the Stereo Depth Net (SDN) neural network [46], by applying a graph-based depth correction (GDC).

Starting from this work, a neural network is proposed here that takes 4-beam LiDAR and stereo images as input. The new method estimates a dense depth map and then uses it for the 3D object detection task. Unlike You et al. [46], taking stereo images to estimate the depth map and then integrating the 4-beam LiDAR as a post-processing step to correct the estimated depth map, this approach fuses the 4-beam LiDAR with stereo images into a deep neural network to obtain a more accurate depth map.

To summarize, the main contributions of this chapter are as follows:

- A Sparse LiDAR and Stereo Fusion network (SLS-Fusion) is proposed, which is the first method fusing LiDAR and stereo together in a deep neural network to focus on 3D object detection for autonomous vehicles.
- By integrating the GDC proposed by You et al. [46] with the proposed SLS-Fusion, experimental results on the validation KITTI object detection dataset demonstrate that the proposed approach can produce very accurate depth maps thanks to LiDAR-Stereo fusion. This outperforms all previous low-cost based methods for 3D object detection task.

4.3 Proposed Architecture

The main idea behind the SLS-Fusion architecture is to take into account both the image and the point cloud as inputs to obtain feature-richness in the depth estimation neural network. This leads to better depth map predictions. Figure 4.1 shows the overview of the proposed 3D object detection architecture.

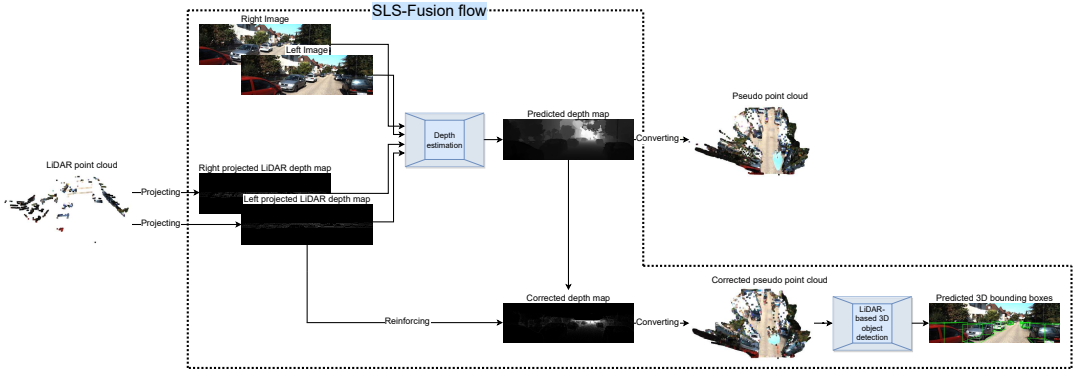


FIGURE 4.1: **Overview of the SLS-Fusion architecture.** It takes both the image and the point cloud as input and outputs 3D bounding boxes.

SLS-Fusion is structured to utilize the advantage of both stereo image and point cloud, those containing complementary information for 3D object detection. On the one hand, 2D image with high resolution provides reliable objectness information. On the other hand, point cloud provides exact spatial information of the objects. We design our model based on these characteristics.

4.4 Method

4.4.1 Sparse Projected LiDAR Depth Map

We would like to process data in 2D, additional information from LiDAR can be gathered by projecting point cloud onto the left and the right coordinate systems, as shown in Figure 4.2, to construct a sparse depth map representation using the corresponding LiDAR range value (z). The sparsity depends on the number of LiDAR beams that map to pixels. Sparse depth maps are convenient and accurate range data as compared to predicted depth map from a camera or a stereo camera.

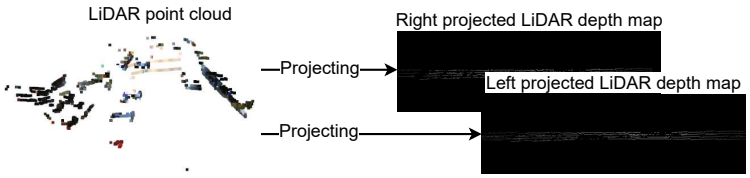


FIGURE 4.2: **LiDAR point cloud preprocessing.** The point cloud is projected in both left and right camera coordinates. This will generate the right projected LiDAR depth map and the left projected LiDAR depth map to be taken as inputs for our model.

4.4.2 Depth Map Estimation

To enrich the representation of a normal stereo matching network, a decision has been taken to join the geometry information from the LiDAR point cloud. However, instead of directly using a 3D point cloud from LiDAR, like in [170], the 4-beam LiDAR point cloud is projected to both left and right image coordinates using the calibration parameters to obtain the two sparse 4-beam LiDAR depth maps corresponding to stereo images.

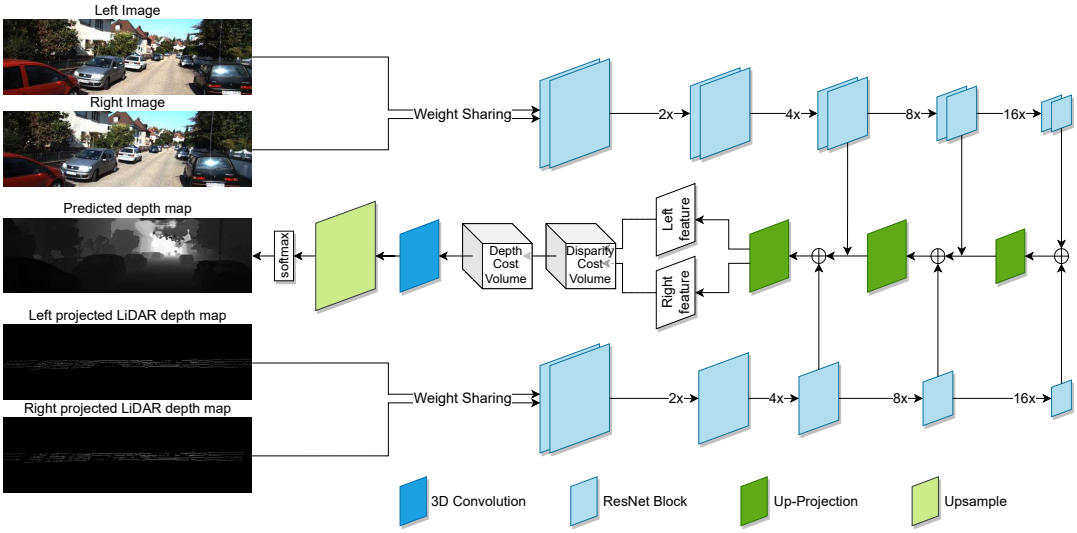


FIGURE 4.3: **Depth estimation part of SLS-Fusion architecture [163, Fig. 2].** The left and right input stereo images, I_l and I_r, are fed to two weight-sharing pipelines consisting of ResNet blocks to encode features shown on the top flow (the same way for the left and right projected 4-beam LiDAR images, S_l and S_r, on the bottom flow). In the decoding phase, each LiDAR and image pair are fused, (S_l, I_l) and (S_r, I_r) and corresponding feature tensors are obtained, left feature and right feature, for each LiDAR-image input pair. Then left and right feature tensors go through a Disparity Cost Volume (DiCV) and then a Depth Cost Volume (DeCV) to directly learn the depth.

Our depth estimation part is shown in Figure 4.3. The proposed method uses as input stereo images and projected LiDAR depth maps. However, we use a late fusion approach, presented as follows, instead of using a simple early fusion paradigm by concating stereo image and the corresponding sparse projection LiDAR depth map as.

Encoder-decoder feature extraction – Figures 4.4 and 4.5 show the left (or right) branch, including left (right) image and left (right) projected LiDAR depth map, and its corresponding diagram showing the sequence of Resnet blocks.

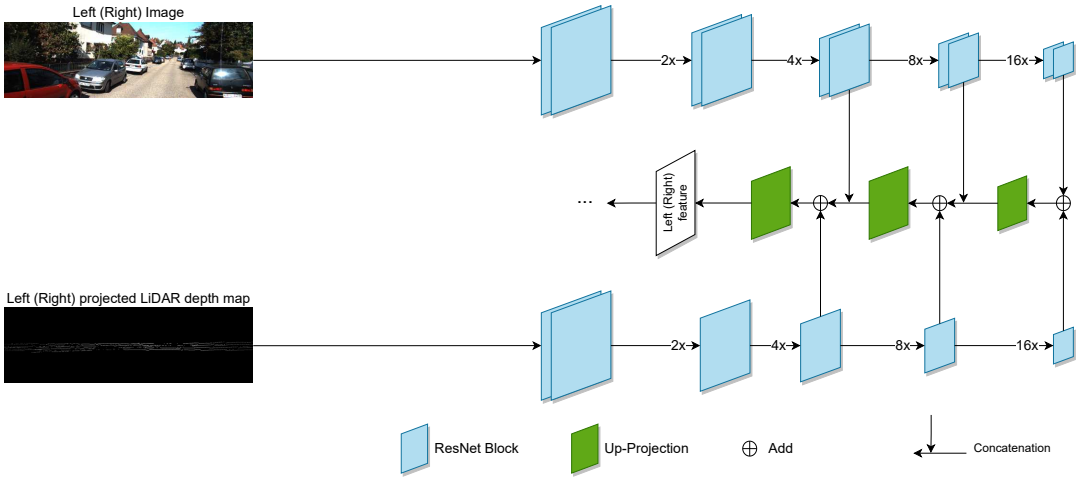


FIGURE 4.4: **Feature extraction of SLS-Fusion architecture.**

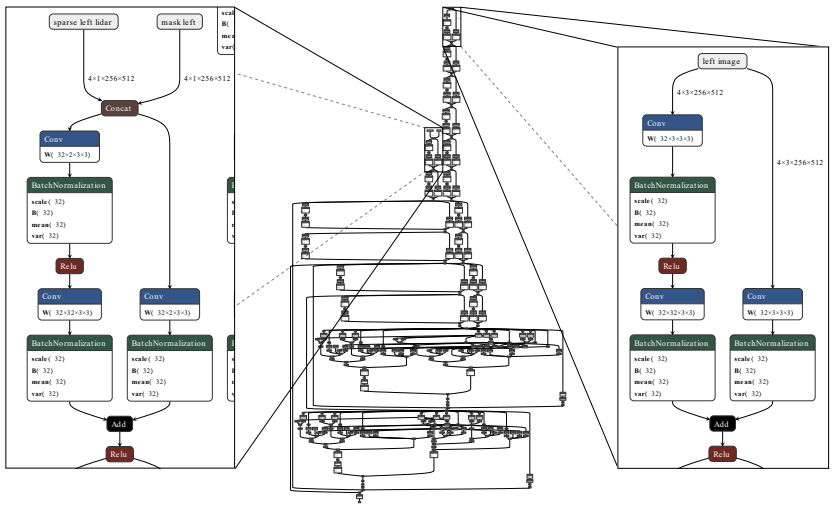


FIGURE 4.5: **Sequence of Resnet blocks from SLS-Fusion model** represented for a left image and left projected LiDAR branch in feature extraction part of our method.

Disparity Cost Volume – A pair of left-right images I_l and I_r, taken from two cameras with a horizontal offset (i.e., baseline), are inputted into a disparity estimation algorithm. Without losing generality, we suppose that the method creates a disparity map d that records the horizontal disparity to I_r for each pixel and uses the left image, I_l, as reference (u, v). $I_l(u, v)$ and $I_r(u, v + d(u, v))$ should ideally depict the same 3D space. Therefore, using the following transformations, we can extract the depth map $D(u, v)$ corresponding to pixel (u, v) as follows:

$$D(u, v) = \frac{f_u b}{d(u, v)}, \tag{4.1}$$

where f_u and b are the horizontal focal length of the left camera and the horizontal offset (baseline) of the stereo camera, respectively.

Building a 4D disparity cost volume C_d, in which $C_d(u, v, d, :)$ is a feature tensor that captures the pixel difference between $I_l(u, v)$ and $I_r(u, v + d(u, v))$, is a frequent step in the disparity estimation process. Then, using the cost volume C_d, it calculates the disparity $d(u, v)$ for each pixel (u, v). Calculating a 3D cost volume is as follows:

$$C_d(u, v, d) = ||I_l(u, v) - I_r(u, v + d(u, v))||_2, \tag{4.2}$$

and selecting $d(u, v)$ as an argument for $C_d(u, v, d)$ is a simple algorithm. Later methods [43, 46, 47] create C_d using more reliable features and do structured prediction on D.

Similar to You et al. [46] and Chang and Chen [47], we start by extracting deep feature maps, left feature F_l and right feature F_r, from I_l and I_r, respectively. But the difference is that our model extracts more information from LiDAR instead of just using the stereo camera. From here it is expected that the strong properties of LiDAR can be useful and increase the results of the model. It then constructs $T_d(u, v, d)$ by concatenating features of $F_l(u, v)$ and $F_r(u, v + d)$, followed by layers of 3D convolutions. The resulting 3D tensor T_d is then used to calculate the pixel disparity using the weighted combination shown below, with the feature channel size ending up being one:

$$d(u, v) = \sum_d \text{softmax}(-T_d(u, v, d)) \times d, \tag{4.3}$$

where softmax is applied along T_d third dimension. To reduce the disparity error, Chang and Chen [47] proposed a method can be trained from beginning to end, including the image feature extractor and 3D convolution kernels and try to optimize this loss:

$$L(d, \hat{d}) = \sum_{(u,v \in A)} |d(u,v) - \hat{d}(u,v)|, \tag{4.4}$$

where $L(d, \hat{d})$ is the smooth L1 loss, $\hat{d}$ is the ground truth disparity map, and A contains pixels with valid ground truths.

Depth Cost Volume – You et al. [46] pointed out that learning disparity firstly instead of directly learning depth information leads to a suboptimal problem. It may cause a large depth error even if the disparity error is very small. The 3D convolutions within the 4D disparity cost volume, where the same kernels are applied for the whole cost volume, are a significant source of mistakes. This is really troublesome because it implies that a convolution effect is uniform throughout. Equation (4.1) shows that depth D is inversely proportional to disparity d:

$$D \propto \frac{1}{d}. \tag{4.5}$$

Consequently,

$$\begin{aligned} \partial D &\propto \frac{1}{d^2} \partial d \\ &\propto D^2 \partial d. \end{aligned} \tag{4.6}$$

Equation (4.6) shows that the depth error ∂D increases quadratically with the disparity error ∂d.

Similar to You et al. [46], we use the depth cost volume C_D, in which $C_D(u, v, z)$ will encode features describing how likely the depth of the pixel (u, v) is z. This insight and the fundamental tenet of the convolutions that all neighborhoods can be operated upon in the same way are taken into consideration. The next 3D convolutions will thus operate on the grid of depth rather than disparity, having a similar impact on nearby depths regardless of where they are. In order to estimate the pixel depth $D(u, v)$ similarly to Equation (4.3), the generated 3D tensor T_D is employed as follows:

$$D(u,v) = \sum_z \text{softmax}\left(-T_D(u,v,z)\right) \times z. \tag{4.7}$$

4.4.3 Depth Correction

Our predicted depth map provides a denser information than 64-beam LiDAR. However, it cannot be as accurate as the real point cloud from LiDAR. Because of the discrete nature of pixels, it has restrictions. The disparity, which is the difference in the horizontal coordinate between matching pixels, is generally quantized at the

level of individual pixels, whereas the depth is continuous. While high-resolution images can reduce quantization error, the computational cost grows as the resolution increases, so GPU can be overworked in autonomous vehicles.

You et al. [46] attempted to remedy this bias using a low-cost LiDAR that takes precise depth measurements with extremely sparse but accurate beams (e.g., 4-beam). They presented a graph-based depth correction (GDC) algorithm that successfully blends sparse accurate LiDAR observations with dense stereo depth that has generated object forms. The rectified depth map should theoretically contain the following characteristics: locally, item forms acquired by nearby 3D points, back-projected from the input depth map as provided in Equation (4.10), should be preserved. Globally, landmark pixels associated with LiDAR points should have the precise depths.

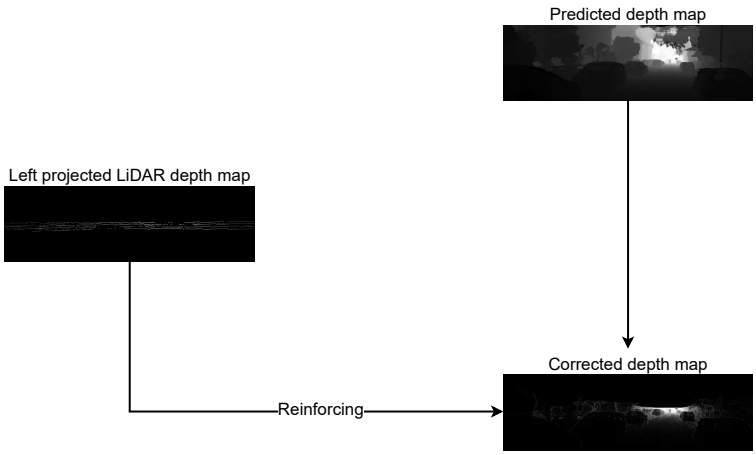


FIGURE 4.6: **Depth correction part.**

The depth correction flow is shown in Figure 4.6. The design process of GDC involves learning the depth relationship among neighbors, and then constraining the changes in depth based on the relationship. Following GDC, we take as input the projected depth map from LiDAR (L) and predicted depth map (PL) obtained by our depth estimation neural network. First, we characterize the local shapes by the directed K-nearest neighbor (KNN) graph in the pseudo point cloud (using accelerated KD-trees [171]) that connects each 3D point to its KNNs with appropriate weights. Here, we match the projection LiDAR depth map to corresponding predicted depth map pixel. Assuming that ground truth LiDAR depth map for the first n points, but no ground truth for the remaining m points, without sacrificing generality. The predicted depth estimations are expressed as $D \in \mathbb{R}^{n+m}$ and the LiDAR depth ground truth as $\hat{DL} \in \mathbb{R}^{n}$.

The KNN graph in 3D is constructed by ignoring the LiDAR information on the first

n points and using only the predicted depth in D. Let N_i denote the set of K neighbors of the i^{th} point. Let $W \in \mathbb{R}^{(n+m)\times(n+m)}$ denote the weight matrix, where W_{ij} denotes the edge-weight between points i and j. Weights are used to reconstruct any point's depth from its neighbors's depths in N_i. The following constrained quadratic optimization problem can be used to solve for these weights:

$$W = \text{argmin}_W \|D - WD\|_2^2, \text{ s.t. } W1 = 1 \text{ and } W_{ij} = 0 \text{ if } j \notin N_i. \tag{4.8}$$

$1 \in \mathbb{R}^{n+m}$ denotes the all-ones vector. It is infinitely possible solutions that satisfy $D = WD$ when they use a $k > 3$ and the points are positioned generally, the L2 norm is used as a solution.

They denote the corrected depth values as $D^{'} \in \mathbb{R}^{n+m}$, with $D^{'} = [D_L^{'}; D_{PL}^{'}]$ and $D_L^{'} \in \mathbb{R}^n$ and $D_{PL}^{'} \in R^m$, where $D_L^{'}$ are the depth values of points with LiDAR ground-truth and $Z_{PL}^{\prime}$ otherwise. For the n points with LiDAR measurements we update the depth to the (ground truth) values $D_L^{'} = DL$. We then solve for $D_{PL}^{'}$ given DL and the weighted KNN graph encoded in W. Concretely, we update the remaining depths $D_{PL}^{'}$ such that the depth of any point i can still be reconstructed with high fidelity as a weighted sum of its KNN's depths using the learned weights W; i.e. if point $i : 1 \leq i \leq n$ is moved to its new depth DL_i, then its neighbors in N_i must also be corrected such that $DL_i \approx \sum_{j \in N_i} W_{ij} D_j^{'}$. Further, the neighbors of the neighbors must be corrected and the depth of the few n points propagates across the entire graph. The final $D^{'}$ ca be obtained by optimizing:

$$D' = \text{argmin}_{D'} \left\|D' - WD'\right\|_2^2. \tag{4.9}$$

4.4.4 Corrected Depth Map Conversion into Corrected Pseudo Point Cloud

Once the predicted depth map D is obtained, with $D(u, v)$ being the depth corresponding to each image pixel (u, v), a pseudo point cloud can be generated by using the pinhole camera model. It is illustrated in Figure 4.7.

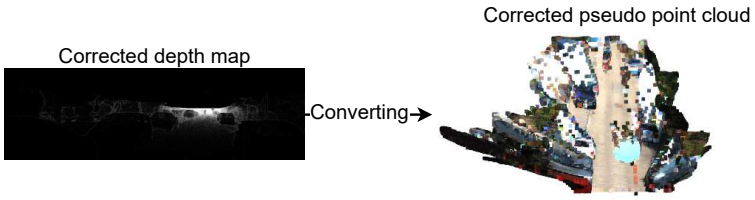


FIGURE 4.7: **Depth map to pseudo point cloud transformation.** The predicted depth map D is converted into pseudo point cloud. Depth map image represents the depth value for each pixel. Objects that are far away are brighter, and objects that are near are darker. For the pseudo point cloud, the points are colored by RGB image for visualization only.

Given the estimated depth map D and camera intrinsic matrix, deriving the 3D location (X_c, Y_c, Z_c) in the camera coordinate system for each pixel (u, v) is simply as:

$$\text{(depth) } Z_c = D(u, v), \tag{4.10a}$$

$$\text{(width) } X_c = \frac{(u - c_u) \times Z_c}{f_u} = \frac{(u - c_u) \times D(u, v)}{f_u}, \tag{4.10b}$$

$$\text{(height) } Y_c = \frac{(v - c_v) \times Z_c}{f_v} = \frac{(v - c_v) \times D(u, v)}{f_v}, \tag{4.10c}$$

where c_u and c_v is the pixel location corresponding to the projection of the camera center (principal point); f_u and f_v are the focal length in pixel widths and the focal length in pixel heights. Then this point is transformed into (X_l, Y_l, Z_l) in the LiDAR coordinate system (the real world coordinate system). Given the camera extrinsic matrix:

$$C = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix}, \tag{4.11}$$

where R and t are respectively the rotation matrix and translation vector, the relationship between (X_c, Y_c, Z_c) and (X_l, Y_l, Z_l) is as follows:

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = C \begin{bmatrix} X_l \\ Y_l \\ Z_l \\ 1 \end{bmatrix}, \tag{4.12}$$

and then the corresponding pseudo point in the LiDAR coordinate system can be obtained by computing:

$$\begin{bmatrix} X_l \\ Y_l \\ Z_l \\ 1 \end{bmatrix} = C^{-1} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}. \tag{4.13}$$

To make the pseudo point cloud more similar to real LiDAR signals, following Wang et al. [43] and Yang, Luo, and Urtasun [172], reflectance is set to 1 for each point and points higher than 3 m above the ground are removed. Afterwards, LiDAR-based 3D object detector can be applied to this pseudo point cloud.

4.4.5 3D Object Detection on Pseudo Point Cloud

Figure 4.8 shows the final part of the proposed architecture, which is LiDAR-based 3D object detection.

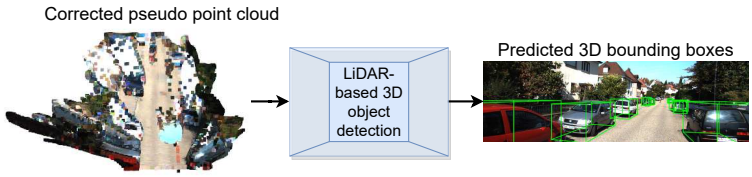


FIGURE 4.8: **3D object detection part of SLS-Fusion architecture.** The corrected pseudo point cloud can be used directly as the real point cloud. We can apply any LiDAR-based method to it.

The purpose of this part as well as the conversion of data from the predicted depth map to pseudo point cloud is to take advantage of the efficiency of the LiDAR-based 3D object detection methods, which often outperform other methods. A LiDAR-based 3D object detection method can be applied as an add-on part to the predicted pseudo point cloud.

4.5 Experiment Settings

4.5.1 Metrics

To evaluate our approach, we evaluate both the depth estimation part and the final result of 3D object detection.

We first evaluate the depth estimation part on the KITTI object detection dataset. The evaluation calculates four metrics: the root mean squared error (RMSE in mm), the mean absolute error (MAE in mm), the root mean squared error of the inverse depth (iRMSE in km^{-1}) and the mean absolute error of the inverse depth (iMAE in km^{-1}). Following the KITTI depth completion benchmark, the root mean squared error (RMSE) is used as the main metric. These metrics are defined as follows:

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{(u,v)\in A} \left(D(u,v) - \hat{D}(u,v)\right)^2}, \tag{4.14}$$

$$\text{MAE} = \frac{1}{n} \sum_{(u,v)\in A} \left|D(u,v) - \hat{D}(u,v)\right|, \tag{4.15}$$

$$\text{iRMSE} = \sqrt{\frac{1}{n} \sum_{(u,v)\in A} \left(\frac{1}{D_{(u,v)}} - \frac{1}{\hat{D}(u,v)}\right)^2}, \tag{4.16}$$

$$\text{MAE} = \frac{1}{n} \sum_{(u,v)\in A} \left|\frac{1}{D(u,v)} - \frac{1}{\hat{D}(u,v)}\right|, \tag{4.17}$$

where $\hat{D}(u,v)$ is taken from the accumulated 11 frames of the 64-beam LiDAR point cloud presented in Section 3.2.2 and used as the ground truth depth map, A contains pixels with ground truths and n is the number of valid pixels in A. A comparison is made in experiments with the Stereo Depth Network (SDN), which is the depth network of Pseudo-LiDAR++ [46] to test the advantage of LiDAR features in the deep neural network.

For the evaluation of object detection, we report the average precision (AP) with an intersection over union (IoU) thresholds at 0.5 and 0.7, as per normal practice. With B and b denoting the area of predicted and ground truth 3D bounding boxes, respectively, IoU is defined as follows:

$$\text{IoU} = \frac{B \cap b}{B \cup b}. \tag{4.18}$$

We denote 3D AP and bird's eye view (BEV) AP of the object detection as AP_{3D} and AP_{BEV}, respectively. AP is calculated according to the formula below:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{4.19}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{4.20}$$

$$AP = \frac{1}{11} \sum_{\text{Recall}_i} \text{Precision}\left(\text{Recall}_i\right), \tag{4.21}$$

where TP, FP, FN are the numbers of true positives, false positives and false negatives, respectively, determined based on a given threshold IoU described above.

Objects in the KITTI dataset are divided into three levels of difficulty: Easy, Moderate and Hard, depending on their 2D bounding box sizes, occlusion, and truncation extent following the KITTI definitions. Similar to Li, Ku, and Waslander [169], experiments are compared to six recent stereo-based detectors: TLNet [167], Stereo-RCNN [166], DSGN [168], CG-Stereo (with PointRCNN) [169], Pseudo-LiDAR (with PointRCNN) [43], the baseline method Pseudo-LiDAR++ (with PointRCNN) [46] and also the original LiDAR-based detector PointRCNN [64]. But unlike Li, Ku, and Waslander [169], we do not compare with the result of OC-Stereo [173] which used the AVOD [44] detector while the proposed method uses the PointRCNN [64] detector. Here, CG-Stereo (with PointRCNN) means the result from an experiment where CG-Stereo uses PointRCNN in their experiment and the same for other cases.

4.5.2 Dataset

We conduct our experiments in the KITTI dataset as introduced in Chapter 3.2, the 3D object detection benchmark. It provides synchronized 2D images and 3D LiDAR point cloud with annotations for car, pedestrian, and cyclist class. We focus on the car class because it contains the most training examples and is the main ingredient that appears on the streets. We use the KITTI object detection dataset for training both depth estimation and 3D object detection parts of our method.

4.5.3 Training and Inference

Firstly, we learn our depth estimation part to directly minimize the depth error:

$$L(D, \hat{D}) = \sum_{(u,v) \in A} smooth_{L_1}(D(u,v) - \hat{D}(u,v)), \tag{4.22}$$

where $L(D, \hat{D})$ is the smooth L1 loss, $\hat{D}(u,v)$ is the ground truth depth map, and A contains pixels with valid ground truths.

For faster convergence, full depth maps from Scene Flow dataset [144] are firstly used for training. Here the LiDAR input is set as zeros because this synthetic dataset is designed exclusively for RGB stereo and no LiDAR scans are provided. Then fine-tuning is done on the 3,712 training samples of the KITTI dataset for 100 epochs, with batch size 4 and the learning rate set to 0.001. The Adam optimizer was used to optimize the depth estimation network.

For 3D object detection, PointRCNN [64] is applied, which is a LiDAR-based method with a high performance and used by many other methods as a baseline. As it was designed to take into account a sparse point cloud, the dense point cloud is sub-sampled to 64-beam LiDAR. Then, the released implementation of PointRCNN is

used directly, and its guidelines followed to train it on the training set of the KITTI object detection (KOD) dataset [49] only for the "Car" class because car is one of the main objects and occupies the largest percentage in the KITTI dataset which causes the unbalance between "Car" and other classes.

Both trainings run on 2 NVIDIA GeForce GTX 1080 Ti GPUs with 11G memory.

4.6 Experiment Results

This section firstly presents an evaluation of the depth estimation accuracy of the SLS-Fusion network. The results are shown in Table 4.1. The results show that the proposed network outperforms the baseline Pseudo-LiDAR++ [46] by a large margin over all metrics evaluated on pixels with a valid depth ground truth in the range [1 m, 80 m] on the KOD dataset [49]. They support the idea that adding LiDAR in the stereo network can boost the accuracy of the predicted depth map.

Method	**RMSE** (mm)	MAE (mm)	iRMSE (km^{-1})	iMAE (km^{-1})
baseline∗ [46]	1506.92	443.81	5.59	1.90
ours∗	**845.21**	**307.36**	**2.72**	**1.28**

TABLE 4.1: **Evaluation of the depth estimation part of our method compared to our baseline.** The results show the mean error (shown in different metrics) for a pixel from all the test images of KOD dataset [49]. RMSE (red) is the main metric to rank methods.

Method	Input	Easy	Moderate	Hard
TLNet [167]	S	62.46 / 59.51	45.99 / 43.71	41.92 / 37.99
Stereo-RCNN [166]	S	87.13 / 85.84	74.11 / 66.28	58.93 / 57.24
Pseudo-LiDAR [43]	S	88.4 / 88.0	76.6 / 73.7	69.0 / 67.8
DSGN [168]	S	-	-	-
CG-Stereo [169]	S	**97.04** / 90.58	88.58/ **87.01**	80.34/ 79.76
baseline∗ [46]	S	89.8 / 89.7	83.8 / 78.6	77.5 / 75.1
ours∗	S	**90.13** / **89.9**	**83.89** / **78.81**	**78.03** / **76.14**
baseline∗∗ [46]	S+L4	90.3 / 90.3	87.7 / **86.9**	**84.6**/ **84.2**
ours∗∗	S+L4	**93.16** / **93.02**	**88.81** / 86.19	83.35 / 84.02
PointRCNN [64]	L64	97.3 / 97.3	89.9 / 89.8	89.4 / 89.3

(a) IoU = 0.5

Method	Input	Easy	**Moderate**	Hard
TLNet [167]	S	29.22 / 18.15	21.88 / 14.26	18.83 / 13.72
Stereo-RCNN [166]	S	68.50 / 54.11	48.30 / 36.69	41.47 / 31.07
Pseudo-LiDAR [43]	S	73.4 / 62.3	56.0 / 44.9	52.7 / 41.6
DSGN [168]	S	83.24 / 73.2	63.91 / 54.27	57.83 / 47.71
CG-Stereo [169]	S	87.31 / 76.17	68.69 / 57.82	65.80 / 54.63
baseline∗ [46]	S	82.0 / 67.9	64.0 / 50.1	57.3 / 45.3
ours∗	S	**82.38** / **68.08**	**65.42** / **50.81**	**57.81** / **46.07**
baseline∗∗ [46]	S+L4	**88.2** / 75.1	**76.9** / 63.8	73.4 / **57.4**
ours∗∗	S+L4	87.51 / **76.67**	76.88 / **63.90**	**73.55** / 56.78
PointRCNN [64]	L64	90.2 / 89.2	87.9 / 78.9	85.5 / 77.9

(b) IoU = 0.7

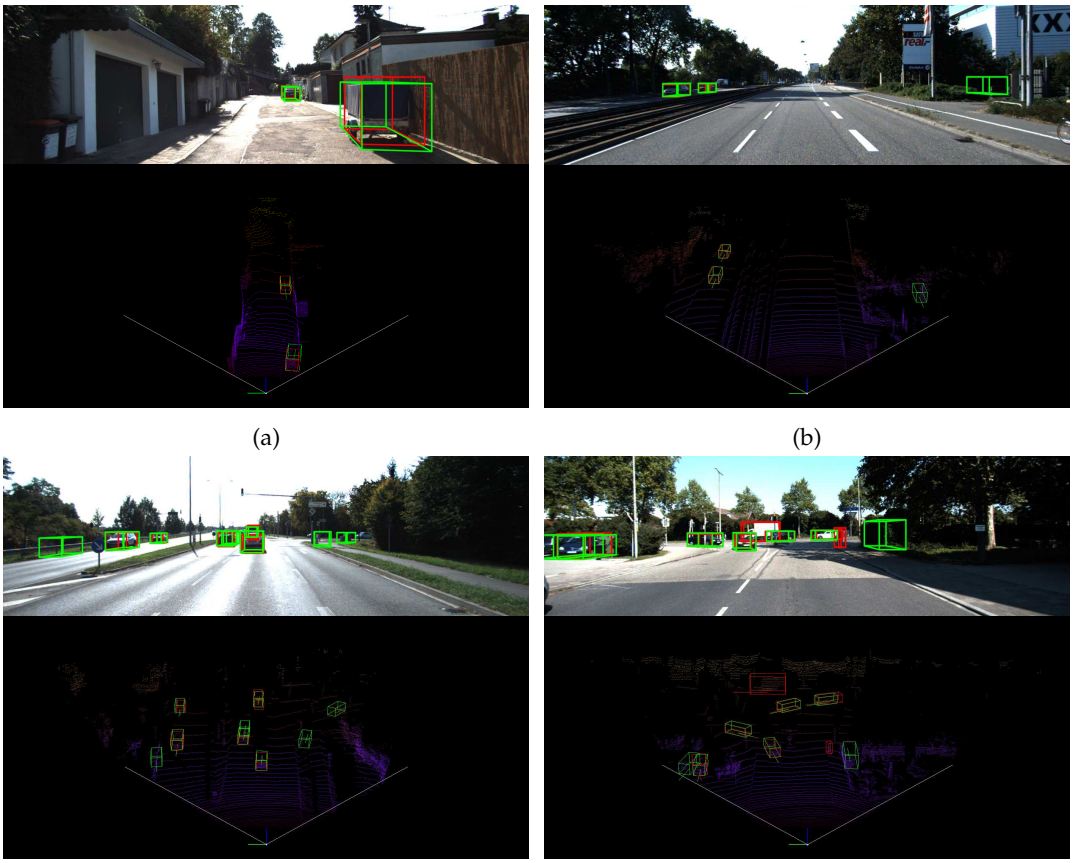
TABLE 4.2: **Evaluation of the 3D object detection part of our method compared to SOTA.** AP_{BEV} / AP_{3D} results on the KITTI validation set for "Car" category with IoU at 0.5 and 0.7 and on three levels of difficulty: Easy, Moderate and Hard. (a) and (b) correspond to results with IoU of 0.5 and IoU of 0.7. The results are evaluated using the original KITTI metrics. All methods are ranked based on the moderately difficult AP_{3D} results (red). S, L4 and L64 respectively denote stereo, simulated 4-beam LiDAR and 64-beam LiDAR inputs. Baseline∗, baseline∗∗, ours∗ and ours∗∗ respectively are Pseudo-LiDAR++ (stereo), Pseudo-LiDAR++ (stereo + 4-beam LiDAR), the proposed method SLS-Fusion (stereo) and SLS-Fusion (stereo + 4-beam LiDAR).

Following the promising results from the predicted map, results for 3D object detection task are reported in Table 4.2, showing two results (one for IoU of 0.5 and another one for IoU of 0.7) for the proposed approach. Ours∗ is the base proposed model, while ours∗∗ includes the GDC proposed in [46]. Pseudo-LiDAR++ is taken as a baseline and in the same way, baseline∗ and baseline∗∗ denote respectively Pseudo-LiDAR++ (S) and Pseudo-LiDAR++ (S+L4). The proposed approach is also compared with six other outstanding methods: Pseudo-LiDAR [43], TLNet [167],

Stereo-RCNN [166], DSGN [168], CG-Stereo [169] which are based on low-cost sensors and the original 3D object detector PointRCNN [64] which uses 64-beam LiDAR. Firstly, comparing ours$*$ with the baseline$*$, the proposed method outperforms the baseline over all indicators, thereby showing the effect of 4-beam LiDAR in the proposed stereo matching model. Secondly, ours$**$ outperforms the baseline$**$ and other stereo-based methods on some indicators. The proposed method gets the best result (63.90%) on 3D AP with IoU of 0.7 which is the main indicator to rank all methods on KITTI. Lastly, compared to the original detector PointRCNN [64] which used 64-beam LiDAR as input, ours$*$ and ours$**$ have inferior performance, which is expected, but it helps to get closer to the method trained on real LiDAR.

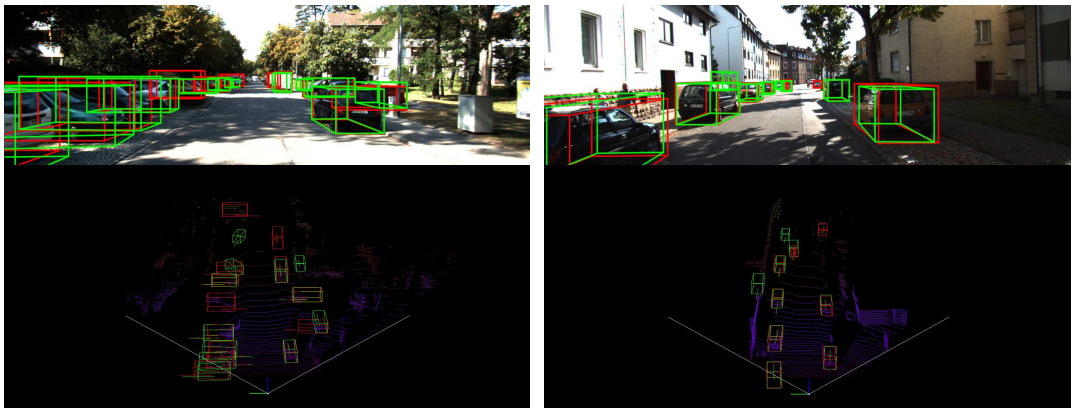
The good results achieved by CG-Stereo [169] on 0.5 IoU (Easy and Moderate) could be explained by the method used. In fact, this latter splits the scene into foreground/background and applies for each part independent depth estimation. This leads to easier detection of Easy objects in the foreground. However, we notice that our method is more stable because it achieves better results on hard objects.

Figure 4.9 shows some detection results obtained by the proposed approach on data from the KITTI validation dataset.

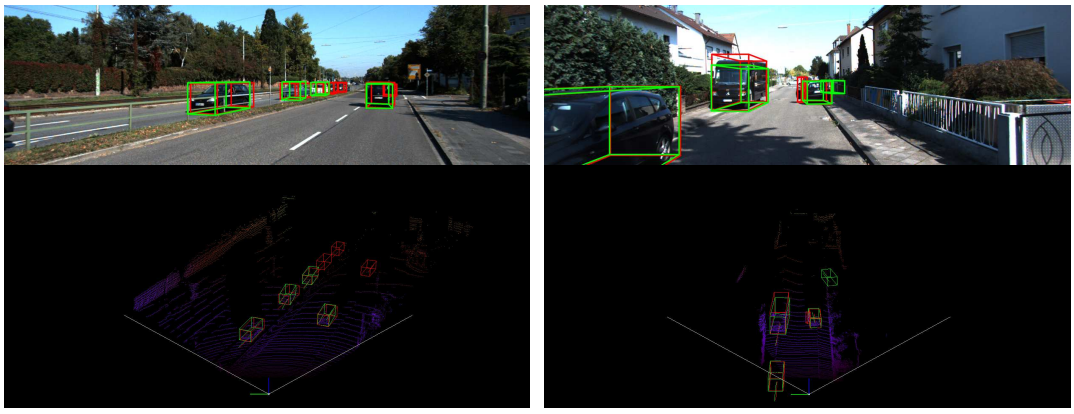


(a) (b)

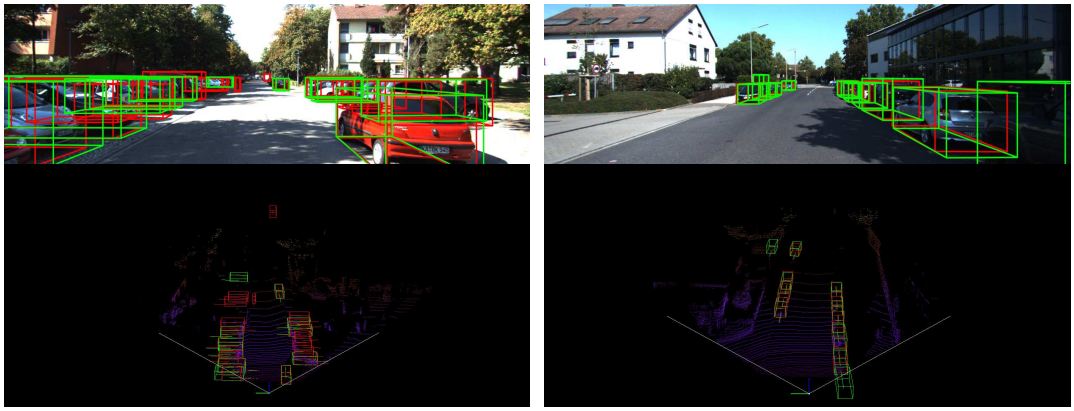
(c) (d)



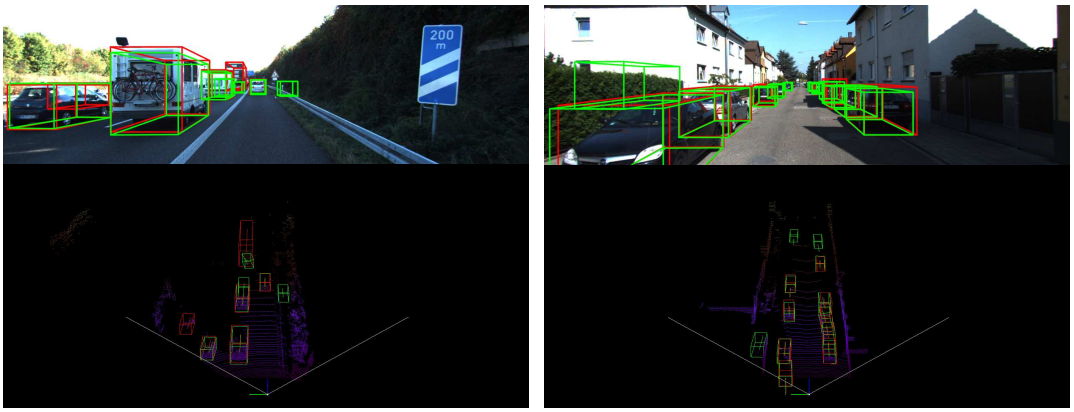
(e) (f)



(g) (h)



(i) (j)



(k) (l)

FIGURE 4.9: **Qualitative Comparison**. We illustrate the results of SLS-Fusion with different input point clouds on a KITTI validation scene. They are visualized in a frontal-view image as well as a BEV point map. Ground-truth and predicted boxes are in red and green, respectively.

4.7 Conclusion

In this chapter, we have introduced the SLS-Fusion network, a novel depth prediction framework taking 4-beam LiDAR and stereo images as input which is low-cost sensors in the autonomous driving domain, for 3D object detection. Our architecture can take advantage of features from both images and point cloud by using an encoder-decoder network and then optimize the depth estimation loss via a Depth Cost Volume. The experiments here are performed on the KITTI dataset, where data is collected under conditions unaffected by bad weather. Experimental results demonstrate that this model by fusing a stereo camera and 4-beam LiDAR can improve both depth estimation and 3D object detection accuracy. Experiments under extreme weather conditions will be presented in the next chapter.

Chapter 5

Sensor Fusion for 3D Object Detection in Foggy Weather Conditions

Contents

Chapter overview: The objective of this chapter is to evaluate a fusion solution that uses cameras and LiDAR (4-beam and 64-beam) for 3D object detection in foggy weather conditions. Unlike the previous chapter, where we did not take into account the effect of weather, this chapter analyzes and evaluates the impact of fog on the sensors and their ability to identify the environment. As part of this analysis, we also examine the contribution of each sensor to the results because the SLS-Fusion model fuses data from both input sensors through deep fusion, and we need to understand how well each one contributes. Firstly, it should be noted that the performance of 3D object detection is significantly reduced when the data is trained on fog-free conditions and evaluated on foggy data. This highlights the importance of

selecting appropriate training datasets. For instance, when evaluating Moderate objects in foggy weather conditions, the performance of 3D object detection is reduced by 42.67% if trained on the KITTI dataset without fog. However, by using a specific training strategy that includes both datasets with and without fog (i.e., training on the KITTI and the Multifog KITTI), the results are significantly improved by 26.72%. Moreover, the detection still performs well on the original dataset (Clear KITTI) with only an 8.23% drop compared to training and testing on the Clear KITTI dataset. In summary, fog can cause significant challenges for 3D object detection in driving scenes. By training with an augmented dataset, we can significantly improve the performance of the proposed 3D object detection algorithm for self-driving cars in foggy weather conditions.

5.1 Introduction

In the previous chapter, we conducted an analysis for the detection of 3D objects by simultaneously involving cameras and LiDAR. These tests were carried out under normal weather conditions. In this chapter, it is a question of carrying out the same analysis, but on data tainted with fog. The main objective is to verify the behavior of our model when the data contains fog.

(a) Original sample (b) Foggy sample

FIGURE 5.1: **Example of image distortion and the corresponding point cloud in the proposed Multifog KITTI dataset [162, 174, Fig. 1]** Note that the color of each point in the point cloud is the color of the corresponding pixel in the image (for visualization only). (a) shows the original sample from the KITTI dataset [12] with the image above and the point cloud below collected by Velodyne HDL64 S2 LiDAR [12]. (b) shows the image and the point cloud simulated in foggy conditions with visibility V equal to 52 m from the proposed dataset. Arising from receiving back-scattered light from water drops, fog makes the image have lower contrast and causes incorrect point measurements in the point cloud.

Many factors can affect the perception ability of self-driving cars sensors, thereby potentially causing serious consequences on the lives of other road users. While

applications using these sensors do quite well in controlled lighting or indoor environments unaffected by weather, outdoor applications face many problems. For example, applications that use a camera to perceive the environment often fail in extreme lighting conditions such as sunburns, low light, or nighttime conditions. Significant challenges arise in self-driving cars under adverse weather conditions such as fog, rain, or snow, in which both the camera and LiDAR are severely affected Mai et al. [175], as shown in Figure 5.1. In particular, fog is a common phenomenon and hinders self-driving cars and even drivers. Therefore, the purpose of this chapter is to analyze the effects of fog on the detection of objects in driving scenes and then to propose methods for improvement.

In reality, it is not always easy to obtain data containing fog. This is all the more important depending on the place of experimentation. There are countries where there is never fog. It is therefore not easy to have complete datasets in foggy weather conditions. A solution to overcome this drawback is to generate synthetic data containing fog. It involves applying a number of basic treatments to the original data to taint it with fog. In terms of processing tasks, we test our 3D object detector based on stereo camera and LiDAR (SLS-Fusion) presented in the previous chapter to see how it is affected by foggy weather conditions. We propose to train it using both the original dataset and the augmented dataset to improve performance in foggy weather conditions while keeping good performance under normal conditions.

5.2 Contribution

Methods that produce the best results are mostly based on deep learning architectures by training a model with large amounts of data associated with labeling (supervised learning). While labeling and collecting data in good conditions such as daytime or sunny weather takes time, it takes even more time and effort in extreme weather. Thus, there is an imbalance in the amount of data recorded in extreme weather conditions compared to those in normal conditions [88, 123, 124, 131, 148]. In addition, there may be an imbalance in different levels of fog, rain, or snow when collecting data. Most of the data is only collected during a given time and place, so the data will not be able to fully cover all situations, and therefore using a model trained with a limited range of data may make mistakes. So, in addition to real data, the creation of synthetic data which can be simulated under many controlled parameters is equally important. Chapter 3 introduced a physics-based fog data on a well-known dataset collected in the daytime and mild sunlight conditions to provide improvements under foggy conditions.

Some studies also identify failures and how to fix them in extreme weather conditions [83, 98, 100–102]. However, most of them only experiment on images and not on both images and point cloud [83]. To improve the performance of the perception algorithms, several studies suggest to dehaze fog, rain, or snow on images [98,

176, 177] and more recently for point cloud [178]. Others pointed out how the combination of LiDAR and cameras also affects performance [83, 101]. Other studies proposed to increase the amount of data during the training phase [98–100, 102].

This work extends our work [163] presented in Chapter 3, a 3D object detector for self-driving cars in normal weather conditions, to run in foggy conditions. We point out that the late-fusion-based architecture can perform well with a justifiable training strategy in foggy weather conditions. The main contributions are:

- Firstly, we propose a new public dataset augmenting the KITTI dataset [12] for both images and point cloud through different visibility ranges (in fog) from 20 to 80 meters to be as close as possible to a realistic foggy environment.
- Secondly, we show that the data collected from the camera and LiDAR is significantly distorted under foggy scenes. It directly affects the performance of self-driving cars 3D object detection algorithm, as confirmed through our experiments.
- Finally, extending from our initial work [163] (described in the previous chapter) on the original dataset [12] (good weather condition dataset), we propose a specific training strategy which uses both normal and foggy weather datasets as training datasets. Experiments show that the model can run better in foggy weather conditions while keeping performance close to that obtained in normal weather conditions.

Additionally, since the data from the two input sensors are fused in the early stages of the SLS-Fusion model, we also analyze the contribution of each sensor to the model. For this:

- We divide and adapt the SLS-Fusion neural network [163] to take into account the stereo camera or LiDAR separately. This leads to two different neural subnetworks.
- We study the performance of each neural subnetwok in foggy weather conditions in comparison with SLS-Fusion.
- We then analyze the performance of the 3D object detection models (for the stereo camera and LiDAR) according to six levels of fog applied to the KITTI dataset.

5.3 Impact of Fog on Camera and LiDAR

Fog is the phenomenon of water vapor condensing in tiny cloud-like particles that appear on and near the ground instead of in the sky. The Earth's moisture slowly evaporates, and when it does, it moves up, cools, and condenses to form fog. Fog can be seen as a form of low clouds.

Physically, fog causes dispersion. Before entering the camera sensor, light is dispersed by the suspended water droplets. Two main outcomes result from this scattering phenomenon. Both a signal floor of scattered light and the attenuation of the chief ray before it enters the sensor are present. These effects, reduce contrast, as shown in Figure 5.3, the intensity range is filled with intensity values (a single value for a gray-level image for each pixel) that decrease with the intensity of the fog (clear, 80, 50, 20 m). Therefore, the contrast of the image is inversely proportional to fog density, and it may cause driving difficulties for both human drivers and sensor-based autonomous systems or driving aids.

Meanwhile, even point clouds are more affected because real object points are not recognized, but instead are water mist droplets. Figure 5.2 shows that more and more distant point clouds are deformed proportional to the density of the fog. In the case of the visibility V = 20 m, we cannot recognize the shape of the object in the point cloud, but we can still see some cars in the fog image. It gives us the feeling that LiDAR is much more affected than the camera in foggy weather condition. From here, we will show how it affects the object detection of self-driving cars through experiments in the following section.

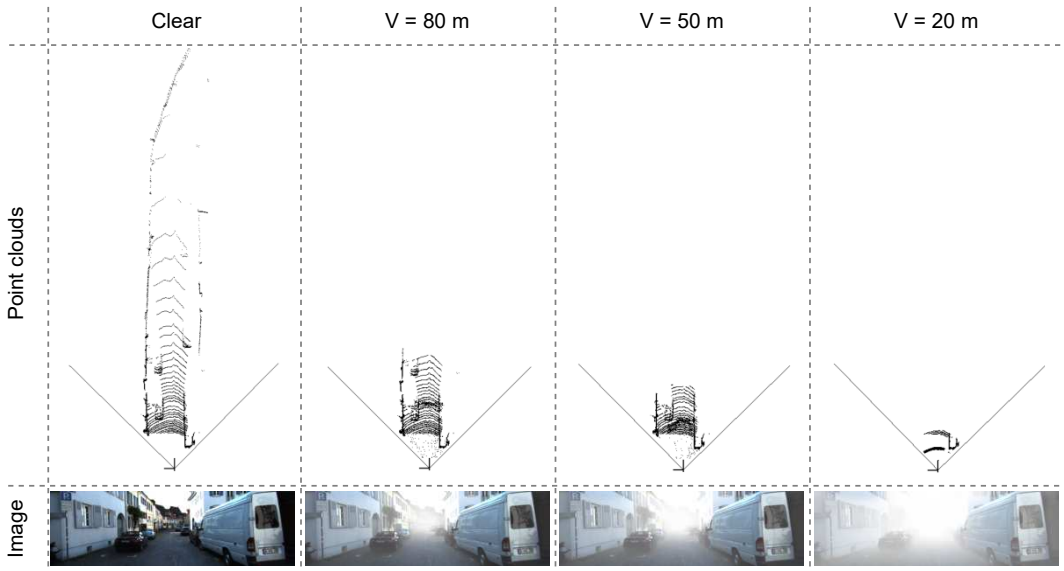


FIGURE 5.2: **Visualization of image and point cloud corresponding to different levels of visibility [162, Fig. 2(a)].** From left to right, image, and point cloud are shown for "Clear" sample (original sample) and augmented foggy samples (64 beams) through different levels of visibility. The point cloud is shown in bird's eye view (BEV) representation. As visibility decreases, the contrast of the image and the range of the point cloud also decrease significantly. 64-beam LiDAR is used here to more clearly see the effect of fog (visualization purpose), but the 4-beam LiDAR is used in the experiment (Section 5.5).

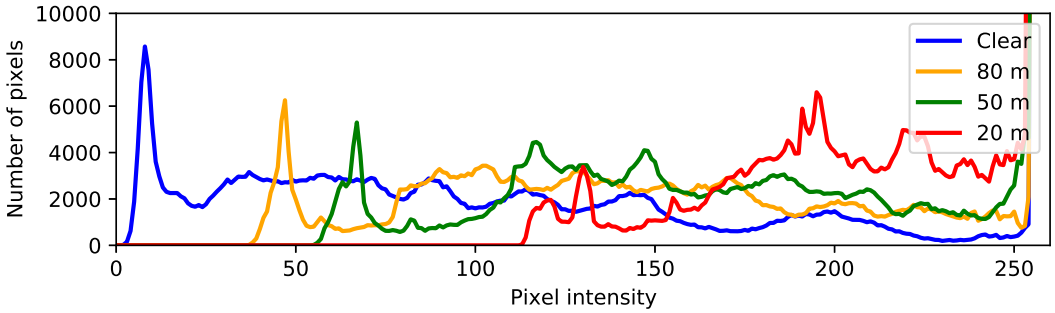
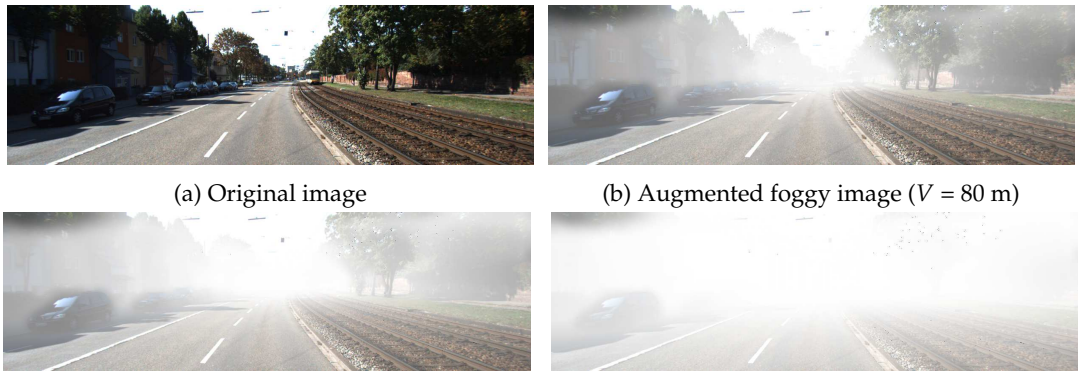


FIGURE 5.3: **Histogram of images with different levels of visibility [162, Fig. 2(b)].** This figure shows the pixel intensity distribution of images in Figure 5.2 corresponding to different levels of visibility. Images are converted to gray scale images here for simplicity.

Figures 5.4 and 5.5 provide illustrations of the effect of fog when applied to the original images. The visibility distances selected in these figures are 20 m, 50 m and 80 m. We can visually note that at a distance of 20 m, we can no longer distinguish certain objects, including cars. Figure 5.4 relate to vehicle objects while Figure 5.5 relates to the distinction of pedestrians. In the case of pedestrians in Figure 5.5, the initial image also contains two wheels, with a lot of shadows. In this case, visibility of 20 m provides degraded images that are difficult to interpret.

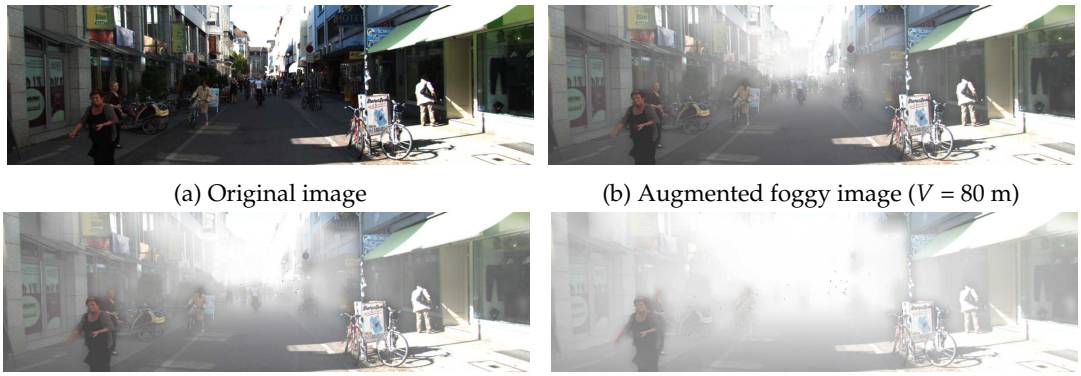
The purpose of the study in this section is to show that our algorithms are sensitive enough to detect things at these distances, even if 20 m is a very close distance between an autonomous vehicle and an obstacle. We present that in the sections below.



(a) Original image (b) Augmented foggy image (V = 80 m)

(c) Augmented foggy image (V = 50 m) (d) Augmented foggy image (V = 20 m)

FIGURE 5.4: **Images with different visibility levels – example 1.** 5.4 (a) show the original sample from the KITTI dataset. 5.4 (b), 5.4 (c) and 5.4 (d) shows the augmented foggy images with visibility V equal to 80 m (β = 0.0375), 50 m (β = 0.0599) and 20 m (β = 0.1498) respectively. β is the fog density (or attenuation) coefficient corresponding to the visibility V.



(a) Original image (b) Augmented foggy image (V = 80 m)

(c) Augmented foggy image (V = 50 m) (d) Augmented foggy image (V = 20 m)

FIGURE 5.5: **Images with different visibility levels – example 2.** The visibility distances are the same as in Figure 5.4.

5.4 Implementation Methodology

The proposed method in Chapter 4, SLS-Fusion [163], involves three main steps: depth estimation, data conversion, and LiDAR-based 3D object detection. In our experiments, we try to analyze the contribution of each sensor (camera or LiDAR) for the 3D object detection task. Thus, the two sensors are considered separately.

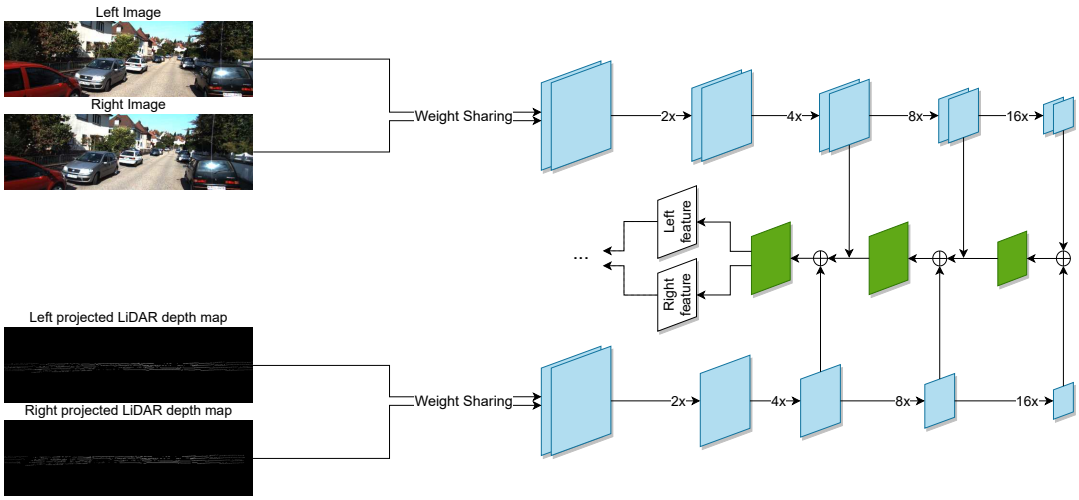


FIGURE 5.6: **SLS-Fusion feature extraction part.** It takes both images and point cloud as input.

Figure 5.6 shows the encoder-decoder network that takes both images and point cloud in the same model to extract feature information.

5.4.1 Modified SLS-Fusion Taking Stereo Camera as Input

Figure 5.7 shows the network representative for the stereo camera only. We start from the global representation in Figure 5.6. In this global representation, we indeed have two inputs to set up our neural network: one for the LiDAR and one for the stereo camera. As it can be seen in Figure 5.7, the neural network is changed to take into account only a stereo camera as input. The last step, obstacle detection, remains the same whether the depth map obtained from LiDAR or the depth map obtained from the stereo camera is used.

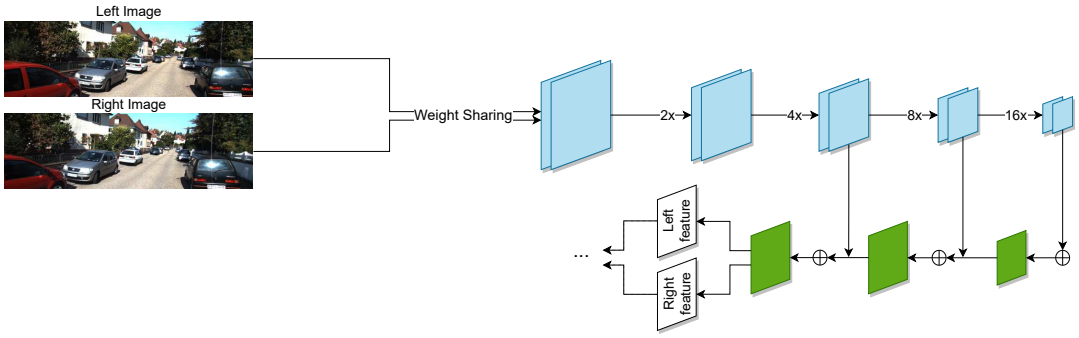


FIGURE 5.7: **Modified feature extraction part of the SLS-Fusion algorithm to adapt it for taking only the stereo camera as input.**

5.4.2 Modified SLS-Fusion Taking LiDAR as Input

As can be seen in Figure 5.7, the LiDAR has been eliminated. Thus, the output depth map is based on the point cloud as input. Here, to build a model that takes LiDAR as input, we do the same thing as above, but instead of removing the point cloud branch we remove the image branch. Figure 5.8 shows the model that uses LiDAR only. In this case, the input data is the LiDAR and the output depth map is based on the LiDAR only.

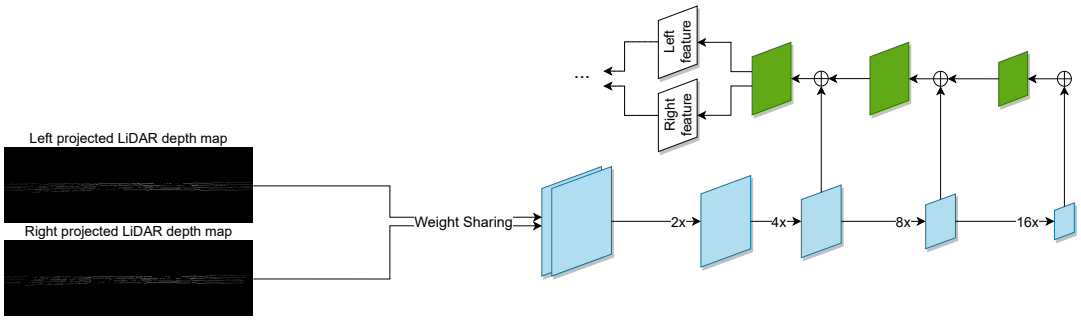


FIGURE 5.8: **Modified feature extraction part of the SLS-Fusion algorithm to adapt it for taking only the LiDAR as input.**

5.5 Experiment Settings

5.5.1 Metrics

The same metrics presented in the previous chapter are used here to evaluate object detection algorithms. The Average Precision (AP) for both 3D and BEV is reported as AP_{3D} and AP_{BEV}, respectively, with thresholds of 0.5 and 0.7 for the intersection over union (IoU). The objects are classified into three difficulty levels, Easy, Moderate and Hard, depending on the size of the 2D bounding box, the occlusion, and the degree of truncation of the object appearing in the RGB image according to Geiger, Lenz, and Urtasun [12]. The experiments are performed using the KITTI [12] and the Multifog KITTI [162] datasets presented in Chapter 3. The original 3D object detection model SLS-Fusion [163] and its modified versions presented in Section 5.4 are used for all experiments.

5.5.2 Datasets

In order to overcome the difficulty of detecting objects in the presence of fog, we are pushed to use stereo camera and LiDAR at the same time. We need to use the two sensors. We will therefore bring the LiDAR and the stereo camera together first. Thereafter, as said previously, we will consider the two sensors separately to evaluate their respective contributions.

To enable the operation of the SLS-Fusion system, the synthetic Scene Flow dataset [144] is first used to train the neural network model for depth estimation, as in the previous chapter. After depth prediction, the KITTI dataset [12] is used to train the 3D object detection part of the SLS-Fusion method. On the other hand, there are few datasets recorded in extreme weather conditions. To address this problem, we decided to create a dataset, derived from the KITTI dataset, augmented with foggy weather conditions. To do this, we use a physics-based fog simulator [83] that converts normal weather data from the KITTI dataset to foggy weather data (this modified dataset is then called Multifog KITTI) as described in Chapter 3. In this dataset, the level of fog is characterised according to a visibility distance expressed in meters. Based on the results of the simulated foggy system, we divide foggy weather into six conditions or levels, according to visibility V (level 1, really dense fog: 20-29 m, level 2: 30-39 m, level 3: 40-49 m, level 4: 50-59 m, level 5: 60-69 m, and level 6, medium fog: 70-79 m).

The detection results of the SLS-Fusion method were obtained by training with the Multifog KITTI dataset [162]. The performance of the 3D object detector can be analyzed by comparing the 3D predictor results with the annotated KITTI data results. However, after creating the new dataset by using the fog simulator, we found that many labeled objects (ground truths) that were visible without fog became invisible in the fog, resulting in incorrect ground truths. For this reason, the ground truths were filtered. The bounding boxes, which have a greater distance to the sensor than the meteorological visibility, are removed and not taken into account in the evaluation. When we manually review the LiDAR and camera data, we see that this method does remove objects that are completely invisible.

5.5.3 Training and Inference

It has been shown that the combination of LiDAR and camera does not outperform the results of models using only LiDAR in normal weather conditions (without fog). On the other hand, LiDAR and camera data are strongly affected by noise in foggy weather. In this case, we do not know what is the best choice: to continue to fuse the two sensors or to use them separately. That is the question we want to answer here. We perform the following experiments: we test the SLS-Fusion model with only camera or LiDAR inputs to see how each sensor contributes to the performance of the model when the data is affected by fog. To analyze the contribution of each sensor, we first compare the results on the KITTI dataset with those on the Multifog KITTI dataset. Knowing that the LiDAR gives poor results in foggy weather, we try to see if a LiDAR with more laser beams (64 instead of 4) would give better results in those adverse conditions.

The visibility interval (corresponding to fog density) can be retrieved. For each level of visibility V (level 1 (dense fog) to level 6 (medium fog)), object detection performance is considered. This can then determine up to what visibility range the models

can provide acceptable performance.

5.6 Experiment Results

As shown in Tables 5.1, 5.2, 5.3, 5.4 and 5.5, the results of different tests using the SLS-Fusion method are given for IoU of 0.5 and 0.7. In each cell of these tables, a pair of numbers A/B corresponds to the results obtained with the AP_{BEV}/AP_{3D} metrics on the KITTI or Multifog KITTI datasets. AP_{BEV} is average precision for BEV and AP_{3D} is average precision for 3D object detection. Below, we describe the result AP_{3D} as the primary metric because it better reflects how well the bounding boxes are predicted for the 3D object detection task.

In Table 5.1, the experiments are divided into two parts. The upper part consists of experiments Idx 1.1, 1.2, 1.3 and 1.4 on the KITTI dataset. In contrast, the experiments in the lower part Idx 2.1, 2.2, 2.3, 2.4 and 2.5 were performed on the Multifog KITTI dataset.

The upper part Idx 1.1 reports the experiment described in Chapter 4). It is a test of the SLS-Fusion model, with a stereo camera and a 4-beam LiDAR fused (S+L4) as inputs experimented on the KITTI dataset. In the next two experiments, we present the results in the same way, but 1.2 only takes a 4-beam LiDAR as input (L4) and 1.3 only takes a stereo camera as input (S).

In this part which concerns using the KITTI dataset without fog, one notices, as expected, that the fusion of the stereo camera and LiDAR provides good results: **93.02%** for Easy and IoU of 0.5. In this configuration, a stereo camera provides better results than a 4-beam LiDAR (**89.39%** versus **78.16%**). Separating the two sensors results in lower performance. This prompts us to consider the two sensors jointly. On the other hand, if we consider the 64 beam LiDAR (Idx 1.4), the results are better (**97.3%** for Easy objects and an IoU of 0.5). Nonetheless, the 64-beam LiDAR price is very high compared to that of cameras and 4-beam LiDAR.

In the second part of Table 5.1, we calculated the same indicators as before, but with data with fog (Multifog KITTI). Performance when fusing a stereo camera and a 4-beam LiDAR (S+L4) remains high (**90.15%**) for Easy objects, which demonstrates the viability of using both sensors at the same time. The performance of the 4-beam LiDAR alone (L4) decreases sharply (**13.43%**), which seems to show that the 4-beam LiDAR is not useful in fog, and it is the stereo camera (S) that has the highest contribution, as its performance remains very high (**89.36%**). The combination of stereo camera and 64-beam LiDAR provides good results (**89.26%** for Easy objects) and for most of the types of objects, but the stereo camera still plays the most important role.

Idx	Method	Dataset	Input	0.5 IoU		
				Easy	Moderate	Hard
1.1	SLS-Fusion	Clear	S+L4	93.16/ **93.02**	88.81/ 86.19	83.35/ 84.02
1.2	SLS-Fusion	Clear	L4	84.02/ **78.16**	72.98/ 68.92	66.34/ 63.92
1.3	SLS-Fusion	Clear	S	89.50/ **89.39**	78.54/ 77.46	75.19/ 69.77
1.4	PointRCNN	Clear	L64	97.3/ **97.3**	89.9/ 89.8	89.4/ 89.3
2.1	SLS-Fusion	Multifog	S+L4	90.27/ **90.15**	79.17/ 78.01	76.12/ 70.21
2.2	SLS-Fusion	Multifog	L4	15.44/ **13.43**	10.63/ 9.58	9.75/ 9.65
2.3	SLS-Fusion	Multifog	S	89.52/ **89.36**	78.75/ 77.71	75.63/ 69.90
2.4	SLS-Fusion	Multifog	L64	81.61/ 77.31	56.89/ 53.87	49.83/ 47.80
2.5	SLS-Fusion	Multifog	S+L64	89.42/ **89.26**	78.79/ 77.82	75.92/ 74.58

TABLE 5.1: **3D object detection results (IoU of 0.5) with different inputs and datasets.** AP_{BEV}/ AP_{3D} results on the Clear KITTI (Clear) and the Multifog KITTI (Multifog) datasets for the category "Car" with IoU of 0.5 and on three levels of difficulty: Easy, Moderate, and Hard. S, L4, L64 denote the stereo camera, the 4-beam LiDAR, and the 64-beam LiDAR, respectively.

Idx	Method	Dataset	Input	0.7 IoU		
				Easy	Moderate	Hard
1.1	SLS-Fusion	Clear	S+L4	87.51/ 76.67	76.88/ **63.90**	73.55/ 56.78
1.2	SLS-Fusion	Clear	L4	56.72/ 38.82	49.25/ 32.02	44.14/ 29.75
1.3	SLS-Fusion	Clear	S	82.21/ 66.54	62.18/ 47.18	56.41/ 43.07
1.4	PointRCNN	Clear	L64	90.2/ 89.2	87.9/ 78.9	85.5/ 77.9
2.1	SLS-Fusion	Multifog	S+L4	83.42/ 69.57	62.79/ 48.19	56.84/ 44.85
2.2	SLS-Fusion	Multifog	L4	10.87/ 9.09	9.09/ 9.09	9.09/ 9.09
2.3	SLS-Fusion	Multifog	S	82.41/ 70.52	62.59/ **48.27**	57.11/ 45.75
2.4	SLS-Fusion	Multifog	L64	58.35/ 42.57	38.80/ 29.27	34.63/ 25.25
2.5	SLS-Fusion	Multifog	S+L64	82.85/ 71.49	62.33/ 48.39	57.10/ 45.78

TABLE 5.2: **3D object detection results (IoU of 0.7) with different inputs and datasets.** AP_{BEV}/ AP_{3D} results on the KITTI dataset for the category "Car" with IoU of 0.7 and on three levels of difficulty: Easy, Moderate, and Hard. S, L4, L64 denote the stereo camera, the 4-beam LiDAR, and the 64-beam LiDAR, respectively.

In order to describe the results in more details, we propose to further illustrate our point. For this, we use illustrations of 3D objects and additional result tables. Figure 5.9 presents some illustrative results of car detection, showing 3D predicted bounding boxes. It is consistent with the results reported above that training on Clear and Multifog gives better results. It shows the 3D bounding boxes on both image and point cloud, and contains four parts: Figure 5.9 (a) gives an example resulting from training on the Clear KITTI and evaluating on the Clear KITTI; Figure 5.9 (c) training on the Clear+Multifog KITTI and evaluating on the Clear KITTI; Figure 5.9 (b) training on the Clear KITTI and evaluating on the Multifog KITTI;

Figure 5.9 (d) training on the Clear+Multifog KITTI and evaluating on the Multifog KITTI. Figure 5.9 (a) shows that the model detects all objects in the context when trained on Clear and evaluated on Clear. When augmenting training on Multifog KITTI in Figure 5.9 (c), the model can still detect all objects, even though the number of false detections is higher. In Figure 5.9 (b), we see that the model cannot detect all objects and misses a distant object when the image data, and especially the LiDAR, are greatly distorted in the foggy environment. In Figure 5.9 (d), still in the foggy environment, but the model is trained on Multifog KITTI, we see that the model can recognize all the objects again despite the increased number of false positives.

(a) Training: Clear; evaluation: Clear

(b) Training: Clear+Multifog; evaluation: Clear

(c) Training: Clear; evaluation: Multifog
Visibility V = 45 m

(d) Training: Clear+Multifog; evaluation: Multifog
Visibility V = 45 m

FIGURE 5.9: **Qualitative Comparison – example 1.** Qualitative results of our SLS-Fusion method for 3D object detection on the Clear KITTI dataset and the proposed dataset (the Multifog KITTI). 3D bounding boxes in red and in green denote the ground truth and the prediction for objects in the scene, respectively. Note that the point cloud is shown in BEV representation.

Tables 5.3, 5.4 and 5.5 show results on AP_{BEV} and AP_{3D} according to the visibility distances (from level 1 to level 6). In this configuration, we consider 3D detection performance by combining a stereo camera and a 64-beam LiDAR (in Table 5.3) analyzed by visibility distance. For the Easy category (IoU of 0.5), the combination of the two sensors (S+L64) leads to an AP greater than **82.52%**. On the other hand, for the Moderate category (with an IoU of 0.7), visibility distance plays a role, and we have an AP gain of about **7.4%** (difference between **46.71%** at level 1 and **54.10%** at level 6).

If we now look at the sensors separately and depending on the visibility distance, we see the following. For the 64-beam LiDAR (Table 5.4), objects classified as Easy (with an IoU of 0.5) are detected at **71.11%** for visibility level of 20 m (level 1), up to **84.95%** for level 6. On the other hand, if we consider Moderate objects (with an IoU of 0.7) we see that at low visibility level 1, the AP is **17.24%**. Conversely, the AP at level 6 is **35.19%**. We see that the results are better when the visibility is better. However, the two results are bad given the safety requirements of autonomous vehicles. All this shows that in the presence of fog, the use of a LiDAR alone does not provide satisfactory results.

Table 5.5 provides the results for the stereo camera taken into account alone, in foggy weather and as a function of the visibility distance. For Easy objects (IoU of 0.5) we notice that the AP does not vary much depending on visibility (maximum **96.17%** of AP). If we consider Moderate objects (with an IoU of 0.7) we can note that the AP varies according to the visibility distance, ranging from **46.70%** to **54.09%**. If we compare the results to those of a 64-beam LiDAR, we can clearly see the superiority of the stereo camera. Therefore, in foggy weather, LiDAR alone is not suitable.

V	0.5 IoU			0.7 IoU		
	Easy	Moderate	Hard	Easy	Moderate	Hard
1	88.95/ 88.68	75.29/ 69.78	69.59/ 68.15	77.05/ 64.99	57.15/ **46.71**	54.83/ 44.05
2	96.42/ 96.16	78.98/ 78.31	75.96/ 74.65	86.36/ 75.49	63.58/ 52.68	57.76/ 46.22
3	89.24/ 89.03	78.15/ 77.27	76.45/ 74.37	83.91/ 71.51	62.69/ 47.98	57.19/ 45.49
4	84.30/ **82.52**	59.41/ 57.83	54.55/ 50.24	62.08/ 47.80	40.77/ 30.85	35.60/ 27.24
5	89.85/ 89.75	83.31/ 77.86	76.68/ 74.60	85.88/ 74.76	62.98/ 52.52	56.85/ 46.52
6	89.68/ 89.46	78.88/ 77.94	76.93/ 74.85	85.70/ 74.79	64.95/ **54.10**	58.28/ 48.12

TABLE 5.3: **Detailed results for each fog density using the stereo camera and the 64-beam LiDAR.** The SLS-Fusion algorithm is applied to the Multifog KITTI dataset.

In Figures 5.11 5.10, we show detection results using our SLS-Fusion method, taking the stereo camera and 4-beam LiDAR as input on four randomly chosen scenes in the Clear KITTI and Multifog KITTI datasets. Specifically, we show the predicted 3D bounding box results of the frontal-view images and the BEV point cloud. We have the corresponding pairs of pictures in Clear KITTI and Multifog KITTI as follows: (5.10 (a), 5.10 (b)), (5.10 (c), 5.10 (d)), (5.11 (a), 5.11 (b)), (5.11 (c), 5.11 (d)). We can see

V	0.5 IoU			0.7 IoU		
	Easy	Moderate	Hard	Easy	Moderate	Hard
1	73.64/ 71.11	48.72/ 45.54	44.73/ 40.27	43.84/ 26.12	27.55/ **17.24**	26.30/ 16.94
2	85.46/ 77.42	57.86/ 53.62	50.05/ 47.06	57.04/ 41.07	37.90/ 28.04	33.82/ 24.76
3	83.38/ 78.72	57.60/ 54.59	50.38/ 48.69	56.03/ 43.49	38.18/ 29.29	32.00/ 25.71
4	89.88/ 89.78	84.50/ 79.39	78.54/ 77.13	84.27/ 72.15	64.99/ 54.63	57.48/ 47.27
5	83.47/ 82.84	58.26/ 57.17	50.44/ 49.29	66.79/ 54.62	42.25/ 34.84	36.83/ 30.28
6	86.24/ **84.95**	59.87/ 58.01	56.14/ 51.38	65.77/ 51.37	41.95/ **35.19**	39.42/ 30.89

TABLE 5.4: **Detailed results for each fog density using only the 64-beam LiDAR.** The SLS-Fusion algorithm is applied to the Multifog KITTI dataset. It takes 64-beam LiDAR as input in this case.

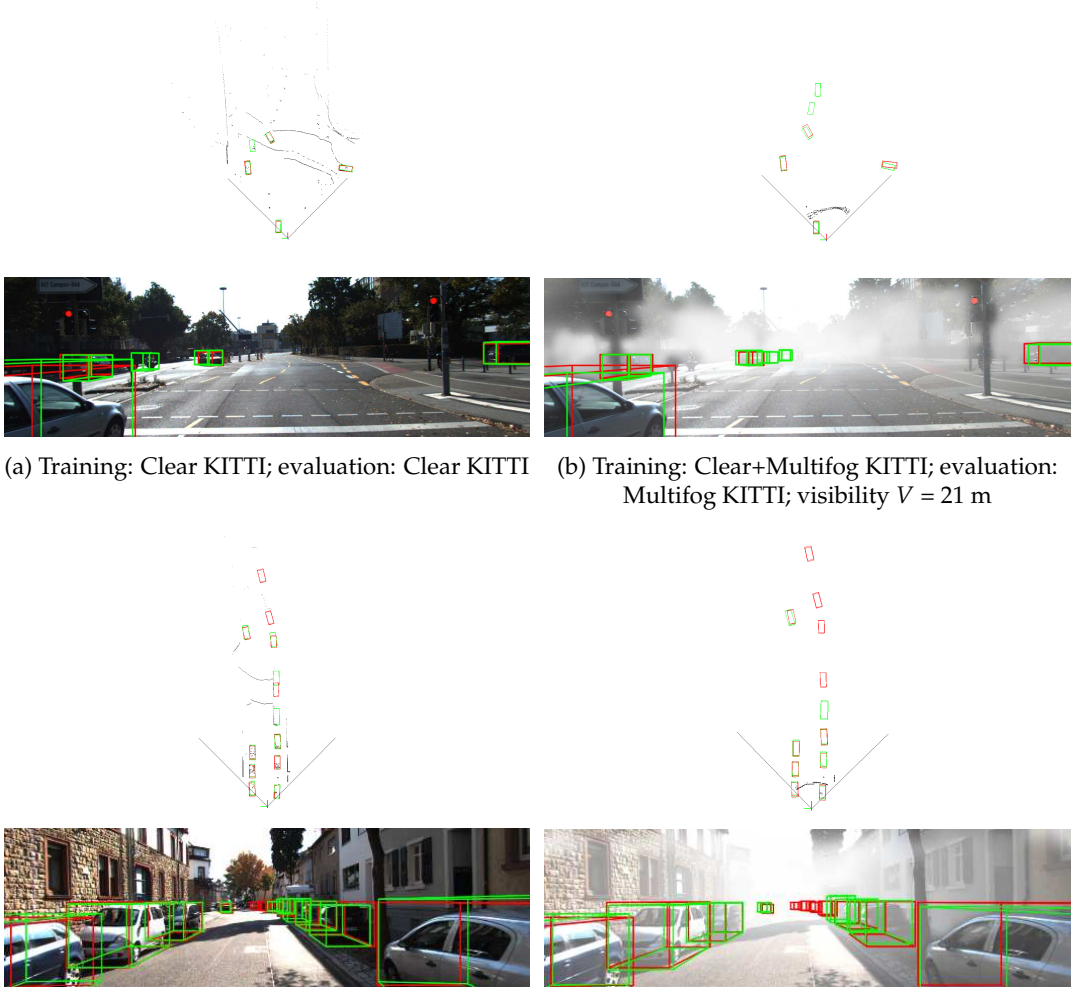
V	0.5 IoU			0.7 IoU		
	Easy	Moderate	Hard	Easy	Moderate	Hard
1	89.02/ 88.77	75.19/ 69.56	69.41/ 68.10	77.02/ 64.85	57.11/ **46.70**	54.82/ 44.03
2	96.47/ **96.17**	78.97/ 78.29	75.91/ 74.66	86.31/ 75.45	63.57/ 52.66	57.75/ 46.20
3	89.25/ 89.05	78.14/ 77.25	76.44/ 74.33	83.88/ 71.50	62.66/ 47.97	57.20/ 45.50
4	84.33/ 82.55	59.40/ 57.81	54.53/ 50.20	62.07/ 47.79	40.75/ 30.84	35.60/ 27.23
5	89.86/ 89.77	83.30/ 77.85	76.65/ 74.59	85.86/ 74.75	62.97/ 52.50	56.83/ 46.50
6	89.70/ 89.47	78.85/ 77.92	76.92/ 74.84	85.68/ 74.77	64.93/ **54.09**	58.26/ 48.10

TABLE 5.5: **Detailed results for each fog density using only the stereo camera.** The SLS-Fusion algorithm is applied to the Multifog KITTI dataset.

that the model predicts well under favourable weather conditions (Clear KITTI). However, the model encountered difficulty in foggy weather conditions (Multifog KITTI). In particular, in addition to recognizing easy objects, it often predicts more false positive objects as well as missing distant objects (false negative objects). This can be explained by the fact that distant objects are much more affected by foggy weather conditions.

5.7 Discussion and Conclusion

In this chapter, the main objective was to analyze the effect of fog on the SLS-Fusion method for 3D object detection. To do this, a novel synthetic dataset was created for foggy driving scenes. It is called the Multifog KITTI dataset as described in Chapter 3. This dataset was generated from the original dataset, the KITTI dataset, by applying fog at different levels of visibility (20 to 80 m). This dataset covers the left and right images, 4-beam LiDAR data and 64-beam LiDAR data, although the foggy 64 beams data are yet to be exploited. We have also tested the capabilities of 3D obstacle detection using two types of sensors: two versions of LiDAR (4-beam and 64-beam) and a stereo camera. Based on object detection performance, we have analyzed several aspects: the contribution of the two types of sensors (camera and LiDAR) both in normal weather and in foggy weather conditions, when they are combined and when they are used separately. The main result is that using LiDAR in



(a) Training: Clear KITTI; evaluation: Clear KITTI

(b) Training: Clear+Multifog KITTI; evaluation: Multifog KITTI; visibility V = 21 m

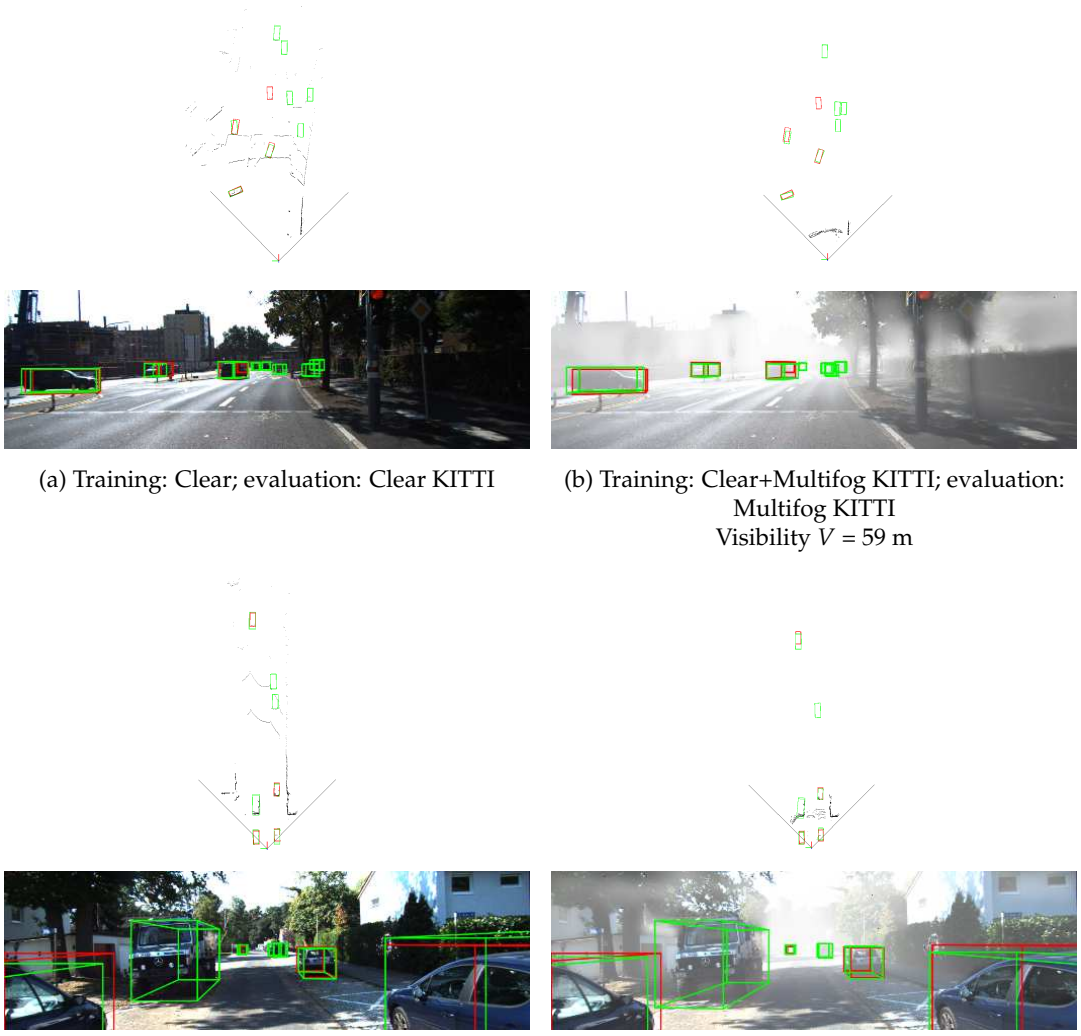
(c) Training: Clear KITTI; evaluation: Clear KITTI

(d) Training: Clear+Multifog KITTI; evaluation: Multifog KITTI; visibility V = 37 m

FIGURE 5.10: **Qualitative Comparison – example 2.** Qualitative results of our SLS-Fusion method for 3D object detection on the Clear KITTI dataset and the proposed dataset (the Multifog KITTI). 3D bounding boxes in red and in green denote the ground truth and the prediction for objects in the scene, respectively. Note that the point cloud is shown in BEV representation.

foggy weather leads to a slightly worse obstacle detection performance (even worse when the LiDAR is a 4-beam laser sensor). On the other hand, results based on stereo camera are promising in foggy weather conditions, regardless of level of visibility. And finally, the original fusion of the stereo camera and the LiDAR still gives slightly better results in foggy weather conditions than the camera-only model, but almost negligible.

The results in Tables 5.1, 5.2, 5.3, 5.4 and 5.5 contain a lot of information that needs to be interpreted in detail. As a reminder, the results of this study are obtained on real LiDAR and camera data, initially acquired in clear weather, to which fog has been



(a) Training: Clear; evaluation: Clear KITTI

(b) Training: Clear+Multifog KITTI; evaluation: Multifog KITTI
Visibility V = 59 m

(c) Training: Clear KITTI; evaluation: Clear KITTI

(d) Training: Clear+Multifog KITTI; evaluation: Multifog KITTI
Visibility V = 67 m

FIGURE 5.11: **Qualitative Comparison – example 3.** Qualitative results of our SLS-Fusion method for 3D object detection on the Clear KITTI dataset and the proposed dataset (the Multifog KITTI). 3D bounding boxes in red and in green denote the ground truth and the prediction for objects in the scene, respectively. Note that the point cloud is shown in BEV representation.

added. The model used is a simple model, which considers only the macroscopic attenuation phenomenon. This model has been calibrated on tests performed in a controlled environment on standard sensors. However, this model has two limitations. (i) First, it should be verified that it is valid for the sensors used in the KITTI dataset, because, depending on the sensor (brand, type, internal settings), the impact of fog can be more or less strong. In the case of the LiDAR, the threshold effects when passing from the raw signals to the point cloud can have a strong impact on the model we use. For the camera, the exposure setting is not considered here, and may once again have an impact not modeled here. (ii) The model used here does

not consider the microscopic phenomena of light diffusion. Thus, the halo effects for the camera and the backscattering effects for a LiDAR sensor are not simulated here. These different elements are known limitations of the model and clearly explained. They can have an impact on the results, so the results presented here should not be taken as categorical, but as initial results, allowing to compare sensors, and to find data fusion solutions adapted to the autonomous vehicle.

For future work, two options can be considered to circumvent the limitations of the model used here. The first would be to perform acquisitions on a real site, under real adverse weather conditions. The second, more promising option, would be to use more complex 3D models that implement microscopic scattering phenomena and simulate the complete path of light from objects to the sensor and the sensor itself.

Chapter 6

Analysis of the Role of Each Sensor in the 3D Object Detection Task: Ablation Study

Contents

Chapter overview: In the previous chapter, we evaluated SLS-Fusion using the KITTI data set, which utilized the improved work of PoinRCNN for predicting the 3D bounding boxes of detected objects. The results showed that SLS-fusion outperformed most advanced methods, particularly the Pseudo-LiDAR++, as shown in Table 4.2 in Chapter 4. However, when compared to the original PointRCNN detector that used expensive 64-beam LiDAR, the performance of SLS-fusion was lower. It is expected that the 64-beam LiDAR without fusion with stereo cameras is superior because high-resolution LiDAR sensors can provide precise depth information. However, these LiDAR sensors are very expensive, with a 64-beam model costing approximately 75k dollars (USD). As the number of LiDAR beams increases, thus increasing the amount of point clouds generated, the cost of the LiDAR sensor also increases (1k dollars to 75k dollars). In this chapter, we aim to find a compromise between the quality of the LiDAR used and the 3D object detection performance obtained. The optimized solution is to find the cheapest LiDAR model (minimum beam number) used or not used in the SLS-fusion model that corresponds to an acceptable accuracy error achieved by the 3D object detector (PointRCNN in our

case). Furthermore, in this chapter, we analyze how the stereo and LiDAR sensors contribute to the performance of the SLS-fusion model for 3D object detection, depending on the number of LiDAR beams used. As explained in Section 5.4 of Chapter 5 and shown in Figure 6.1, we divide the decoder of the SLS-fusion model into independent decoder networks to separate the component parts of the network that represent LiDAR and stereo camera architectures. The decoder inside the SLS-fusion model is the only component responsible for fusing features between LiDAR and stereo sensors. One of the best-known techniques employed for this purpose is called "ablation study" [179].

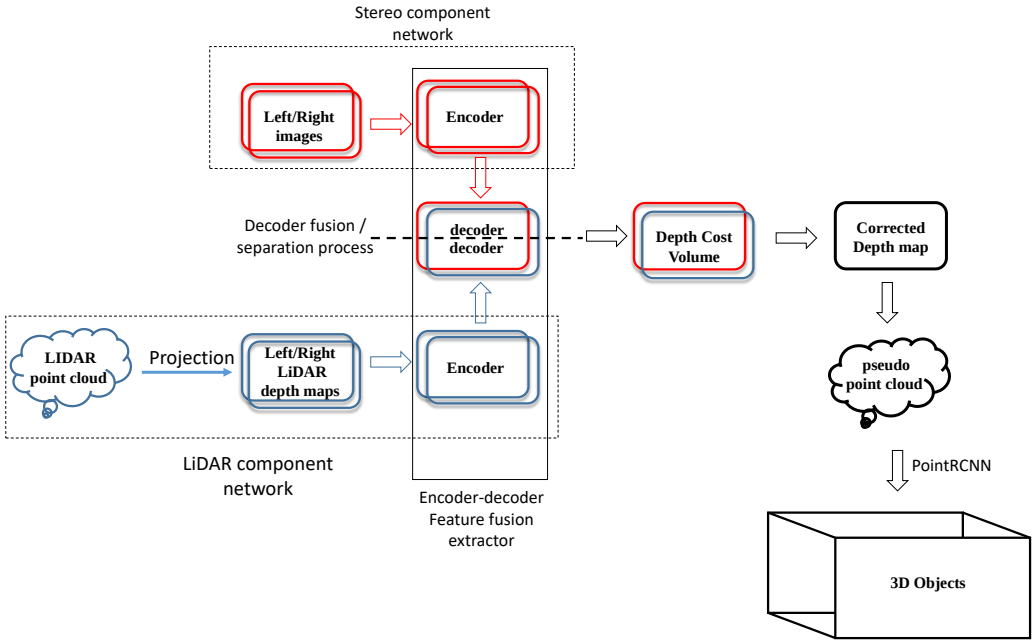


FIGURE 6.1: **Overall structure of SLS-Fusion neural network.**

6.1 Introduction

Given a pair of images from a stereo camera and point clouds from a LiDAR as inputs to detect 3D objects, SLS-fusion based deep learning approach have shown high performance in this domain. The analysis of this performance is depending on the contribution of the neural network component of each sensor (LiDAR or stereo) and of the type of LiDAR selected to the overall architecture of the system. In this work, LiDAR sensors compared to their number of beams are grouped into three main types: low cost LiDARs (4 or 8 beams), medium cost LiDARs (16 beams) and high cost LiDARs (32 or 64 beams). This kind of study, particularly in artificial intelligence, is known as an ablation study [179, 180]. An ablation study is when we remove certain components from a system to understand how the other components

of the system perform. This characterizes the impact of every action on the overall performance and capacity of the system.

In fact, the study of the structure of neural networks has become best practice for machine learning research, as they provide an overview of the relative contribution of individual architectures and components to model performance. It consists of several trials such as removing a layer from our neural network, removing a regularizer, removing or replacing a component from our model architecture, optimizer retraining the network, and then observing how that affect the performance of the model. However, as machine learning architectures become deeper and deeper and the size of the training data is increasing [181], there is an explosion in the number of different architectural combinations that must be assessed to understand their relative performance.

6.2 Characteristics of the Neural Network Architecture Used

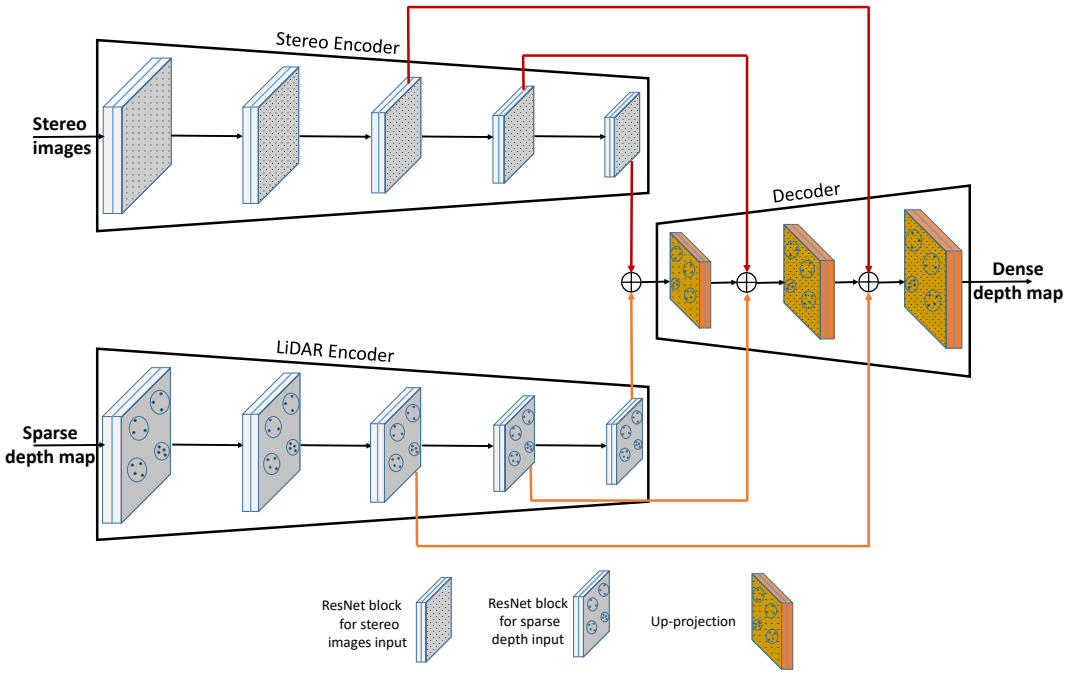


FIGURE 6.2: **Presentation of the SLS-Fusion encoder-decoder architecture.**

In our case, we will use this technique of analysis to identify the influence of each sensor to the overall system of our proposed neural network, taking into account the type of selected LiDAR (low, medium, or high cost) and the average precision of both LiDAR and stereo components of the model architecture in detecting 3D

objects. Finally, we will try to interpret the results by analyzing what happen inside the neural network.

LiDAR and cameras are sensors which are widely seen in almost all autonomous vehicles. The main important advantage of a LiDAR is that it provides extremely accurate depth values, however, their output is always sparse and low resolution. The camera, in contrast with LiDAR, is normally a passive sensor that provides high resolution outputs, but they give, after a certain processing, low to medium precision depth values. So, the combination of the outputs of LiDAR and cameras helps to overcome their individual limits. However, how LiDAR and the camera can be fused?

In this work, we have proposed a new strategy based neural network to fuse data of stereo images and projected LiDAR point clouds. The main component of our proposed neural network SLS-Fusion, used to fuse or separate LiDAR and stereo camera feature (for an ablation study), is the encoder decoder component (see Figures 6.1 and 6.2). It is the main part of SLS-fusion network that aims to enrich the feature maps and thus leads to get better predicted depth maps from the stereo camera and the projected LiDAR images. To understand all of this, we will explain how encoder-decoder component work and how it will help to improve the precision of the system when we use low, medium, or high cost LiDAR.

As shown in Figure 6.2, both stereo camera and LiDAR encoders are composed of a series of residual blocks of neural networks ResNet followed by step-down convolution to reduce the feature resolution of the input. ResNet is a pile of residual neural network blocks, and each residual block is a stack of layers placed in such a way that the output of one layer is taken and added to another deeper layer within the block, as shown in Figure 6.3. The main advantage of ResNet is it prevent the accuracy to saturate and then degrades rapidly during training of deeper neural networks (networks with more than 20 layers). This advantage helps us to choose a network as deep as we need for our problem. This is what we need in our case to extract as much as possible detailed features of sparse LiDAR data and high resolution stereo images. This process has considerably assisted the decoder network to properly fuse the extracted features.

The network of the decoder consists of adding the functions of both LiDAR and stereo encoders, then up projecting the result to progressively increase the resolution of the features and to generate a dense depth map as a decoder output. Because the sparse input of LiDAR is heavily linked to the depth decoder output, features related to the LiDAR sensor should contribute more to the decoder than features related to the stereo sensor. However, as the add operation promotes the features on both sides [182], the decoder is encouraged to learn more features related to stereo images in order to keep consistent with the features related to the sparse depth from

LiDAR. In this way, whatever the type and associated resolution of the selected LiDAR (low, medium, or high cost types), the decoder network will correctly learn merged features. Consequently, the proposed SLS-fusion always outperforms all types of LiDAR sensors in the 3D object detection, as shown in the next section.

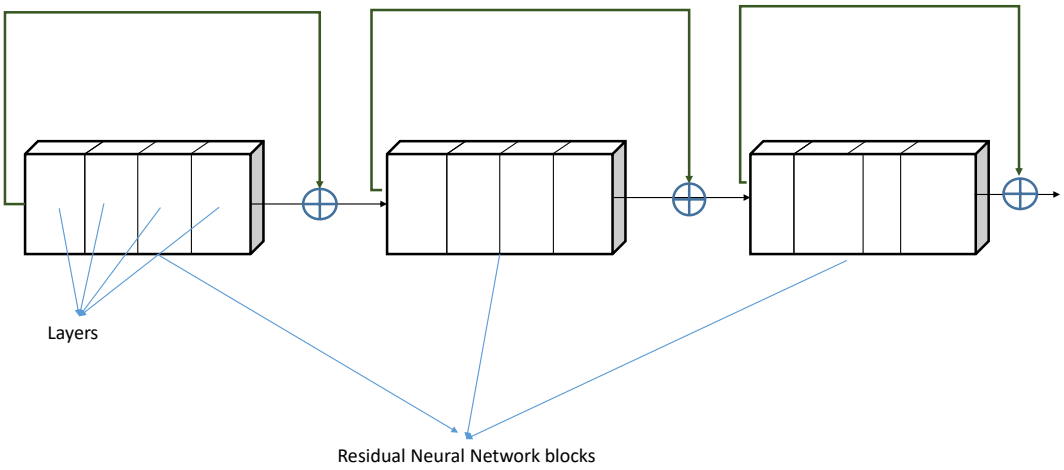


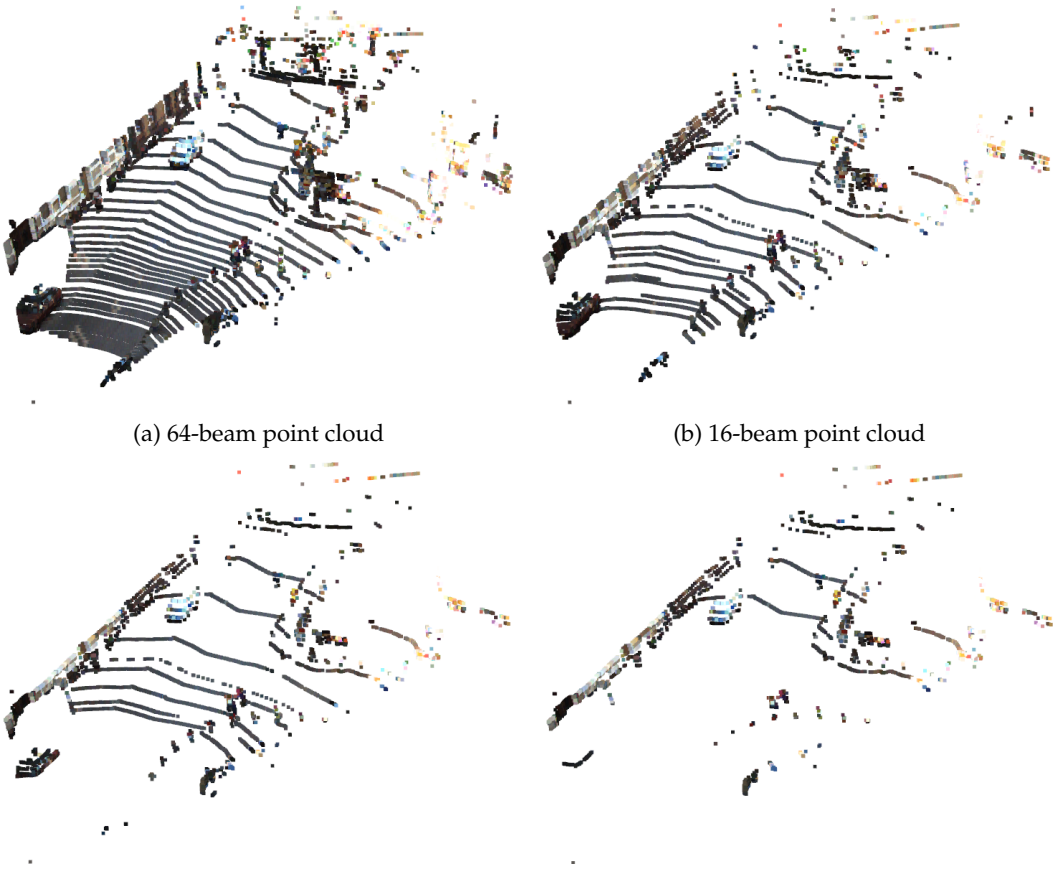
FIGURE 6.3: **Resnet block architecture.**

6.3 Assessment of the Different Network Architectures Implemented

In this section, the results obtained with each component of the SLS-Fusion model (Stereo camera, LiDAR 4 to 64 beams) are presented in order to understand the impact of each component on the ultimate detection performance of 3D objects and show how changes results. To do this, we perform a complete ablation study by disabling each component as previously explained, or by changing the number of LiDAR model component beams. As shown in Figure 6.4, the impact of increasing the number of LiDAR beams will increase the amount of point clouds that represent the environment around the autonomous car. Consequently, the LiDAR contribution to the performance of the object detection model will be enhanced. However, as shown in Table 6.1, increasing the number of beams from 4 to 64 beams will significantly increase the cost of the LiDAR sensor (from one thousand dollars to one hundred thousand dollars). An optimized solution consists in selecting the number of corresponding LiDAR beams that provides us with an expected performance value. For a more complete survey on the tested LiDARs in the market, the reader is referred to [9].

Model	Channels (vertical)	Range (m)	HFoV/ RES (degree)	VFoV/ RES (degree)	Cost ($)
VLS-128 [183]	128	300	360°/ 0.2°@10Hz	+15°to -25°/ 0.11°	100k
AT128 [184]	128	200	120°/0.1°	25.4°/ 0.2°	NA
Pandar128 [185]	128	200	360°/ 0.1°@10Hz	+15°to -25°/ 0.125°	NA
HDL-64E S2, S3 [186]	64	120	360°/ 0.17°@10Hz	+2.0°to -24.9°/ 0.4°	75k
Pandar64 [187]	64	200	360°/ 0.2°@10Hz	+15°to -25°/ 0.167°	30k
HDL-32E [188]	32	100	360°/ 0.2°@10Hz	+10.67°to -30.67°/ 1.33°	30k
RS-LiDAR-32 [189]	32	200	360°/ 0.1°	+15°to -25°/ 0.33°	16,8k
VLP-16 [190]	16	100	360°/ 0.2°@10Hz	±15°/ 2°	8k
HS8 [191]	8	100	120°/ 0.18°	6.66°/ 0.36°	4k
Scala [192]	4	200	145°/ 0.25°	3.2°/ 0.8°	0.6k

TABLE 6.1: **Comparison of some LiDAR sensors.** Channels shows the number of laser beams of the LiDAR sensor vertically. Range indicates the maximum distance that a LiDAR can detect objects. HFoV/ RES and VFoV/ RES decode horizontal and vertical field of view and angular resolution, respectively. There are a number of LiDARs whose resolution depends on the frequency.



(a) 64-beam point cloud

(b) 16-beam point cloud

(c) 8-beam point cloud

(d) 4-beam point cloud

FIGURE 6.4: **LiDAR point clouds representing the environment around the autonomous vehicles.** The point cloud is colored according to RGB image.

6.3.1 Metrics

To better understand the detection process and the results achieved by our study, detection assessment measurements are used to quantify the performance of our proposed detection algorithm in various situations. Among the popular measures for reporting results, there are basic concepts and evaluation criteria used for object detection [193] as follows:

Confidence level: As outlined in Chapter 4, the output of an object detector is characterized by a 3D boundary box, a class, and a confidence level. The confidence level may be taken into account in the precision and recall calculations by considering as positive detections only those which have confidence above the confidence threshold T.

Intersection over union: As the aim of our work is to develop an automated object detection method, the evaluation metric used measures the degree to which the detected boundary boxes are close to the ground-truth boundary boxes. This measurement, called intersection over union (IoU), is performed by assessing the amount of overlap between the predicted boundary box and the ground truth boundary box divided by the area of union between them. A perfect match happens when IoU=1 and, if both limit boxes fail to intercept, IoU=0. A correct detection is considered when the IoU is above a defined IoU threshold. IoU values are usually expressed in percentages, and the most used IoU threshold values are 50% and 70%.

Basic measures: The concept of True Positive, True Negative, False Positive and False Negative, is rather standard and often used in the detection theory. In our work, we use these measures to calculate some evaluation measures such as precision and recall as follows:

- True positive (TP): A ground-truth bounding box correctly detected; Detection with IoU greater than or equal to a IoU threshold
- False positive (FP): A nonexistent object incorrectly detected or a detection of an existing object incorrectly placed; Detection with IoU less than IoU threshold
- False negative (FN): An undetected ground-truth bounding box.
- True Negative (TN): Not used. All possible bounding boxes correctly not detected (so unlimited possible boxes not detected within an image). For this reason, it is not used by the measures.

Consider a detector that assumes that each rectangular area of the image can hold a target object. So if there is an object to detect, the detector will correctly find it through one of the many predicted boxes. This is not an effective way of detecting objects, since many false predictions are made. On the other hand, a detector that never generates a bounding box, will never detect an error. These extreme examples

highlight two important concepts, called precision and recall, which are explained in more detail below:

- Precision: is the capacity of a model to only identify relevant objects. It is the percentage of correct positive predictions divided by all detected bounding boxes, and is given by:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{6.1}$$

- Recall: is the capacity of a model to find all relevant ground-truth bounding boxes. It is the percentage of correct positive predictions divided by all given ground truths and is given by:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6.2}$$

Precision-Recall curve: The Precision-Recall curve, used by PASCAL VOC challenge [10], is a good way to evaluate the performance of an object detector as the confidence is changed by plotting a curve for each detected object. To make things clearer, we provide an example to better understand how the Prevision-Recall curve is plotted. Consider the detections as seen in Figure 6.5, there are 6 images with 11 ground truth objects represented by the green bounding boxes and 21 red bounding boxes. Each red bounding box must have a confidence level greater than 50% to be considered as a detected object and is identified by a letter (B1, B2,...,B21).

Table 6.2 shows the bounding boxes with their corresponding confidences. The last column identifies the detections as TP or FP. In this example a TP is considered if IoU threshold is equal to 20%, otherwise it is a FP. By looking at the images above, we can roughly tell if the detections are TP or FP.

In some images there is more than one detection overlapping a ground truth (in images 2, 3, 4, 5, 6). In those cases, the predicted box with the highest IOU is considered TP (in image 2: B5 is TP while B4 is FP because IOU between B5 and the ground truth is greater than the IOU between B4 and the ground truth).

The Precision – Recall curve is plotted by calculating the precision and recall values of the accumulated TP or FP detections. For this, first we need to order the detections by their confidences, then we calculate the precision and recall for each accumulated detection as shown in Table 6.3 (Note that for recall computation, the denominator term is constant at 11 since GT boxes are constant irrespective of detections).

6.3.2 Ablation Results

This part concerns of using precision-recall curves to better understand the effect and the role of each component of SLS-Fusion on the entire model performance. It

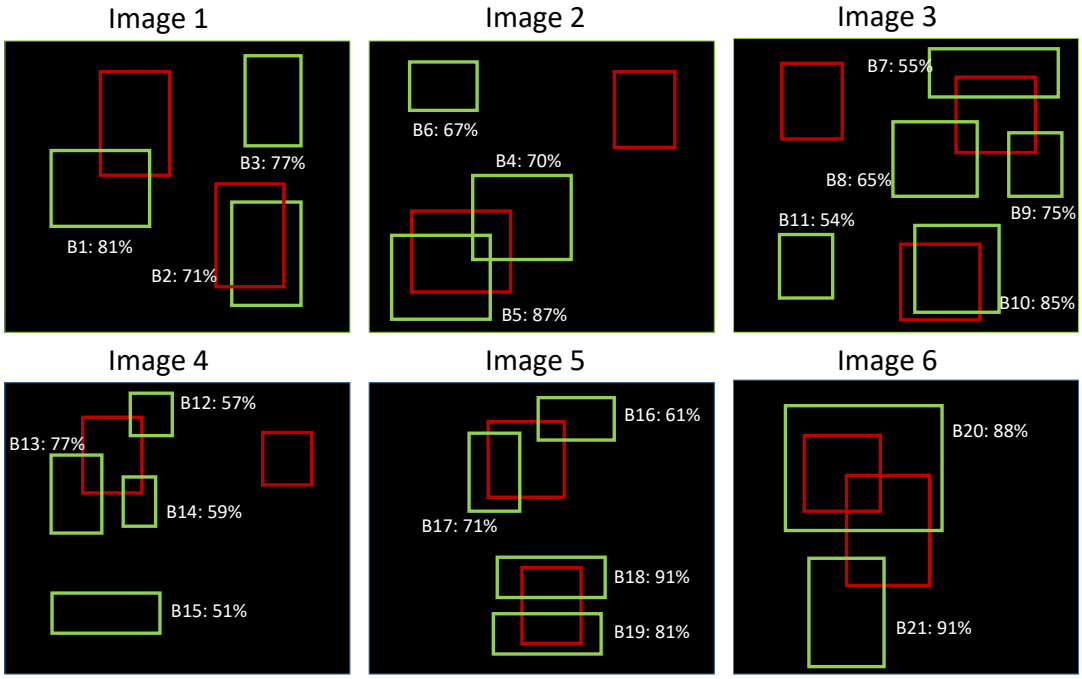


FIGURE 6.5: **Example to explain how the Prevision-Recall curve is plotted.** Red bounding boxes are detected objects, green bounding boxes are ground truth objects.

corresponds to stereo component, LiDAR component, including changing the number of LiDAR beams from 4 to 64. To do this evaluation, we use KITTI evaluation benchmark of 3D bounding boxes or 2D bounding boxes in BEV to compute precision-recall curves for detection as explained in the previous section. The BEV for autonomous vehicles is a vision monitoring system that is used for better evaluation of obstacle detection. This system normally includes between four and six fisheye cameras mounted around the car to provide right, left and front views of the car's surroundings.

Figure 6.7 shows the Precision-Recall (P-R) curves obtained by taking into account respectively stereo cameras, 4-beam LiDAR, 8-beam LiDAR, 16-beam LiDAR and 64-beam LiDAR. As shown in this figure, an object detector is considered good if its precision stays high as recall increases, which means that only relevant objects are detected (0 false positives = high precision) when finding all ground truth objects (0 false negatives = high recall). On the other hand, a poor object detector needs to increase the number of detected objects (increasing false positives = lower precision) in order to retrieve all ground truth objects (high recall). That is why the Precision – Recall curve usually starts with high precision values, decreasing as recall increases. Finally, detection results are divided into three levels of difficulty (Easy, Moderate or Hard) mainly depending on the dimension of the bounding box and the level of

Images	Det.	Conf.	TP/ FP
Image 1	B1	81%	FP
Image 1	B2	71%	TP
Image 1	B3	77%	FP
Image 2	B4	67%	FP
Image 2	B5	70%	TP
Image 2	B6	87%	FP
Image 3	B7	55%	TP
Image 3	B8	65%	FP
Image 3	B9	75%	FP
Image 3	B10	85%	TP
Image 3	B11	54%	FP
Image 4	B12	57%	FP
Image 4	B13	77%	FP
Image 4	B14	59%	FP
Image 4	B15	51%	FP
Image 5	B16	61%	FP
Image 5	B17	71%	TP
Image 5	B18	91%	TP
Image 5	B19	81%	FP
Image 6	B20	88%	TP
Image 6	B21	91%	FP

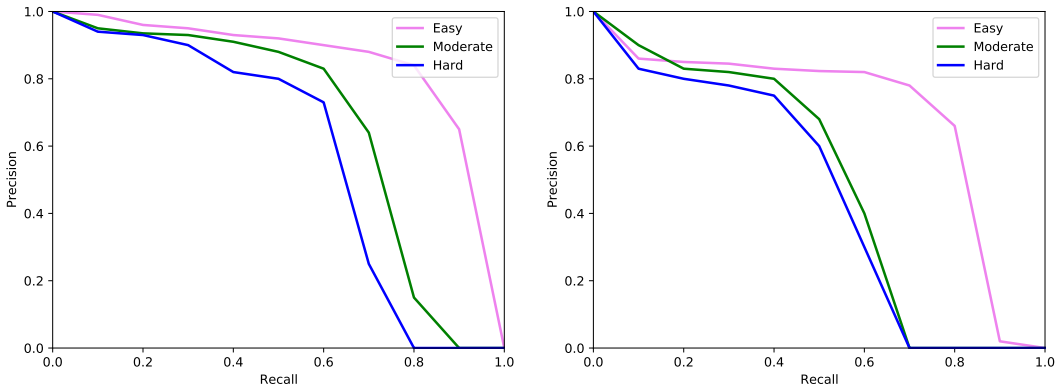
TABLE 6.2: **True and False positive detected bounding boxes with their corresponding confidences.** Det. and Conf. denote detection and confidence, respectively.

occlusion of the detected objects, especially for cars.

All these curves give the shape we are looking for. At points with lower recall, the precision is correspondingly high, and at very high recall, the precision begins to drop sharply. Consider, the minimal recall score corresponds to the point when the P-R curve starts to drop sharply. The best detector is then the detector who can achieve a high precision score (higher than 0.8) while the minimal recall score, is closest to 1. What can also see, based on this idea, the P-R curves obtained for 2D objects in BEV are always better than those obtained for 3D objects. This is due that the level of inaccuracy in detecting bounding boxes in 3D dimensions is always greater than in two dimensions. However, detecting the surrounding cars in BEV projection view reduces the precision on estimating the distance of detected objects (cars) from the autonomous vehicle. One also notices, as expected, P-R curves for stereo camera shows better results than 4-beam LiDAR (minimal recall 0.5 versus minimal recall 0.3 in Hard level of difficulty). However, fusing the two sensors (stereo camera + 4-beam LiDAR) improve detection performance (minimal recall is 0.57 on Hard difficulty). On the other hand, when the number of beams of LiDAR passes from low cost 4-beam LiDAR to high-cost 64-beam LiDAR, the detector provides the best P-R curves (minimal recall is 0.7 in Hard difficulty level).

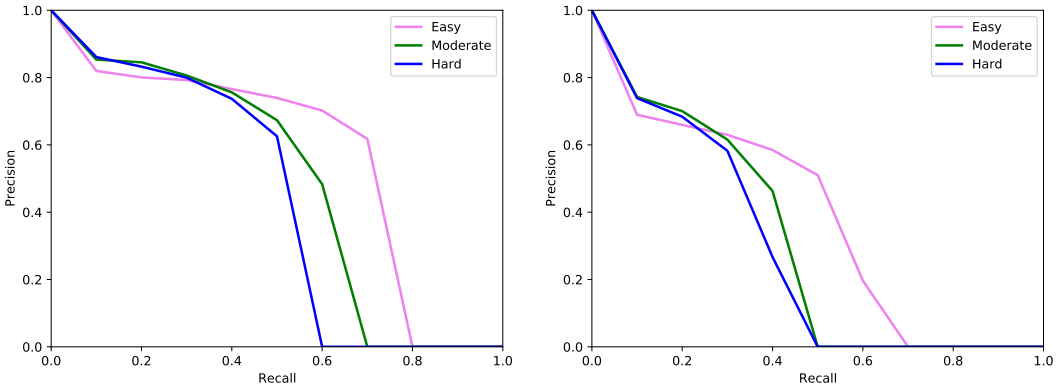
Images	Det.	Conf.	TP	FP	Accum. TP	Accum. FP	Precision	Recall
Image 5	B18	91%	1	0	1	0	1	0.09
Image 6	B21	91%	0	1	1	1	0.5	0.09
Image 6	B20	88%	1	0	2	1	0.666	0.181
Image 2	B6	87%	0	1	2	2	0.5	0.181
Image 3	B10	85%	1	0	3	2	0.6	0.272
Image 1	B1	81%	0	1	3	3	0.5	0.272
Image 5	B19	81%	0	1	3	4	0.428	0.272
Image 1	B3	77%	0	1	3	5	0.375	0.272
Image 4	B13	77%	0	1	3	6	0.333	0.272
Image 3	B9	75%	0	1	3	7	0.3	0.272
Image 1	B2	71%	1	0	4	7	0.363	0.363
Image 5	B17	71%	1	0	5	7	0.416	0.454
Image 2	B5	70%	1	0	6	7	0.461	0.545
Image 2	B4	67%	0	1	6	8	0.428	0.545
Image 3	B8	65%	0	1	6	9	0.4	0.545
Image 5	B16	61%	0	1	6	10	0.375	0.545
Image 4	B14	59%	0	1	6	11	0.353	0.545
Image 4	B12	57%	0	1	6	12	0.333	0.545
Image 3	B7	55%	1	0	7	12	0.368	0.636
Image 3	B11	54%	0	1	7	13	0.35	0.636
Image 4	B15	51%	0	1	7	14	0.333	0.636

TABLE 6.3: **Precision and recall for each accumulated detection** bounding box ordered by their confidence measures. Det., Conf. and Acumm. denote detection, confidence and accumulated, respectively.



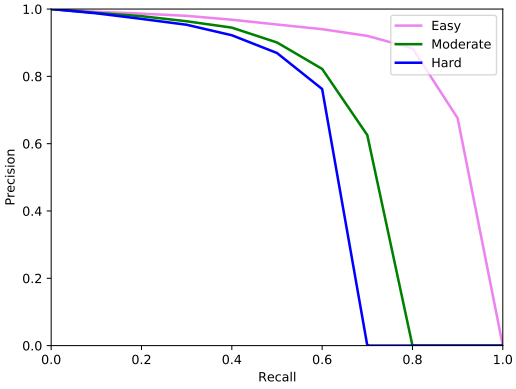
(a) BEV, stereo

(b) 3D, stereo

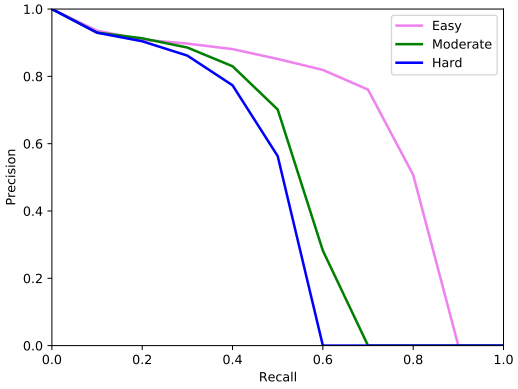


(a) BEV, 4-beam LiDAR

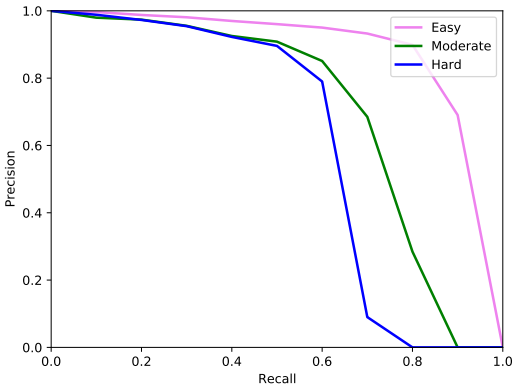
(b) 3D, 4-beam LiDAR



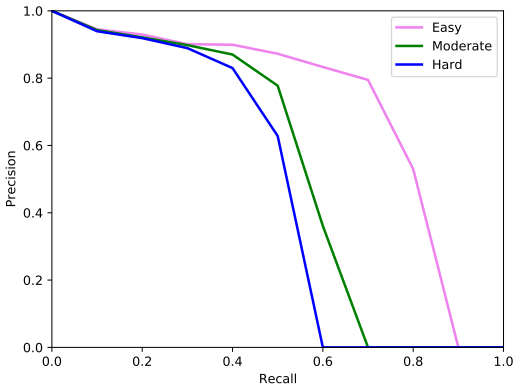
(c) BEV, 8-beam LiDAR



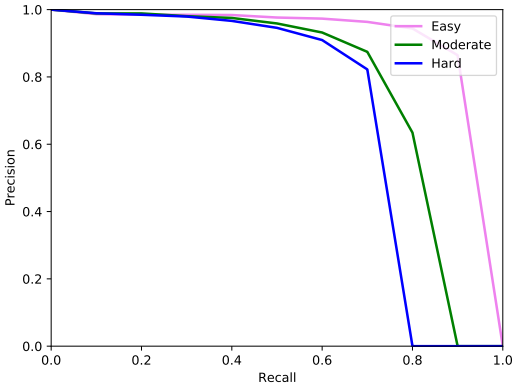
(d) 3D, 8-beam LiDAR



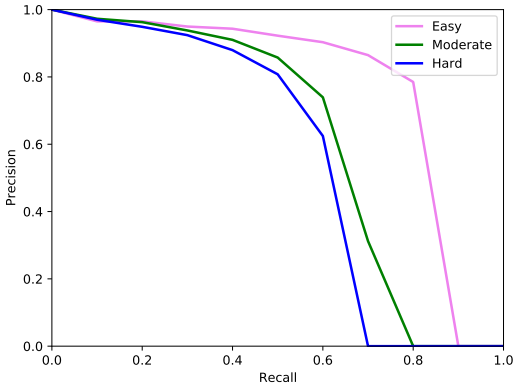
(e) BEV, 16-beam LiDAR



(f) 3D, 16-beam LiDAR



(g) BEV, 64-beam LiDAR



(h) 3D, 64-beam LiDAR

FIGURE 6.7: **Precision-Recall (P-R) curves.**

Table 6.4 is divided into two separate parts and contains the 3D obstacle detection performance results for IOU of 0.7. In each cell of this table, a pair of numbers A/B corresponds to the results obtained with the AP_{bev}/ AP_{3D} metrics as explained in the experimental part of Chapter 5. On the left side of the table, we consider the stereo and the different LiDARs taken separately. We note that when the sensors are taken separately, the stereo provides the following results: 82%, 65% and 57% if we go from Easy to Hard. If we consider the LiDARs taken separately, we can see without surprise that it is the 64-beam that provides the best results: 87%, 75% and 69% if we go from Easy to Hard. If we consider the progression of the detection as a function of the number of layers, we observe an almost linear progression from 4 to 64 beams. As we seek to improve detection, we merged stereo with the different types of LiDARs. The results appear in the right part of Table 6.4.

In the fusion, we note that the detection performance is improved (when using a stereo camera alone) regardless of the combination of stereo camera with LiDAR. However, the fusion results are not significantly improved when passing from one number of layers to another (4 to 64 beams). This is because that the performance results of our model are only related to the KITTI datasets. So, to better analyze the contribution of each component, particularly when changing the number of beams, we need to test the proposed detection model with different datasets than KITTI. In any case, the best solution is always obtained by fusing both sensors. This proves, that each component of SLS-fusion architecture effectively contributes to the final performance of the model, and we cannot eliminate these components of the neural network architecture in all possible cases: low cost, medium cost or high cost LiDAR sensors.

		Easy	Moderate	Hard		Easy	Moderate	Hard
Sensors	S	82.38/ 68.08	65.42/ 50.81	57.81/ 46.07	S	82.38/ 68.08	65.42/ 50.81	57.81/ 46.07
	L4	56.72/ 38.82	49.25/ 32.02	44.14/ 29.75	S+L4	87.51/ 76.67	76.88/ 63.90	73.55/ 56.78
	L8	84.55/ 68.75	65.68/ 50.39	58.78/ 45.75	S+L8	87.52/ 76.67	76.96/ 63.99	73.63/ 56.95
	L16	85.15/ 70.01	68.70/ 52.55	60.13/ 47.49	S+L16	87.74/ 76.88	76.98/ 64.10	73.91/ 57.05
	L64	87.83/ 75.44	75.75/ 60.84	69.07/ 55.95	S+L64	88.06/ 77.44	77.18/ 64.84	74.33/ 57.25

TABLE 6.4: **Evaluation of the 3D object detection part of our method for different input sensors.** This result is obtained with IoU equal to 0.7.

6.4 Conclusion

In this chapter, we have analyzed in an ablation study, the contribution of a stereo camera and different versions of LiDAR (4 to 64 beams) to the performance of the SLS-fusion model in detecting 3D obstacles. Based on our ablation analysis and the different measurements used to evaluate our detection algorithms, we have proved that sensors are always unseparated for better performance. Quantitative results

have shown that detection performance drops reasonably with each component disabled (stereo camera or LiDAR) or by modifying the number of LiDAR beams, and the full model works best. In conclusion, SLS-fusion is an effective obstacle detection solution for low and high cost LiDARs when combined with a stereo camera and the optimal solution is achieved with the most economical 4-beam LiDAR component. However, to better generalize the model and to find the optimal solution between the performance of detecting obstacles and the price of the LiDAR component, we need to test our model in many different datasets and environments.

Chapter 7

Conclusion and Future Directions

Contents

Chapter overview: In this chapter, the primary focus is to summarize and discuss the major findings of this thesis, which involves linking together the various tasks executed and the results achieved. Additionally, the limitations of the proposed approaches are discussed, and future research directions are recommended to conclude this thesis.

7.1 Discussion

The objective of this thesis was to investigate the capability of detecting obstacles for autonomous vehicles using existing methods from the literature (as discussed in Chapter 2). We tested the best combination of sensors to detect obstacles in both favorable and foggy weather conditions, and found that a combination of stereo camera and LiDAR sensors yielded the best results. To test this, we used a well-known KITTI dataset that includes images and point cloud data, and tested LiDAR with a varying number of beams (from 4 to 64). We also tested the ability to detect and recognize objects in foggy conditions, which can be challenging to obtain data for. To address this, we carried out a foggy augmentation procedure on the original KITTI dataset and proposed a novel foggy dataset, called **Multifog KITTI dataset**, with a fog intensity ranging from a visibility of 20 m to 80 m.

In Chapter 4, we introduced **SLS-Fusion**, a novel depth prediction framework for 3D object detection that takes low-cost sensors - 4-beam LiDAR and stereo images - as inputs in the autonomous driving domain. Few studies have fused stereo cameras and LiDAR in a deep neural network for the specific task of 3D object detection,

so our proposed depth estimation neural network is a novel approach. It takes advantage of features from both images and point clouds by using an encoder-decoder network and optimizing the depth loss through a Depth Cost Volume for driving scenes. The predicted depth map (in 2D space) is then converted into a pseudo point cloud (in 3D space) using extrinsic calibration information between the camera and LiDAR. This pseudo point cloud can then be fed into LiDAR-based methods (such as PointRCNN) for object detection, just as a real point cloud would be. Experimental results on the KITTI dataset showed that this model can improve both depth estimation and 3D object detection accuracy.

In Chapter 5, we evaluated the performance of our 3D object detection algorithms using two types of sensors: a stereo camera and two versions of LiDAR (4-beam and 64-beam). The developed algorithms were evaluated on the KITTI dataset as well as an additional dataset with simulated fog (Multifog KITTI). We analyzed the contribution of these two types of sensors to object detection performance under both favorable and foggy weather conditions, both when they were fused and when they were used separately. The main finding was that using LiDAR in foggy weather resulted in slightly worse object detection performance, particularly when the LiDAR was a 4-beam laser sensor. In contrast, the results based on the stereo camera were promising in foggy weather conditions, regardless of the level of visibility.

Our research aimed not only to detect obstacles effectively but also to find a cost-effective sensor fusion solution for autonomous vehicle manufacturers. We found that a combination of cameras and LiDAR can be effective, but not in all configurations. While the camera can be dominant when paired with the correct version of LiDAR, LiDAR is significantly affected by foggy weather conditions. Therefore, in Chapter 6, we sought to identify the best combination of stereo camera and LiDAR sensors by testing LiDAR with 4, 8, 16, and 64 beams, to find a balance between good 3D obstacle detection and a reasonable system cost.

It's important to remember that the results of this study are based on camera and real LiDAR data that were originally acquired in good weather conditions, to which simulated fog was added. The model used in this study is a simple model that only considers the macroscopic attenuation phenomenon, and it has been calibrated on tests conducted in a controlled environment using standard sensors.

7.2 Limitations

The proposed methods have demonstrated efficiency in terms of performance; however, some limitations still exist that require further research to be addressed:

- Although pseudo point clouds have a higher density, in this thesis, we used PointRCNN, which converts pseudo point clouds to 64-beam point clouds, thus negating the benefit of using pseudo point clouds.

- It is necessary to confirm the validity of the methods for the sensors used in the KITTI dataset, as the impact of fog can vary depending on the sensor's brand, type, and internal settings. For example, the threshold effects when converting raw signals to point clouds with LiDAR sensors can significantly impact the model we are using. The camera's exposure setting is not taken into consideration in the current model and may also impact the results.
- The model does not account for microscopic light diffusion phenomena, such as halo effects for cameras and backscattering effects for LiDAR sensors. These limitations are explicitly mentioned and can affect the results. Therefore, the presented results should be considered initial findings useful for comparing sensors and finding data fusion solutions for autonomous vehicles, rather than definitive conclusions.
- The realism of the simulated fog depends on the quality of the depth map information. Inaccurate depth maps will result in unrealistic fog.

7.3 Perspectives

Our future work will focus on expanding the evaluation and refinement of our proposed method beyond KITTI dataset. We plan to train and test our method on other datasets, which will provide a more comprehensive assessment of its performance. We also aim to evaluate our method for other classes such as pedestrians and cyclists, to improve its generalization ability.

To improve the performance of our model in foggy weather conditions, we plan to explore several potential options. One option is to incorporate active sensing for fusion, which can improve the accuracy of depth maps in adverse weather conditions. Another option is to use multi-modal data augmentation to simulate different weather and lighting conditions to train our model. Additionally, we plan to investigate adaptive fusion for dynamic environments, which includes online learning for fusion, adaptive sampling for fusion, and dynamic fusion architectures. These directions could be interesting areas of study for future research.

Appendix A

Datasets

A1. Multifog dataset

This is Multifog KITTI dataset presented in Chapter 3. This dataset contains a training part (7,481 frames) and a testing part (7,518 frames) for image RGB left (cam 2) (.png, 4.3 GB), image RGB right (cam 3) (.png, 4.1 GB), 64-beam LiDAR Velodyne (.bin, 10.7 GB), 4-beam LiDAR Velodyne (.bin, 98 MB), file annotation for each frame visibility (.txt, 20 m to 80 m visibility level).

Link to download: https://maiminh1996.github.io/publications/multifogkitti.html

A2. Sparse LiDAR KITTI datasets

This is a dataset created based on the KITTI dataset that we use in this thesis under favorable weather conditions. This dataset contains a training part (7,481 frames) and a testing part (7,518 frames) for 4-beam, 8-beam, 16-beam and 32-beam LiDAR point cloud (.bin).

Link to download:
https://maiminh1996.github.io/publications/sparse_lidar_kitti_datasets.html

Appendix B

Version française résumée

Chapitre 1
Introduction

Aujourd'hui, la tendance générale est à l'automatisation des systèmes pour réduire les coûts et améliorer la sécurité. En général, dans les systèmes développés, il y a une partie matérielle (capteurs de détection, de surveillance, etc) et une partie logicielle (traitement des données avec des techniques avancées d'intelligence artificielle). Les applications industrielles utilisent des capteurs tels que des caméras, RADAR (Radio Detection and Ranging), Kinect, LiDAR (Light Detection and Ranging), IMU (Unité de mesure inertielle) pour collecter des données avant d'être traitées par des algorithmes complexes. La recherche dans cette thèse se concentre principalement sur les voitures autonomes, où les caméras et le LiDAR jouent un rôle essentiel dans la perception de l'environnement autour de la voiture. Les caméras et autres capteurs sont souvent intégrées sur le véhicule à sa conception.

Dans notre cas, pour les véhicules autonomes, du fait de l'absence de conducteur, la fonction de perception de l'environnement dans lequel évolue le véhicule est très importante. Ainsi, le véhicule autonome doit se rendre d'un point A à un point B en toute sécurité. La surveillance de l'environnement se fait à l'aide d'une série de capteurs, dont la vidéo et le LiDAR sont les plus couramment utilisés.

La perception de l'environnement devant les véhicules autonomes est très importante, car le contexte sécuritaire est très élevé. Dans cette recherche, nous avons choisi de développer un système de détection d'obstacles basé sur une caméra stéréoscopique et un LiDAR. Dans la littérature, il est montré que la combinaison des deux conduit à de bons résultats de détection d'obstacles (voir chapitre 2 : état

de l'art). De plus, ces dernières années, la détection d'objets 3D a été exploitée largement dans l'industrie et dans le milieu académique, et en particulier ses diverses applications dans de nombreux domaines tels que la conduite autonome justement. Avec l'avancement de l'apprentissage approfondi et des réseaux de neurones convolutifs (CNN), les techniques de détection d'objets 3D ont réalisé des progrès remarquables grâce à des architectures de réseaux de neurones très avancés.

Par conséquent, ce travail se concentre sur la détection d'objets 3D avec caméra et capteurs LiDAR. Une dimension supplémentaire abordée dans cette thèse est la possibilité de détection des obstacles 3D dans des conditions météorologiques avec présence de brouillard.

Le travail de thèse est organisé de la manière suivante :

Dans le chapitre 2, un état de l'art est réalisé et concerne la possibilité de détecter des obstacles en 3D devant des véhicules autonomes. La discussion porte sur le choix des méthodes de la littérature concernant à la fois la combinaison de capteurs à utiliser et les algorithmes de traitement à appliquer. Pour chaque solution trouvée, la discussion est menée, ce qui nous a aidés à faire nos choix pour notre système.

Le chapitre 3 est consacré à la description du jeu de données KITTI que nous avons utilisé pour nos développements. En effet, l'absence de données réelles contenant le brouillard nous conduit à utiliser des bases de données académiques, contenant des vérités terrain. Grâce à ces jeux de données, il est également possible de comparer, avec l'état de l'art, nos algorithmes et architectures en termes de performances de détection d'obstacles. Comme notre objectif principal est de développer des algorithmes de détection d'objets 3D dans des conditions de brouillard, le processus d'application du brouillard sur les données KITTI est également décrit dans ce chapitre.

Le chapitre 4 présente SLS-Fusion, une nouvelle approche pour fusionner les données d'un LiDAR à 4 nappes et d'une caméra stéréo via un réseau de neurones pour l'estimation de la profondeur afin d'obtenir de meilleures cartes de profondeur denses et d'améliorer ainsi les performances de détection d'objets 3D. Étant donné que le LiDAR à 4 nappes est moins cher que le LiDAR à 64 nappes bien connu, cette approche est également classée comme une méthode basée sur des capteurs à faible coût. Dans ce chapitre, nous trouverons la description de l'architecture du réseau SLS-Fusion, sa mise en œuvre et la manière de prendre en compte simultanément les données vidéo et les données LiDAR. Une évaluation avec des métriques spécifiques est effectuée sur le jeu de données KITTI sans brouillard dans un premier temps.

Dans le chapitre 5, nous analysons les effets du brouillard sur la détection d'objets dans les scènes pour ensuite proposer des méthodes d'amélioration. La collecte et le traitement de données dans des conditions météorologiques défavorables sont souvent plus difficiles que des données dans de bonnes conditions météorologiques.

Par conséquent, un jeu de données synthétique qui simule de mauvaises conditions météorologiques est un bon choix pour valider une méthode, car il est plus simple et plus économique, avant de travailler avec un jeu de données réel. Dans ce chapitre, nous appliquons du brouillard sur le jeu de données public KITTI (expliqué au chapitre 3) pour générer ce que nous appelons le jeu de données Multifog KITTI pour les images et les nuages de points. En termes de tâches de traitement, nous testons notre précédent détecteur d'objets 3D basé sur LiDAR et caméra, nommé Sparse LiDAR Stereo Fusion Network (SLS-Fusion), pour voir comment il est affecté par des conditions météorologiques en présence de brouillard.

Dans le chapitre 6, une étude d'ablation est menée pour évaluer les rôles respectifs de la caméra et du LiDAR pour la détection d'obstacles. Ainsi différentes combinaisons de caméra et plusieurs types de LiDAR (4 nappes, 8 nappes. . . .64 nappes) sont testés pour bien analyser les complémentarités entre les deux types de capteurs. Une très courte analyse économique est également réalisée.

Enfin au chapitre 7, un bref rappel de tous les résultats obtenus est mentionné. Les enseignements tirés de ce travail sont décrits et des perspectives à court terme proposées.

Chapitre 2
Détection d'objets 3D : l'état de l'art

La détection d'objets est l'une des principales composantes de la vision par ordinateur. Elle a pour but de détecter et de classer des objets dans des images. Elle permet proposer de nombreuses applications directement liées à la vie : détection d'objets dans l'industrie, comptage de piétons, conduite autonome, détection d'anomalies, détection de visages, etc. La détection d'objets est plus difficile lorsqu'il faut reconnaître des objets et les localiser dans un espace 3D. En particulier, la détection d'objets 3D appliquée à la conduite autonome est un sujet en pleine expansion.

L'automatisation de la conduite repose sur de nombreux aspects, tels que la perception, le positionnement, l'analyse de scénarios, la prise de décision, le contrôle commande... Le sujet de cette thèse porte uniquement sur la partie perception. Plus précisément, elle vise à utiliser de la fusion de données des capteurs du véhicule afin d'améliorer les niveaux de détection des véhicules voisins, notamment dans des conditions météorologiques dégradées. Les LiDARs, radars et caméras sont couramment utilisés à bord des véhicules. La thèse se limite à l'utilisation conjointe des caméras et des LiDARs, car ce sont les capteurs qui permettent le mieux la détection des usagers vulnérables de la route et que ce sont les capteurs les plus impactés par les conditions météorologiques dégradées. C'est donc sur cette combinaison de capteurs que notre contribution peut être pertinente. Avant de proposer une nouvelle méthode, il est important de présenter l'état de l'art sur les méthodes de détection 2D et 3D.

Un détecteur d'objets 2D produit, pour chaque objet d'intérêt dans une image, une boite englobante 2D et une classe pour l'objet détecté. Au cours des 20 dernières années, les technologies de détection d'objets ont connu une évolution importante. Dans le passé, les chercheurs utilisaient des méthodes classiques pour détecter les objets avant que l'apprentissage profond ne soit développé. Pour cette raison, la détection d'objets 2D se divise en deux catégories : le traitement d'images traditionnel et les méthodes basées sur l'apprentissage profond. Les méthodes traditionnelles utilisent généralement des caractéristiques (features) créées à la main pour la reconnaissance et la détection des objets. Pour cette raison, ces méthodes se retrouvent

souvent limitées dans des scénarios complexes (ex : occlusions). Contrairement aux méthodes traditionnelles, les récentes méthodes basées sur l'apprentissage profond tirent parti de la puissance de calcul des cartes graphiques et d'une grande quantité de données pour apprendre puis extraire de bonnes caractéristiques dans l'image. Cela permet d'améliorer considérablement les performances de l'algorithme de détection 2D. Cependant, les détecteurs d'objets 2D sont difficilement adaptables pour détecter des objets en 3D. En effet, l'information de profondeur n'est pas présente, ce qui est crucial pour une application de conduite.

Dans le domaine de la conduite autonome, la détection d'objets 3D est primordiale, car elle fournit l'ensemble des informations nécessaires à la décision, telles que l'emplacement, la direction et la taille. Cette thèse se concentre uniquement sur l'utilisation de caméra et de LiDAR. Sur la base de ces capteurs, la détection d'objets 3D a généralement été abordée de trois façons dans la littérature : caméra uniquement (monoculaire ou stéréoscopique), LiDAR uniquement et fusion caméra-LiDAR.

La détection d'objets par caméra est un domaine de recherche de longue date. En effet, les caméras sont peu chères et fournissent beaucoup d'informations sur les objets : texture, couleur, bords, etc. Cependant, l'image manque d'informations sur la profondeur, ce qui est extrêmement important pour la détection 3D. Même avec l'utilisation d'une caméra stéréoscopique, la carte de profondeur n'est pas suffisamment précise. A cause de cela, la recherche s'est progressivement ré-orientée vers le capteur LiDAR, qui apporte une meilleure précision concernant l'estimation des distances.

Le LiDAR permet de reconstruire un nuage de points 3D de la scène. Ce nuage de point contient une multitude d'informations sur la géométrie, la forme et la taille. Cela permet d'extraire des caractéristiques pertinentes qui améliorent les performances de détection. Cependant, le LiDAR possède de nombreuses limites : détection d'objets isolés difficile, surtout à longue distance, surfaces vitrées invisibles, manque d'informations sur les couleurs. De plus, les approches d'apprentissage profond les plus efficaces pour la détection d'objets nécessitent que les données soient ordonnées et structurées (comme dans une image), ce qui n'est pas le cas des nuages de points. En raison de leur nature irrégulière, les nuages de points sont souvent traités de l'une des trois manières suivantes : projection du nuage sur un plan pour générer une pseudo-image régulière (view-based), échantillonnage des points dans des cellules appelées voxels (voxel-based), ou ré-encodage du nuage de points brut (point-based).

Comme la première partie de l'état de l'art le montre, la caméra et le LiDAR présentent tous les deux des avantages et des limites. Ils sont cependant très complémentaires. De nouvelles méthodes consistent donc à fusionner des données issues des

LiDARs et des caméras. C'est d'ailleurs le sujet principal de la thèse. Dans la littérature, il existe trois principales méthodes de fusion :

- la fusion précoce (early), où les données brutes sont fusionnées dès le départ pour former des tenseurs de données à nombreux canaux,
- la fusion tardive (late), où la fusion a lieu au niveau de la décision, après avoir traitées les données brutes issues des deux capteurs de manière indépendante,
- et la fusion profonde (deep), où la fusion est soigneusement construite pour combiner les avantages des systèmes de fusion précoce et tardive.

Pour éviter tout problème lié aux conditions météorologiques, les voitures autonomes sont initialement utilisées et testées dans des endroits ensoleillés pendant la journée. Cependant, pour pouvoir commercialiser une voiture à conduite autonome de niveau 5, le véhicule doit être capable de se déplacer en toutes circonstances, y compris dans différentes conditions météorologiques. Les méthodes présentées obtiennent de bons résultats dans des conditions météorologiques favorables. Cependant, la perception dans des conditions météorologiques défavorables est encore limitée et il reste de nombreux défis à résoudre. La pluie, la neige et le brouillard peuvent tous perturber la vision des capteurs, tout comme ils peuvent affecter les conducteurs humains. Nous nous concentrerons sur les conditions de brouillard dans cette thèse car elles ont un grand impact sur les systèmes de vision artificielle. Concernant les travaux sur la vision artificielle et le brouillard, il existe un beaucoup de travaux de recherche sur la détection et la classification du brouillard. Cependant, il n'y a pas beaucoup de travaux sur la prise en compte du brouillard dans des méthodes de détection d'objets 2D ou 3D. Pourtant, quelques travaux de recherches antérieurs ont démontré que la performance diminue avec le brouillard pour différentes applications comme la segmentation, la détection d'objets 2D ou encore l'estimation de la profondeur. L'absence de la prise en compte du brouillard dans les travaux sur la détection d'objets appliquée au véhicule autonome vient du fait qu'il est difficile d'obtenir des jeux de données avec du brouillard. Dans le cas général, la disponibilité d'énormes ensembles de données étiquetées a contribué de manière significative à l'avancement de la vision par ordinateur au cours des dernières années. En revanche, acquérir et annoter un tel ensemble de données pour chaque nouveau problème (par exemple, le brouillard) serait trop consommateur de temps. Pour répondre à cela, deux approches sont actuellement explorées :

- Le Cerema propose la plateforme PAVIN Brouillard et Pluie [194], qui permet de reproduire différents niveaux de brouillard et de pluie, dans un environnement parfaitement contrôlé et reproductible. Les dimensions de l'enceinte permettent de reproduire des scènes routières pour les tests de LiDARs et de caméras embarquées.
- L'ajout de conditions météorologiques dégradées simulées numériquement sur des images initialement acquises par temps clair [83, 98].

Dans ce chapitre, nous avons vu comment les algorithmes basés sur l'apprentissage profond ont montré des performances exceptionnelles pour la détection d'objets 3D ces dernières années. Notre objectif en réalisant cet état de l'art est de mieux comprendre les modèles d'apprentissage profond utilisés dans la détection d'objets 3D des véhicules autonomes. Sur la base de plus de 100 publications, avec des évaluations détaillées de plusieurs architectures, utilisant des LiDARs et/ ou des caméra, nous pouvons identifier les meilleures architectures pour cette tâche.

On constate que les méthodes basées sur le LiDAR obtiennent les meilleures performances exceptionnelles [64, 195]. Parallèlement, la méthode Pseudo-LiDAR [43] ouvre de nouvelles voies pour les méthodes basées sur les caméras, tandis que la méthode Pseudo-LiDAR++ [46] est obtenue une précision au meilleur niveau de l'état de l'art. Les méthodes basées sur la fusion n'ont pas encore donné de meilleurs résultats car elles utilisent plus de données qu'il est difficile d'utiliser conjointement tant elles sont différentes.

Nous avons donc utilisé ces éléments pour guider la nouvelle architecture que nous présentons dans le chapitre 4. Afin d'exécuter des algorithmes d'apprentissage profond, il est nécessaire de disposer de grandes bases de données, avec ou sans brouillard. Le jeu de données que nous utiliserons dans cet article sera présenté dans le chapitre suivant.

Chapitre 3
Jeu de données utilisé

Cette section présente les bases de données qui seront utilisées tout au long de cette thèse. Avant cela, la première étape consiste à explorer et à analyser les bases de données disponibles publiquement. En effet, il est important d'utiliser des bases de données académiques, qui permettront de se comparer aux autres méthodes de la littérature de façon impartiale.

Les bases de données recherchées doivent répondre aux critères suivants : appliquées à la détection d'objet 3D pour le véhicule autonome ; comprenant des images de caméra stéréoscopique et de LiDAR avec au minimum quatre nappes ; idéalement avec des conditions météorologiques défavorables. En effet, l'objectif de la thèse est de proposer une méthode basée sur des capteurs bas coûts, qui serait capable de fonctionner par temps clair tout aussi bien qu'en conditions météorologiques défavorables (comme le brouillard par exemple).

L'état de l'art réalisé montre que la plupart des bases de données contiennent uniquement des données acquises en conditions météorologiques favorables. Seuls quelques bases de données comme Waymo [126] ou nuScenes [123] possèdent des images acquises dans toutes sortes de conditions météorologiques. Leur inconvénient est que les conditions météorologiques ne sont alors pas définies précisément (ex : intensité du brouillard ou de la pluie). Il sera donc nécessaire de faire appel à une base de données de la littérature sans conditions météorologiques défavorables, avant de simuler par-dessus des conditions de brouillard.

La méthode de détection d'objet 3D proposée dans la suite de la thèse comporte des étapes successives dont l'estimation de la profondeur sur les images stéréo puis la fusion de données du LiDAR et de la caméra stéréo. Pour l'étape d'estimation de profondeur de la caméra stéréo, un premier apprentissage est effectué sur la base de données Scene Flow [144]. Cette base de données présente l'intérêt d'une grande base de données pour faire le premier apprentissage.

La seconde phase de l'algorithme proposé consiste à fusionner les données issues de la caméra stéréoscopique avec le LiDAR, puis à détecter les objets en 3D. Pour cela, la base de données KITTI [12] a été choisie. En effet, cette base de données présente de nombreux avantages :

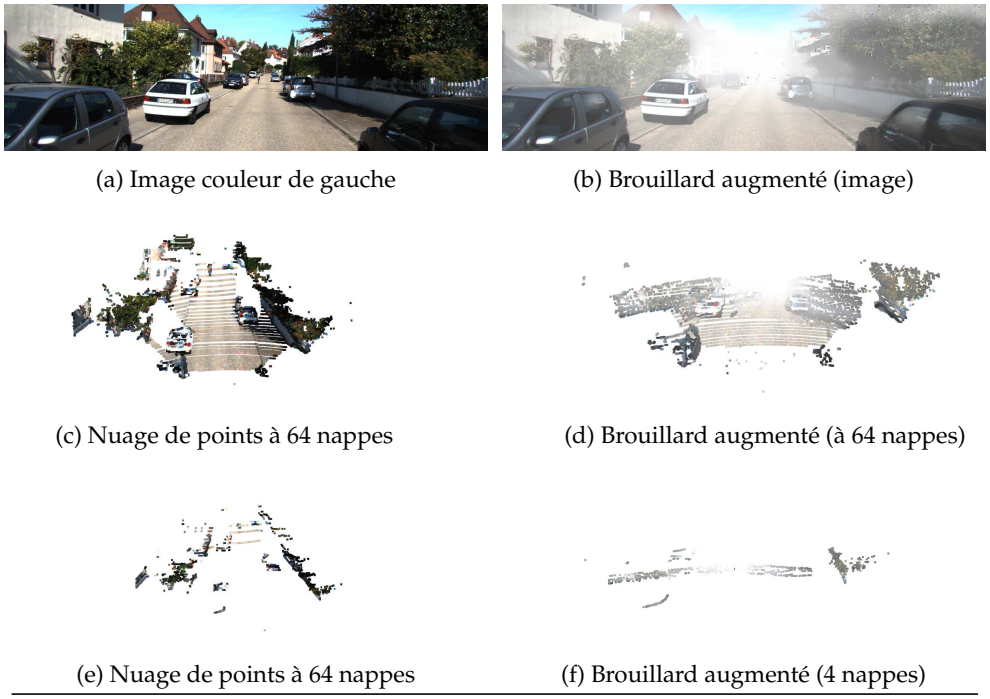
- elle est très largement utilisée ce qui permettra une comparaison de notre méthode à la littérature aisée ;
- elle est collectée dans des conditions météorologiques favorables uniquement, ce qui permettra d'être serein lors de l'ajout de brouillard simulé numériquement sur les images ;
- elle contient des données acquises par un LiDAR et une caméra stéréoscopique ;
- elle est appliquée à la détection 3D pour le véhicule intelligent ;
- elle contient des données de bonne qualité (filtrées, synchronisées, labellisées...)

Afin d'utiliser la base de données KITTI, plusieurs pré-traitements ont dû être effectués :

- D'abord, les labels ont été filtrés pour ne garder que les objets qui entrent dans les champs de caméra du LiDAR et de la caméra stéréoscopique à la fois. En effet, le champ de vision de la stéréo caméra est moins large que celui du LiDAR.
- Afin de permettre l'estimation de la profondeur dans notre algorithme, une carte de profondeur dense doit être obtenue comme vérité terrain. Pour obtenir ce nuage de point dense, les nuages de points de 11 images successives ont été accumulés.
- Au lieu d'utiliser le coûteux LiDAR 64 nappes, nous avons extrait un nuage de points 4 nappes du LiDAR 64 nappes original de la base de données KITTI. Cela permet de simuler un LiDAR bas coût.

De plus, la méthode proposée dans le cadre de cette thèse doit être robuste face au brouillard. Nous avons ainsi proposé une augmentation de la base de données KITTI, en simulant du brouillard sur chaque image de la base de données pour la caméra stéréoscopique et le LiDAR. Pour cela, nous avons rappelé la méthode de simulation de brouillard employée. Cette méthode est adaptée à des images issues de la caméra stéréoscopique mais aussi pour les données issues du LiDAR. La figure B.1 présente un exemple de données obtenues.

Pour conclure, après avoir présenté un état de l'art sur les bases de données, nous avons décidé d'utiliser la base de données KITTI qui est la plus répandue dans le domaine de la détection d'objet 3D appliquée au véhicule intelligent. Nous avons ensuite ajouté du brouillard sur cette base de données. Nous disposons donc de deux bases dans la cadre de la thèse « Clear KITTI » et « Multifog KITTI ». Ces bases, nous permettrons d'abord de développer notre méthode de détection d'objet 3D par fusion caméra stéréoscopique / LiDAR bas coût, avant de renforcer cette méthode dans le cas où il y a présence de brouillard.



(a) Image couleur de gauche

(b) Brouillard augmenté (image)

(c) Nuage de points à 64 nappes

(d) Brouillard augmenté (à 64 nappes)

(e) Nuage de points à 64 nappes

(f) Brouillard augmenté (4 nappes)

FIGURE B.1: **Exemple de KITTI à notre jeu de données Multifog KITTI proposé.** Un exemple de jeu de données KITTI comprend l'image couleur de gauche, le nuage de points à 64 nappes, le nuage de points à 4 nappes extrait et celui correspondant avec l'augmentation brumeuse (visibilité V = 45 m).

Chapitre 4
Fusion de capteurs pour la détection d'objects 3D

La perception de l'environnement d'un véhicule autonome passe par la reconnaissance des objets et par leur localisation tridimensionnelle précise, notamment en termes de distance par rapport au véhicule. Des études précédentes ont montré que les performances des approches reposant sur des LiDAR sont supérieures à celles qui utilisent des caméras, notamment grâce à la précision de l'information de profondeur apportée par ces capteurs. Mais leur inconvénient réside dans le coût qui s'avère beaucoup plus élevé que celui des caméras.

Ce chapitre décrit l'approche que nous proposons et qui repose sur l'idée initiale de fusionner les informations provenant d'un couple stéréoscopique de caméras avec celles provenant d'un LiDAR à bas coût. Plus précisément, il s'agit de remplacer l'utilisation d'un LiDAR à 64 nappes par un LiDAR à 4 nappes associé à un couple de caméras. Nous proposons l'architecture SLS-Fusion dont les grandes étapes sont décrites par la figure B.2.

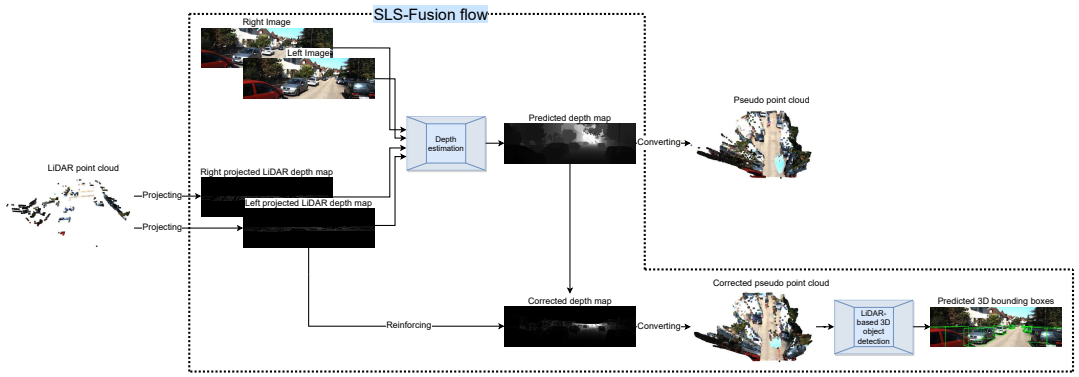


FIGURE B.2: **Vue d'ensemble schématisée de la méthode proposée SLS-Fusion pour la détection d'objets 3D.** Elle prend en entrée les images provenant des caméras ainsi que le nuage de points provenant du LiDAR à 4 nappes et délivre en sortie les boites englobantes 3D des objets détectés.

La première étape de la méthode SLS-Fusion consiste à estimer une carte de profondeur à partir des deux images provenant des caméras et des projections sur les deux images du nuage de points provenant du LiDAR. Les projections du nuage de points sur chacune des deux images sont effectuées grâce aux données de calibrage des capteurs. Ces projections constituent deux cartes éparses de profondeurs qui sont associées aux deux images provenant des caméras. Une carte de profondeur dense est obtenue grâce à une réseau de neurones inspiré de celui proposé dans [170] mais utilisant une fusion tardive. Il s'agit d'une architecture de type encodeur-décodeur. La partie encodage repose sur deux flux partageant leurs poids constitués de blocs de type ResNet, l'un traitant les images provenant des caméras, l'autre traitant les deux projections provenant du LiDAR. Dans la partie décodage, les données provenant de la caméra gauche et celle provenant de la projection sur l'image gauche sont fusionnées pour en extraire les caractéristiques. La même opération est effectuée pour les données provenant de la droite. Ces caractéristiques gauches et droites sont ensuite utilisées pour constituer un volume des coûts des disparités (DiCV), qui est ensuite transformé en un volume des coûts des profondeurs (DeCV), comme proposé dans [46], qui permet de prédire une carte de profondeur.

Dans un second temps, la carte dense de profondeur obtenue est corrigée en utilisant de nouveau le nuage épars de points provenant du LiDAR. En effet, la précision de la profondeur de ces points permet de compenser celle de la carte de profondeur initiale qui souffre de l'imprécision des disparités estimées en pixels. Pour effectuer cette correction, nous avons utilisé la méthode GDC décrite dans [46]. Cette approche repose sur l'utilisation d'un graphe représentant les K plus proches voisins de chaque point 3D et dont la détermination des poids revient à résoudre un problème d'optimisation quadratique sous contrainte. La correction des profondeurs est également réalisée en résolvant un autre problème d'optimisation quadratique sous contrainte. La carte des profondeurs corrigées est ensuite transformée en un pseudo-nuage de points grâce aux données de calibrage des capteurs.

Enfin, une méthode de détection d'objets 3D conçue pour être appliquée à un nuage de point provenant d'un LiDAR est appliquée au pseudo-nuage de points. Nous avons utilisé pour cela la méthode PointRCNN proposée dans [64].

Nous avons évalué les performances de l'approche proposée à la fois sur l'estimation de la profondeur et sur la détection des objets 3D. Le jeu de données que nous avons utilisé pour l'ensemble des évaluations est KITTI et les objets détectés étaient les voitures car, étant très largement majoritaires dans KITTI, ce sont les objets habituellement utilisés dans les évaluations. Concernant l'estimation de la profondeur, les évaluations ont montré l'apport du LiDAR par rapport à l'utilisation des caméras. De même, les résultats obtenus pour la détection des objets sont satisfaisant. La méthode proposée a été comparée à huit méthodes existantes qui utilisent des capteurs à bas coût ainsi qu'à la méthode PointRCNN [64] qui utilise le nuage de points issu d'un LiDAR à 64 nappes. La méthode SLS-Fusion s'avère un peu moins

performante que PointRCNN utilisant 64 nappes, mais elle obtient les meilleurs résultats sur plusieurs indicateurs par rapport aux autres méthodes reposant sur des capteurs à bas coût (stéréo caméra seule ou stéréo caméra associée à un LiDAR à 4 nappes).

Ces résultats ont été obtenus sur des données acquises par temps clair. Nous avons souhaité prendre en compte des conditions météorologiques beaucoup moins favorables, comme le brouillard : c'est l'objet du chapitre suivant.

Chapitre 5
Fusion de capteurs pour la détection d'objets 3D dans des conditions de brouillard

L'objectif de ce chapitre est d'évaluer une solution de fusion de caméras et LiDAR (4 et 64 nappes) pour la détection d'objets 3D par temps de brouillard. Dans le précédent chapitre, nous ne nous sommes pas souciés de l'effet de la météo et avions détecté les objets dans la base KITTI sans que celle-ci ne subisse l'influence du brouillard. Dans ce chapitre, nous analysons et évaluons l'impact de la météo (en particulier le brouillard) sur les capteurs et la capacité d'identifier les objets présents dans la scène sur la base de ces capteurs.

En réalité, il n'est pas toujours facile d'obtenir des données contenant du brouillard. C'est d'autant plus important selon le lieu d'expérimentation. Il y a des pays où il n'y a jamais brouillard. Il n'est donc pas facile d'avoir des ensembles de données complets dans des conditions météorologiques brumeuses. Une solution pour pallier cet inconvénient est de générer des données synthétiques contenant du brouillard. Ça implique d'appliquer un certain nombre de traitements de base aux données d'origine pour les entacher de brouillard (la procédure que nous avons suivie figure dans le chapitre 3). En termes de tâches de traitement, nous testons notre détecteur d'objets 3D basé sur LiDAR et caméra, nommé Spare LiDAR Stereo Fusion Network (SLS-Fusion), pour voir comment il est affecté par des conditions météorologiques avec brouillard. Nous proposons de faire des apprentissages en utilisant à la fois l'ensemble des données d'origine et l'ensemble de données fogifiées pour améliorer les performances par temps de brouillard tout en gardant de bonnes performances dans des conditions normales.

Dans l'analyse, nous cherchons à évaluer donc la contribution des deux types de capteurs dans notre modèle SLS-Fusion. Dans le modèle, les données des deux capteurs d'entrée sont fusionnées à un stade précoce et nous ne savons pas dans quelle mesure chaque capteur contribue aux résultats. Il convient tout d'abord de noter que les performances de la détection d'objets 3D sont considérablement réduites si l'on évalue sur les données avec brouillard avec une phase d'apprentissage

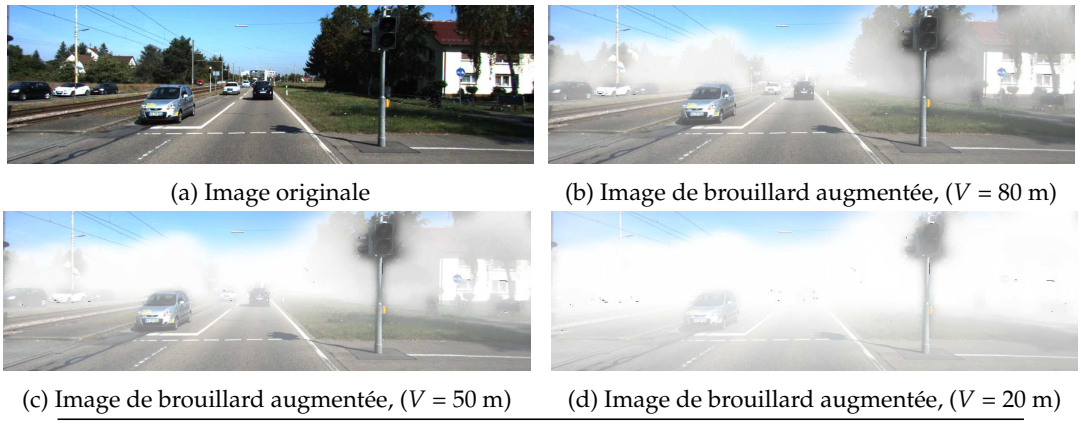
sur les données sans brouillard. Cela montre l'importance d'une bonne sélection des ensembles de données d'apprentissage. Ainsi, les performances de détection d'objets 3D sont réduites de 42,67% si le traitement se fait sur une base avec du brouillard alors que l'apprentissage se fait sur une base sans brouillard. En utilisant une stratégie spécifique d'apprentissage qui consiste à prendre en compte les données contenant du brouillard et les données sans brouillard (apprentissage conjoint sur le KITTI et le Multifog KITTI), les résultats sont significativement améliorés de 26,72% et la détection reste performante assez bien sur le jeu de données d'origine (le jeu de données Clear KITTI) avec une baisse de seulement 8,23% par rapport au cas de l'apprentissage sur le Clear KITTI et du test sur le Clear KITTI. En résumé, le brouillard provoque souvent l'échec de la détection 3D.

Dans cette partie qui concerne l'utilisation du jeu de données KITTI sans brouillard, on remarque, comme prévu, que la fusion de la caméra stéréo et du LiDAR donne de bons résultats : 93,02% pour les objets Easy et pour un IOU de 0,5. Dans cette configuration, une caméra stéréo donne de meilleurs résultats qu'un LiDAR 4 nappes (89,39% contre 78,16%). Séparer les deux capteurs entraîne une baisse des performances. Ceci nous amène à considérer les deux capteurs conjointement. D'autre part, si l'on considère le LiDAR 64 nappes, les résultats sont meilleurs (97.3% pour les objets Easy et un IOU de 0.5). Néanmoins, le prix du LiDAR 64 nappes est très élevé par rapport à celui des caméras et des LiDAR 4 nappes. Par la suite, nous avons calculé les mêmes indicateurs que précédemment, mais sur des données avec brouillard (Multifog KITTI). La performance de la fusion d'une caméra stéréo et d'un LiDAR à 4 nappes reste élevé (90,15%) pour les objets Easy, ce qui démontre la viabilité de l'utilisation de deux capteurs en même temps. Les performances du LiDAR 4 nappes seul diminuent fortement (13,43%), ce qui semble montrer que le LiDAR 4 nappes n'est pas utile dans le brouillard, et c'est la caméra stéréo qui a la contribution la plus élevée, car ses performances restent très élevées (89,36%). La combinaison d'une caméra stéréo et d'un LiDAR à 64 nappes donne de bons résultats (89,26% pour les objets Easy) et pour la plupart des types d'objets, mais la caméra stéréo tient toujours le rôle le plus important.

Le résultat principal à retenir est que l'utilisation du LiDAR par temps de brouillard entraîne une performance de détection d'obstacles légèrement dégradée (surtout sir le le LiDAR est un capteur laser à 4 nappes). D'autre part, les résultats basés sur une caméra stéréo sont prometteurs par temps de brouillard, quel que soit le niveau de visibilité. Et enfin, la fusion originale de la caméra stéréo et du LiDAR améliore un peu les résultats qu'avec la camera seule.

Pour les travaux futurs, deux options peuvent être envisagées pour contourner les limites du modèle utilisé ici. La première consisterait à réaliser des acquisitions sur site réel, sous conditions météorologiques défavorables. La deuxième option, plus prometteuse, serait d'utiliser des modèles 3D plus complexes mettant en œuvre des

phénomènes de diffusion microscopique et simulant le trajet complet de la lumière des objets vers le capteur et le capteur lui-même.



(a) Image originale (b) Image de brouillard augmentée, (V = 80 m)

(c) Image de brouillard augmentée, (V = 50 m) (d) Image de brouillard augmentée, (V = 20 m)

FIGURE B.3: **Exemple d'images avec différents niveaux de visibilité.** (a) montre l'échantillon original de l'ensemble de données KITTI [12]. (b), (c), (d) montre les images brumeuses augmentées avec une visibilité V égale à 80 m (β = 0.0375), 50 m (β = 0.0599) et 20 m (β = 0.1498) respectivement. β est le coefficient de densité (ou d'atténuation) du brouillard correspondant à la visibilité V.

Chapitre 6
Analyse du rôle de chaque détecteur (LiDAR, caméra) dans la détection d'objets 3D : étude d'ablation.

La détection d'objets 3D par fusion LiDAR et stéréo a toujours été un défi pour les chercheurs dans le domaine. Comme présenté aux chapitres 4 et 5, ce travail offre une méthode de fusion pour les caméras LiDAR et stéréo basée sur un réseau de neurones profond (SLS-fusion) pour la détection d'objets 3D. Tout d'abord, un décodeur-encodeur basé sur le réseau ResNet est conçu pour extraire et fusionner les fonctionnalités gauche / droite des images de la caméra stéréo et la carte de profondeurs LIDAR projetée. Par la suite, le réseau de décodeurs construit une carte de profondeur à gauche et à droite pour obtenir une profondeur finale corrigée. Une fois la carte de profondeurs dense attendue obtenue, un nuage de pseudo-points est généré à l'aide de caméras étalonnées. Enfin, une méthode LiDAR de détection des objets 3D (PointRCNN) est appliquée au pseudo-nuage de points prédit (voir Figure B.4).

De plus, le SLS-Fusion, qui utilise les travaux de raffinement de PoinRCNN pour obtenir des prédictions de boîtes 3D, a été testé sur les données publiques KITTI. L'expérimention exploitant KITTI et un LiDAR à 4 nappes à faible coût montre que la fusion SLS proposée surpasse les méthodes les plus avancées, en particulier le Pseudo-LiDAR++ présenté dans le chapitre 4. Toutefois, comparativement au détecteur PointRCNN d'origine qui utilisait LiDAR à 64 nappes, SLS-fusion performance est plus faible.

La supériorité du LiDAR à 64 nappes, utilisé sans fusion avec des caméras stéréo, est évidente car les capteurs LiDAR haute résolution peuvent fournir des informations de profondeur très précises. Les capteurs LiDAR très précis peuvent être extrêmement coûteux : un modèle à 64 nappes peut coûter environ 75000 dollars (USD). Dans ce cas, plus le nombre de nappes est élevé, augmentant ainsi la quantité de nuages ponctuels générés, plus le coût du capteur LiDAR est élevé (1000 dollars à 75000

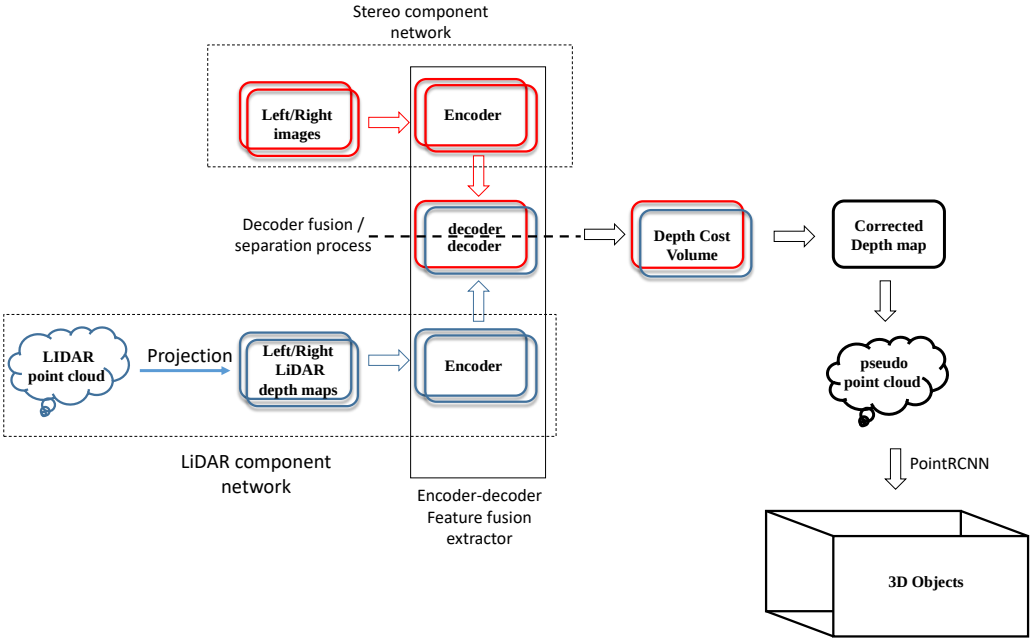


FIGURE B.4: **Structure globale du réseau de neurones SLS-Fusion**.

dollars). La solution optimale consiste à trouver un compromis entre un LiDAR pas trop coûteux et pouvant fournir des résultats de détection acceptables.

Dans ce chapitre et en fonction du nombre de nappes de LiDAR exploitées, nous avons essayé d'analyser comment les capteurs stéréo et LiDAR contribuent à la performance du modèle de SLS-Fusion sur la détection d'objets 3D. Comme le montre la figure B.4, pour séparer les composants du réseau SLS-Fusion qui représentent les architectures LiDAR et stéréo, il suffit de diviser le décodeur du modèle en réseaux de décodeurs indépendants. Le décodeur à l'intérieur du modèle SLS-fusion est le seul composant responsable de la fusion des fonctions entre les capteurs LiDAR et stéréo. L'une des techniques les plus connues utilisées à cette fin est appelée « étude d'ablation ».

Sur la base de notre analyse d'ablation et des différentes mesures utilisées pour évaluer nos algorithmes de détection, nous avons prouvé que les capteurs doivent être non séparés pour une meilleure performance. Les résultats quantitatifs ont montré que les performances de détection diminuent raisonnablement avec chaque composant désactivé (stéréo ou LiDAR) ou en modifiant le nombre de nappes de LiDAR.

Chapitre 7
Conclusion

L'objectif de ce travail de thèse a consisté à analyser la possibilité de détecter automatiquement des obstacles devant des véhicules autonomes. Pour cela, nous nous sommes inspirés des systèmes existants dans la littérature pour réaliser une telle fonction. La meilleure de combinaison de capteurs pour détecter des obstacles est basée sur la combinaison de la stéréo caméra et du LiDAR.

La combinaison des deux capteurs de détection a été testée sur un jeu de données de la littérature : KITTI qui est une base très connue et qui contient un certain nombre de scénarios d'obstacles avec les bonnes annotations. Ainsi, la combinaison stéréo caméra + LiDAR a été testée avec des LiDARs dont le nombre de nappes variait de 4 à 64 nappes.

Une autre dimension supplémentaire qui a été testée est la possibilité de détecter et reconnaitre des objets par temps de brouillard. Comme il n'est pas souvent facile d'avoir des données par temps de brouillard, une procédure de fogification de la base de données KITTI a été réalisé. Ainsi une intensité de brouillard (allant d'une visibilité de 20 m jusqu'á 80 m) a été appliquée sur la base KITTI. Logiciellement, il a fallu aussi concevoir un réseau de neurones qui tienne compte à la fois des données vidéo et des données LiDAR et y compris par temps de brouillard.

Dans cette recherche, nous avons cherché aussi, outre une bonne détection d'obstacles, une combinaison de capteurs qui ne soit pas trop onéreuse pour les constructeurs de véhicules autonomes. Nous savons que la combinaison stéréo caméra et LiDAR est bonne, mais pas dans toutes les configurations. La stéréo caméra est prépondérante mais lorsqu'elle est accompagnée de la bonne information de LiDAR. Ainsi, on constate que le LiDAR 4 nappes étatit inutile par temps de brouillard. Le travail mené dans le chapitre 6 a consisté à analyser quelle est la meilleure combinaison vidéo et LiDAR. Ainsi les LiDARs 4, 8, 16 et 64 nappes ont été testés. L'objectif est de trouver un compromis entre une bonne détection d'obstacle 3D avec un coût du système raisonnable.

Bibliography

[1] Sunil Singh, Umang Ahuja, Munish Kumar, Krishan Kumar, and Monika Sachdeva. "Face mask detection using YOLOv3 and faster R-CNN models: COVID-19 environment". In: *Multimedia Tools and Applications* 80.13 (2021), pp. 19753–19768. DOI: 10.1007/s11042-021-10711-8.

[2] Meygen Cruz, Jefferson James Keh, Ramiel Deticio, Carl Vincent Tan, John Anthony Jose, and Elmer Dadios. "A People Counting System for Use in CCTV Cameras in Retail". In: *Proceedings of IEEE International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. 2020, pp. 1–6. DOI: 10.1109/HNICEM51456.2020.9400048.

[3] Sang-gil Lee, Jae Seok Bae, Hyunjae Kim, Jung Hoon Kim, and Sungroh Yoon. "Liver Lesion Detection from Weakly-Labeled Multi-phase CT Volumes with a Grouped Single Shot MultiBox Detector". In: *Proceedings of Medical Image Computing and Computer Assisted Intervention (MICCAI)*. 2018, pp. 693–701. DOI: 10.1007/978-3-030-00934-2_77.

[4] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. "FCOS3D: Fully Convolutional One-Stage Monocular 3D Object Detection". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 2021, pp. 913–922. DOI: 10.1109/ICCVW54120.2021.00107.

[5] Ali Mansour, Isabelle Leblond, Denis Hamad, and Luis Felipe Artigas. "Sensor Networks for Underwater Ecosystem Monitoring and Port Surveillance Systems". In: *Sensor Networks for Sustainable Development*. CRC Press, 2017, pp. 431–467.

[6] Alex Davies. *INFOGRAPHIC – Autonomous Vehicle Landscape*. 2021. URL: https://www.selfdrivingcars360.com/infographic-autonomous-vehicle-landscape/.

[7] Eric Paul Dennis. *AV Deployment Timeline*. 2021. URL: https://www.cargroup.org/av-deployment-timelines/.

[8] On-Road Automated Driving (ORAD) committee. *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. SAE International, 2021, pp. 9–50. DOI: 10.4271/J3016_202104.

[9] De Jong Yeong, Gustavo Velasco-Hernandez, John Barry, and Joseph Walsh. "Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review". In: *Sensors* 21.6 (2021). DOI: 10.3390/s21062140.

[10] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. "The Pascal Visual Object Classes (VOC) Challenge". In: *International Journal of Computer Vision (IJCV)* 88.2 (2010), pp. 303–338. DOI: 10.1007/s11263-009-0275-4.

[11] 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: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2014, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for autonomous driving? The KITTI vision benchmark suite". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012, pp. 3354–3361. DOI: 10.1109/CVPR.2012.6248074.

[13] Paul Viola and Michael Jones. "Rapid object detection using a boosted cascade of simple features". In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2001, pp. I–I. DOI: 10.1109/CVPR.2001.990517.

[14] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2005, pp. 886–893. DOI: 10.1109/CVPR.2005.177.

[15] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788. DOI: 10.1109/CVPR.2016.91.

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "SSD: Single Shot MultiBox Detector". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2016, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. "Focal Loss for Dense Object Detection". In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[18] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". 2018. arXiv: 1804.02767.

[19] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal Speed and Accuracy of Object Detection". 2020. arXiv: 2004.10934.

[20] Chien-Yao Wang, I.-Hau Yeh, and Hong-Yuan Mark Liao. "You Only Learn One Representation: Unified Network for Multiple Tasks". 2021. arXiv: 2105.04206.

[21] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 580–587. DOI: 10.1109/CVPR.2014.81.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 37.9 (2015), pp. 1904–1916. DOI: 10.1109/TPAMI.2015.2389824.

[23] Ross Girshick. "Fast R-CNN". In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[24] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 39.6 (2017), pp. 1137–1149. DOI: 10.1109/TPAMI.2016.2577031.

[25] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. "Feature Pyramid Networks for Object Detection". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 936–944. DOI: 10.1109/CVPR.2017.106.

[26] Jianfeng Wang and Xiaolin Hu. "Convolutional Neural Networks with Gated Recurrent Connections". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 44.7 (2021), pp. 3421–3435. DOI: 10.1109/TPAMI.2021.3054614.

[27] Joseph Redmon and Ali Farhadi. "YOLO9000: Better, Faster, Stronger". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6517–6525. DOI: 10.1109/CVPR.2017.690.

[28] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. "Selective Search for Object Recognition". In: *International Journal of Computer Vision (IJCV)* 104.2 (2013), pp. 154–171. DOI: 10.1007/s11263-013-0620-5.

[29] Charles R. Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J. Guibas. "Frustum PointNets for 3D Object Detection from RGB-D Data". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 918–927. DOI: 10.1109/CVPR.2018.00102.

[30] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The 2005 DARPA Grand Challenge: The Great Robot Race*. Springer, 2007. DOI: 10.1007/978-3-540-73429-1.

[31] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Celine Teuliere, and Thierry Chateau. "Deep MANTA: A Coarse-to-Fine Many-Task Network for Joint 2D and 3D Vehicle Analysis from Monocular Image". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1827–1836. DOI: 10.1109/CVPR.2017.198.

[32] Zengyi Qin, Jinglu Wang, and Yan Lu. "MonoGRNet: A Geometric Reasoning Network for Monocular 3D Object Localization". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* 33.01 (2019), pp. 8851–8858. DOI: 10.1609/aaai.v33i01.33018851.

[33] Peixuan Li, Huaici Zhao, Pengfei Liu, and Feidao Cao. "RTM3D: Real-Time Monocular 3D Detection from Object Keypoints for Autonomous Driving". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020, pp. 644–660. DOI: 10.1007/978-3-030-58580-8_38.

[34] Zechen Liu, Zizhang Wu, and Roland Tóth. "SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2020, pp. 4289–4298. DOI: 10.1109/CVPRW50498.2020.00506.

[35] Yingjie Cai, Buyu Li, Zeyu Jiao, Hongsheng Li, Xingyu Zeng, and Xiaogang Wang. "Monocular 3D Object Detection with Decoupled Structured Polygon Estimation and Height-Guided Depth Estimation". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* 34.07 (2020), pp. 10478–10485. DOI: 10.1609/aaai.v34i07.6618.

[36] Ivan Barabanau, Alexey Artemov, Evgeny Burnaev, and Vyacheslav Murashkin. "Monocular 3D Object Detection via Geometric Reasoning on Keypoints". In: *Proceedings of the International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP)*. 2020, pp. 652–659. DOI: 10.5220/0009102506520659.

[37] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. "CenterNet: Keypoint Triplets for Object Detection". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6568–6577. DOI: 10.1109/ICCV.2019.00667.

[38] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Košecká. "3D Bounding Box Estimation Using Deep Learning and Geometry". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5632–5640. DOI: 10.1109/CVPR.2017.597.

[39] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. "Deep Fitting Degree Scoring Network for Monocular 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1057–1066. DOI: 10.1109/CVPR.2019.00115.

[40] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. "GS3D: An Efficient 3D Object Detection Framework for Autonomous Driving". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1019–1028. DOI: 10.1109/CVPR.2019.00111.

[41] Jason Ku, Alex D. Pon, and Steven L. Waslander. "Monocular 3D Object Detection Leveraging Accurate Proposals and Shape Reconstruction". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 11859–11868. DOI: 10.1109/CVPR.2019.01214.

[42] Bin Xu and Zhenzhong Chen. "Multi-level Fusion Based 3D Object Detection from Monocular Images". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 2345–2353. DOI: 10.1109/CVPR.2018.00249.

[43] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. "Pseudo-LiDAR From Visual Depth Estimation: Bridging the Gap in 3D Object Detection for Autonomous Driving". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 8437–8445. DOI: 10.1109/CVPR.2019.00864.

[44] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L. Waslander. "Joint 3D Proposal Generation and Object Detection from View Aggregation". In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 1–8. DOI: 10.1109/IROS.2018.8594049.

[45] Xinshuo Weng and Kris Kitani. "Monocular 3D Object Detection with Pseudo-LiDAR Point Cloud". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 2019, pp. 857–866. DOI: 10.1109/ICCVW.2019.00114.

[46] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. "Pseudo-LiDAR++: Accurate Depth for 3D Object Detection in Autonomous Driving". In: *Proceedings of International Conference on Learning Representations (ICLR)*. 2020. URL: https://openreview.net/forum?id=BJedHRVtPB.

[47] Jia-Ren Chang and Yong-Sheng Chen. "Pyramid Stereo Matching Network". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 5410–5418. DOI: 10.1109/CVPR.2018.00567.

[48] Xinzhu Ma, Zhihui Wang, Haojie Li, Pengbo Zhang, Wanli Ouyang, and Xin Fan. "Accurate Monocular 3D Object Detection via Color-Embedded 3D Reconstruction for Autonomous Driving". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6850–6859. DOI: 10.1109/ICCV.2019.00695.

[49] Xinlong Wang, Wei Yin, Tao Kong, Yuning Jiang, Lei Li, and Chunhua Shen. "Task-Aware Monocular Depth Estimation for 3D Object Detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* 34.07 (2020), pp. 12257–12264. DOI: 10.1609/aaai.v34i07.6908.

[50] Saifullahi Aminu Bello, Shangshu Yu, Cheng Wang, Jibril Muhmmad Adam, and Jonathan Li. "Review: Deep Learning on 3D Point Clouds". In: *Remote Sensing (RS)* 12.11 (2020). DOI: 10.3390/rs12111729.

[51] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. "PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 77–85. DOI: 10.1109/CVPR.2017.16.

[52] Charles R. 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)*. Vol. 30. 2017, pp. 5105–5114. URL: https://proceedings.neurips.cc/paper/2017/hash/d8bf84be3800d12f74d8b05e9b89836f-Abstract.html.

[53] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. "Multi-view 3D Object Detection Network for Autonomous Driving". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6526–6534. DOI: 10.1109/CVPR.2017.691.

[54] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle Detection from 3D LiDAR Using Fully Convolutional Network". In: *Proceedings of Robotics: Science and Systems (RSS)*. 2016. DOI: 10.15607/RSS.2016.XII.042.

[55] Gregory P. Meyer, Ankit Laddha, Eric Kee, Carlos Vallespi-Gonzalez, and Carl K. Wellington. "LaserNet: An Efficient Probabilistic 3D Object Detector for Autonomous Driving". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 12669–12678. DOI: 10.1109/CVPR.2019.01296.

[56] Leonardo Gigli, B Ravi Kiran, Thomas Paul, Andres Serna, Nagarjuna Vemuri, Beatriz Marcotegui, and Santiago Velasco-Forero. "Road segmentation on low resolution LiDAR point clouds for autonomous vehicles". 2020. arXiv: 2005.13102.

[57] Martin Simon, Stefan Milz, Karl Amende, and Horst-Michael Gross. "Complex-YOLO: Real-time 3D Object Detection on Point Clouds". 2018. arXiv: 1803.06199.

[58] Jorge Beltrán, Carlos Guindel, Francisco Miguel Moreno, Daniel Cruzado, Fernando García, and Arturo De La Escalera. "BirdNet: A 3D Object Detection Framework from LiDAR Information". In: *Proceedings of IEEE InternationalConference Intelligent Transportation Systems (ITSC)*. 2018, pp. 3517–3523. DOI: 10.1109/ITSC.2018.8569311.

[59] Waleed Ali, Sherif Abdelkarim, Mahmoud Zidan, Mohamed Zahran, and Ahmad El Sallab. "YOLO3D: End-to-End Real-Time 3D Oriented Object Bounding Box Detection from LiDAR Point Cloud". In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. Lecture Notes in Computer Science. 2019, pp. 716–728. DOI: 10.1007/978-3-030-11015-4_54.

[60] Ze Wang, Sihao Ding, Ying Li, Minming Zhao, Sohini Roychowdhury, Andreas Wallin, Guillermo Sapiro, and Qiang Qiu. "Range Adaptation for 3D Object Detection in LiDAR". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 2019, pp. 2320–2328. DOI: 10.1109/ICCVW.2019.00285.

[61] Martin Engelcke, Dushyant Rao, Dominic Zeng Wang, Chi Hay Tong, and Ingmar Posner. "Vote3Deep: Fast object detection in 3D point clouds using efficient convolutional neural networks". In: *Proceedings of IEEE International*

Conference on Robotics and Automation (ICRA). 2017, pp. 1355–1361. DOI: 10.1109/ICRA.2017.7989161.

[62] Bo Li. "3D fully convolutional network for vehicle detection in point cloud". In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 1513–1518. DOI: 10.1109/IROS.2017.8205955.

[63] Yin Zhou and Oncel Tuzel. "VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4490–4499. DOI: 10.1109/CVPR.2018.00472.

[64] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. "PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 770–779. DOI: 10.1109/CVPR.2019.00086.

[65] Zetong Yang, Yanan Sun, Shu Liu, and Jiaya Jia. "3DSSD: Point-Based 3D Single Stage Object Detector". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11037–11045. DOI: 10.1109/CVPR42600.2020.01105.

[66] Shaoshuai Shi, Chaoxu Guo, Li Jiang, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. "PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 10526–10535. DOI: 10.1109/CVPR42600.2020.01054.

[67] Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. "From Points to Parts: 3D Object Detection From Point Cloud With Part-Aware and Part-Aggregation Network". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 43.8 (2021), pp. 2647–2664. DOI: 10.1109/TPAMI.2020.2977026.

[68] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. "Fast Point R-CNN". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 9774–9783. DOI: 10.1109/ICCV.2019.00987.

[69] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Jiaya Jia. "STD: Sparse-to-Dense 3D Object Detector for Point Cloud". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 1951–1960. DOI: 10.1109/ICCV.2019.00204.

[70] Anshul Paigwar, Ozgur Erkent, Christian Wolf, and Christian Laugier. "Attentional PointNet for 3D-Object Detection in Point Clouds". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019, pp. 1297–1306. DOI: 10.1109/CVPRW.2019.00169.

[71] Peng Yun, Lei Tai, Yuan Wang, Chengju Liu, and Ming Liu. "Focal Loss in 3D Object Detection". In: *IEEE Robotics and Automation Letters (RA-L)* 4.2 (2019), pp. 1263–1270. DOI: 10.1109/LRA.2019.2894858.

[72] Weijing Shi and Raj Rajkumar. "Point-GNN: Graph Neural Network for 3D Object Detection in a Point Cloud". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 1708–1716. DOI: 10.1109/CVPR42600.2020.00178.

[73] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. "PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 244–253. DOI: 10.1109/CVPR.2018.00033.

[74] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. "Multi-Task Multi-Sensor Fusion for 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7337–7345. DOI: 10.1109/CVPR.2019.00752.

[75] Martin Simon, Karl Amende, Andrea Kraus, Jens Honer, Timo Sämann, Hauke Kaulbersch, Stefan Milz, and Horst Michael Gross. "Complexer-YOLO: Real-Time 3D Object Detection and Tracking on Semantic Point Clouds". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. 2019, pp. 1190–1199. DOI: 10.1109/CVPRW.2019.00158.

[76] Vishwanath A. Sindagi, Yin Zhou, and Oncel Tuzel. "MVX-Net: Multimodal VoxelNet for 3D Object Detection". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2019, pp. 7276–7282. DOI: 10.1109/ICRA.2019.8794195.

[77] Sourabh Vora, Alex H. Lang, Bassam Helou, and Oscar Beijbom. "PointPainting: Sequential Fusion for 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 4603–4611. DOI: 10.1109/CVPR42600.2020.00466.

[78] Ramin Nabati and Hairong Qi. "CenterFusion: Center-based Radar and Camera Fusion for 3D Object Detection". In: *Proceedings of IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2021, pp. 1526–1535. DOI: 10.1109/WACV48630.2021.00157.

[79] Su Pang, Daniel Morris, and Hayder Radha. "CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection". In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 10386–10393. DOI: 10.1109/IROS45743.2020.9341791.

[80] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. "Deep Continuous Fusion for Multi-sensor 3D Object Detection". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 663–678. DOI: 10.1007/978-3-030-01270-0_39.

[81] Tengteng Huang, Zhe Liu, Xiwu Chen, and Xiang Bai. "EPNet: Enhancing Point Features with Image Semantics for 3D Object Detection". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020, pp. 35–52. DOI: 10.1007/978-3-030-58555-6_3.

[82] Jin Hyeok Yoo, Yecheol Kim, Jisong Kim, and Jun Won Choi. "3D-CVF: Generating Joint Camera and LiDAR Features Using Cross-view Spatial Feature Fusion for 3D Object Detection". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020, pp. 720–736. DOI: 10.1007/978-3-030-58583-9_43.

[83] Mario Bijelic, Tobias Gruber, Fahim Mannan, Florian Kraus, Werner Ritter, Klaus Dietmayer, and Felix Heide. "Seeing Through Fog Without Seeing Fog: Deep Multimodal Sensor Fusion in Unseen Adverse Weather". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11679–11689. DOI: 10.1109/CVPR42600.2020.01170.

[84] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. "Object detection in 20 years: A survey". 2019. arXiv: 1905.05055.

[85] Eduardo Arnold, Omar Y. Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. "A Survey on 3D Object Detection Methods for Autonomous Driving Applications". In: *IEEE Transactions on Intelligent Transportation Systems (TITS)* 20.10 (2019), pp. 3782–3795. DOI: 10.1109/TITS.2019.2892405.

[86] Yaodong Cui, Ren Chen, Wenbo Chu, Long Chen, Daxin Tian, Ying Li, and Dongpu Cao. "Deep Learning for Image and Point Cloud Fusion in Autonomous Driving: A Review". In: *IEEE Transactions on Intelligent Transportation Systems (TITS)* 23.2 (2022), pp. 722–739. DOI: 10.1109/TITS.2020.3023541.

[87] Rui Qian, Xin Lai, and Xirong Li. "3D Object Detection for Autonomous Driving: A Survey". In: *Pattern Recognition* 130 (2022), p. 108796. DOI: 10.1016/j.patcog.2022.108796.

[88] Khouloud Dahmane, Najoua Essoukri Ben Amara, Pierre Duthon, Frederic Bernardin, Michele Colomb, and Frederic Chausse. "The Cerema pedestrian database: A specific database in adverse weather conditions to evaluate computer vision pedestrian detectors". In: *Proceedings of the International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*. 2016, pp. 472–477. DOI: 10.1109/SETIT.2016.7939916.

[89] Shizhe Zang, Ming Ding, David Smith, Paul Tyler, Thierry Rakotoarivelo, and Mohamed Ali Kaafar. "The Impact of Adverse Weather Conditions on Autonomous Vehicles: How Rain, Snow, Fog, and Hail Affect the Performance of a Self-Driving Car". In: *IEEE Vehicular Technology (VT) Magazine* 14.2 (2019), pp. 103–111. DOI: 10.1109/MVT.2019.2892497.

[90] You Li, Pierre Duthon, Michèle Colomb, and Javier Ibanez-Guzman. "What happens for a ToF LiDAR in fog?" In: *Transactions on Intelligent Transportation Systems (TITS)* 22.11 (2020), pp. 6670–6681. DOI: 10.1109/TITS.2020.2998077.

[91] Karl Montalban, Christophe Reymann, Dinesh Atchuthan, Paul-Edouard Dupouy, Nicolas Riviere, and Simon Lacroix. "A Quantitative Analysis of

Point Clouds from Automotive Lidars Exposed to Artificial Rain and Fog". In: *Atmosphere* 12.6 (2021). DOI: 10.3390/atmos12060738.

[92] Pierre Duthon, Nadav Edelstein, Efi Zelentzer, and Frédéric Bernardin. "Quadsight® Vision System in Adverse Weather Maximizing the benefits of visible and thermal cameras". In: *Proceedings of the International Conference on Pattern Recognition Systems (ICPRS)*. 2022, pp. 1–6. DOI: 10.1109/ICPRS54038.2022.9854076.

[93] Sebastián Bronte, Luis M Bergasa, and Pablo Fernandez Alcantarilla. "Fog detection system based on computer vision techniques". In: *Proceedings of IEEE InternationalConference Intelligent Transportation Systems (ITSC)*. 2009, pp. 1–6. DOI: 10.1109/ITSC.2009.5309842.

[94] Romain Gallen, Aurélien Cord, Nicolas Hautière, and Didier Aubert. "Towards night fog detection through use of in-vehicle multipurpose cameras". In: *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*. 2011, pp. 399–404. DOI: 10.1109/IVS.2011.5940486.

[95] Mario Pavlić, Heidrun Belzner, Gerhard Rigoll, and Slobodan Ilić. "Image based fog detection in vehicles". In: *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*. 2012, pp. 1132–1137. DOI: 10.1109/IVS.2012.6232256.

[96] Rita Spinneker, Carsten Koch, Su-Birm Park, and Jason Jeongsuk Yoon. "Fast fog detection for camera based Advanced Driver Assistance Systems". In: *Proceedings of IEEE InternationalConference Intelligent Transportation Systems (ITSC)*. 2014, pp. 1369–1374. DOI: 10.1109/ITSC.2014.6957878.

[97] Mario Pavlic, Gerhard Rigoll, and Slobodan Ilic. "Classification of images in fog and fog-free scenes for use in vehicles". In: *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*. 2013, pp. 481–486. DOI: 10.1109/IVS.2013.6629514.

[98] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. "Semantic Foggy Scene Understanding with Synthetic Data". In: *International Journal of Computer Vision (IJCV)* 126.9 (2018), pp. 973–992. DOI: 10.1007/s11263-018-1072-8.

[99] Christos Sakaridis, Dengxin Dai, Simon Hecker, and Luc Van Gool. "Model Adaptation with Synthetic and Real Data for Semantic Dense Foggy Scene Understanding". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 707–724. DOI: 10.1007/978-3-030-01261-8_42.

[100] Martin Hahner, Dengxin Dai, Christos Sakaridis, Jan-Nico Zaech, and Luc Van Gool. "Semantic Understanding of Foggy Scenes with Purely Synthetic Data". In: *Proceedings of IEEE International Conference Intelligent Transportation Systems (ITSC)*. 2019, pp. 3675–3681. DOI: 10.1109/ITSC.2019.8917518.

[101] Andreas Pfeuffer and Klaus Dietmayer. "Robust Semantic Segmentation in Adverse Weather Conditions by means of Sensor Data Fusion". In: *Proceedings of the International Conference on Information Fusion (FUSION)*. 2019, pp. 1–8. DOI: 10.23919/FUSION43075.2019.9011192.

[102] Maxime Tremblay, Shirsendu Sukanta Halder, Raoul de Charette, and Jean-François Lalonde. "Rain Rendering for Evaluating and Improving Robustness to Bad Weather". In: *International Journal of Computer Vision (IJCV)* 129.2 (2021), pp. 341–360. DOI: 10.1007/s11263-020-01366-3.

[103] 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: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 3213–3223. DOI: 10.1109/CVPR.2016.350.

[104] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[105] Michèle Colomb, K. Hirech, P. André, J.J. Boreux, P. Lacôte, and J. Dufour. "An innovative artificial fog production device improved in the European project "FOG"". In: *Atmospheric Research (AR)* 87.3 (2008), pp. 242–251. DOI: 10.1016/j.atmosres.2007.11.021.

[106] Jean-Philippe Tarel, Nicolas Hautière, Aurélien Cord, Dominique Gruyer, and Houssam Halmaoui. "Improved visibility of road scene images under heterogeneous fog". In: *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*. 2010, pp. 478–485. DOI: 10.1109/IVS.2010.5548128.

[107] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. "FlowNet: Learning Optical Flow with Convolutional Networks". In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 2758–2766. DOI: 10.1109/ICCV.2015.316.

[108] Ankush Gupta, Andrea Vedaldi, and Andrew Zisserman. "Synthetic Data for Text Localisation in Natural Images". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2315–2324. DOI: 10.1109/CVPR.2016.254.

[109] David Vázquez, Antonio M. López, Javier Marín, Daniel Ponsa, and David Gerónimo. "Virtual and Real World Adaptation for Pedestrian Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 36.4 (2014), pp. 797–809. DOI: 10.1109/TPAMI.2013.163.

[110] Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. "Playing for Benchmarks". In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 2232–2241. DOI: 10.1109/ICCV.2017.243.

[111] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. "Playing for Data: Ground Truth from Computer Games". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2016, pp. 102–118. DOI: 10.1007/978-3-319-46475-6_7.

[112] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. "The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 3234–3243. DOI: 10.1109/CVPR.2016.352.

[113] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. "Driving in the Matrix: Can virtual worlds replace human-generated annotations for real world tasks?" In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 746–753. DOI: 10.1109/ICRA.2017.7989092.

[114] Yuxiao Zhang, Alexander Carballo, Hanting Yang, and Kazuya Takeda. "Autonomous Driving in Adverse Weather Conditions: A Survey". In: (2021). arXiv: 2112.08936.

[115] Alex Teichman, Jesse Levinson, and Sebastian Thrun. "Towards 3D object recognition via classification of arbitrary object tracks". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2011, pp. 4034–4041. DOI: 10.1109/ICRA.2011.5979636.

[116] Yukyung Choi, Namil Kim, Soonmin Hwang, Kibaek Park, Jae Shin Yoon, Kyounghwan An, and In So Kweon. "KAIST Multi-Spectral Day/Night Data Set for Autonomous and Assisted Driving". In: *IEEE Transactions on Intelligent Transportation Systems (TITS)* 19.3 (2018), pp. 934–948. DOI: 10.1109/TITS.2018.2791533.

[117] Xibin Song, Peng Wang, Dingfu Zhou, Rui Zhu, Chenye Guan, Yuchao Dai, Hao Su, Hongdong Li, and Ruigang Yang. "ApolloCar3D: A Large 3D Car Instance Understanding Benchmark for Autonomous Driving". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 5447–5457. DOI: 10.1109/CVPR.2019.00560.

[118] Jianru Xue, Jianwu Fang, Tao Li, Bohua Zhang, Pu Zhang, Zhen Ye, and Jian Dou. "BLVD: Building A Large-scale 5D Semantics Benchmark for Autonomous Driving". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2019, pp. 6685–6691. DOI: 10.1109/ICRA.2019.8793523.

[119] Yiping Chen, Jingkang Wang, Jonathan Li, Cewu Lu, Zhipeng Luo, Han Xue, and Cheng Wang. "LiDAR-Video Driving Dataset: Learning Driving Policies Effectively". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 5870–5878. DOI: 10.1109/CVPR.2018.00615.

[120] Gaurav Pandey, James R McBride, and Ryan M Eustice. "Ford Campus vision and LiDAR data set". In: *The International Journal of Robotics Research (IJRR)* 30.13 (2011), pp. 1543–1552. DOI: 10.1177/0278364911400640.

[121] Abhishek Patil, Srikanth Malla, Haiming Gang, and Yi-Ting Chen. "The H3D Dataset for Full-Surround 3D Multi-Object Detection and Tracking in

Crowded Urban Scenes". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2019, pp. 9552–9557. DOI: 10.1109/ICRA.2019.8793925.

[122] John Houston, Guido Zuidhof, Luca Bergamini, Yawei Ye, Long Chen, Ashesh Jain, Sammy Omari, Vladimir Iglovikov, and Peter Ondruska. "One Thousand and One Hours: Self-driving Motion Prediction Dataset". In: *Proceedings of the Conference on Robot Learning (CoRL)*. 2020, pp. 409–418. URL: https://proceedings.mlr.press/v155/houston21a.html.

[123] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. "nuScenes: A Multimodal Dataset for Autonomous Driving". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11618–11628. DOI: 10.1109/CVPR42600.2020.01164.

[124] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. "1 year, 1000 km: The Oxford RobotCar dataset". In: *The International Journal of Robotics Research (IJRR)* 36.1 (2017), pp. 3–15. DOI: 10.1177/0278364916679498.

[125] Xinyu Huang, Peng Wang, Xinjing Cheng, Dingfu Zhou, Qichuan Geng, and Ruigang Yang. "The ApolloScape Open Dataset for Autonomous Driving and Its Application". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 42.10 (2020), pp. 2702–2719. DOI: 10.1109/TPAMI.2019.2926463.

[126] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurélien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. "Scalability in Perception for Autonomous Driving: Waymo Open Dataset". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2443–2451. DOI: 10.1109/CVPR42600.2020.00252.

[127] Xavier Roynard, Jean-Emmanuel Deschaud, and François Goulette. "Paris-Lille-3D: A large and high-quality ground-truth urban point cloud dataset for automatic segmentation and classification". In: *The International Journal of Robotics Research (IJRR)* 37.6 (2018), pp. 545–557. DOI: 10.1177/0278364918767506.

[128] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan Dirk Wegner, Konrad Schindler, and Marc Pollefeys. "Semantic3D.net: a new large-scale point cloud classification benchmark". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences (ISPRS)* IV-1/W1 (2017), pp. 91–98. DOI: 10.5194/isprs-annals-IV-1-W1-91-2017.

[129] Bruno Vallet, Mathieu Brédif, Andres Serna, Beatriz Marcotegui, and Nicolas Paparoditis. "TerraMobilita/iQmulus urban point cloud analysis benchmark". In: *Computers & Graphics* 49 (2015), pp. 126–133. DOI: 10.1016/j.cag.2015.03.004.

[130] Shenlong Wang, Min Bai, Gellert Mattyus, Hang Chu, Wenjie Luo, Bin Yang, Justin Liang, Joel Cheverie, Sanja Fidler, and Raquel Urtasun. "TorontoCity: Seeing the World with a Million Eyes". In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 3028–3036. DOI: 10.1109/ICCV.2017.327.

[131] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2633–2642. DOI: 10.1109/CVPR42600.2020.00271.

[132] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. "Detection of traffic signs in real-world images: The German traffic sign detection benchmark". In: *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*. 2013, pp. 1–8. DOI: 10.1109/IJCNN.2013.6706807.

[133] Yohann Cabon, Naila Murray, and Martin Humenberger. "Virtual KITTI 2". 2020. arXiv: 2001.10773.

[134] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes". In: *Proceedings of IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 5000–5009. DOI: 10.1109/ICCV.2017.534.

[135] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, Yunlong Wang, and Diange Yang. "PandaSet: Advanced Sensor Suite Dataset for Autonomous Driving". In: *Proceedings of IEEE InternationalConference Intelligent Transportation Systems (ITSC)*. 2021, pp. 3095–3101. DOI: 10.1109/ITSC48978.2021.9565009.

[136] Zhi Yan, Li Sun, Tomáš Krajník, and Yassine Ruichek. "EU Long-term Dataset with Multiple Sensors for Autonomous Driving". In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 10697–10704. DOI: 10.1109/IROS45743.2020.9341406.

[137] Adam Ligocki, Ales Jelinek, and Ludek Zalud. "Brno Urban Dataset - The New Data for Self-Driving Agents and Mapping Tasks". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 3284–3290. DOI: 10.1109/ICRA40945.2020.9197277.

[138] Quang-Hieu Pham, Pierre Sevestre, Ramanpreet Singh Pahwa, Huijing Zhan, Chun Ho Pang, Yuda Chen, Armin Mustafa, Vijay Chandrasekhar, and Jie

Lin. "A 3D Dataset: Towards Autonomous Driving in Challenging Environments". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 2267–2273. DOI: 10.1109/ICRA40945.2020.9197385.

[139] Peng Jiang, Philip Osteen, Maggie Wigness, and Srikanth Saripalli. "RELLIS-3D Dataset: Data, Benchmarks and Analysis". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 1110–1116. DOI: 10.1109/ICRA48506.2021.9561251.

[140] Ze Wang, Sihao Ding, Ying Li, Jonas Fenn, Sohini Roychowdhury, Andreas Wallin, Lane Martin, Scott Ryvola, Guillermo Sapiro, and Qiang Qiu. "Cirrus: A Long-range Bi-pattern LiDAR Dataset". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 5744–5750. DOI: 10.1109/ICRA48506.2021.9561267.

[141] Jiageng Mao, Niu Minzhe, ChenHan Jiang, hanxue liang hanxue, Jingheng Chen, Xiaodan Liang, Yamin Li, Chaoqiang Ye, Wei Zhang, Zhenguo Li, Jie Yu, Chunjing XU, and Hang Xu. "One Million Scenes for Autonomous Driving: ONCE Dataset". In: *Advances in Neural Information Processing Systems (NIPS) Track on Datasets and Benchmarks*. Vol. 1. 2021. URL: https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/file/67c6a1e7ce56d3d6fa748ab6d9af3fd7-Paper-round1.pdf.

[142] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. "End-to-End Learning of Driving Models from Large-Scale Video Datasets". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 3530–3538. DOI: 10.1109/CVPR.2017.376.

[143] Eder Santana and George Hotz. "A2D2: Audi Autonomous Driving Dataset". 2020. arXiv: 2004.06320.

[144] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. "A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4040–4048. DOI: 10.1109/CVPR.2016.438.

[145] Aitor Almeida, Unai Bermejo, Aritz Bilbao, Gorka Azkune, Unai Aguilera, Mikel Emaldi, Fadi Dornaika, and Ignacio Arganda-Carreras. "A Comparative Analysis of Human Behavior Prediction Approaches in Intelligent Environments". In: *Sensors* 22.3 (2022). DOI: 10.3390/s22030701.

[146] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Huimin Ma, Sanja Fidler, and Raquel Urtasun. "3D Object Proposals Using Stereo Imagery for Accurate Object Class Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 40.5 (2018), pp. 1259–1272. DOI: 10.1109/TPAMI.2017.2706685.

[147] A Geiger, P Lenz, C Stiller, and R Urtasun. "Vision meets robotics: The KITTI dataset". In: *The International Journal of Robotics Research (IJRR)* 32.11 (2013), pp. 1231–1237. DOI: 10.1177/0278364913491297.

[148] Jyri Maanpää, Josef Taher, Petri Manninen, Leo Pakola, Iaroslav Melekhov, and Juha Hyyppä. "Multimodal End-to-End Learning for Autonomous Steering in Adverse Road and Weather Conditions". In: *Proceedings of the International Conference on Pattern Recognition (ICPR)*. 2021, pp. 699–706. DOI: 10.1109/ICPR48806.2021.9413109.

[149] Yohann Cabon, Naila Murray, and Martin Humenberger. "DAWN: Vehicle Detection in Adverse Weather Nature Dataset". 2020. arXiv: 2008.05402.

[150] Jiongchao Jin, Arezou Fatemi, Wallace Michel Pinto Lira, Fenggen Yu, Biao Leng, Rui Ma, Ali Mahdavi-Amiri, and Hao Zhang. "RaidaR: A Rich Annotated Image Dataset of Rainy Street Scenes". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*. 2021, pp. 2951–2961. DOI: 10.1109/ICCVW54120.2021.00330.

[151] Matthew Pitropov, Danson Evan Garcia, Jason Rebello, Michael Smart, Carlos Wang, Krzysztof Czarnecki, and Steven Waslander. "Canadian Adverse Driving Conditions dataset". In: *The International Journal of Robotics Research (IJRR)* 40.4-5 (2021), pp. 681–690. DOI: 10.1177/0278364920979368.

[152] Yayun Lei, Takanori Emaru, Ankit A. Ravankar, Yukinori Kobayashi, and Su Wang. "Semantic Image Segmentation on Snow Driving Scenarios". In: *Proceedings of IEEE International Conference on Mechatronics and Automation (ICMA)*. 2020, pp. 1094–1100. DOI: 10.1109/ICMA49215.2020.9233538.

[153] Dengxin Dai, Christos Sakaridis, Simon Hecker, and Luc Van Gool. "Curriculum Model Adaptation with Synthetic and Real Data for Semantic Foggy Scene Understanding". In: *International Journal of Computer Vision (IJCV)* 128.5 (2020), pp. 1182–1204. DOI: 10.1007/s11263-019-01182-4.

[154] Shirsendu Halder, Jean-Francois Lalonde, and Raoul De Charette. "Physics-Based Rendering for Improving Robustness to Rain". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 10202–10211. DOI: 10.1109/ICCV.2019.01030.

[155] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. "ACDC: The Adverse Conditions Dataset with Correspondences for Semantic Driving Scene Understanding". In: *Proceedings of IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 10745–10755. DOI: 10.1109/ICCV48922.2021.01059.

[156] Pierre Duthon, Michèle Colomb, and Frédéric Bernardin. "Light Transmission in Fog: The Influence of Wavelength on the Extinction Coefficient". In: *Applied Sciences* 9.14 (2019). DOI: 10.3390/app9142843.

[157] Harald Koschmieder. "Theorie der horizontalen sichtweite". In: *Beitrage zur Physik der freien Atmosphare* (1924), pp. 33–53. URL: https://cir.nii.ac.jp/crid/1573668924092647424.

[158] WMO. *Guide to Instruments and Methods of Observation (WMO-No. 8) | World Meteorological Organization*. 2018. URL: https://community.wmo.int/activity-areas/imop/wmo-no_8.

[159] Ales Prokes. "Atmospheric effects on availability of free space optics systems". In: *Optical Engineering* 48.6 (2009), p. 066001. DOI: 10.1117/1.3155431.

[160] David S. Hall. "High definition LiDAR system". US7969558B2. 2011. URL: https://patents.google.com/patent/US7969558B2/en.

[161] Jinsun Park, Kyungdon Joo, Zhe Hu, Chi-Kuei Liu, and In So Kweon. "Non-local Spatial Propagation Network for Depth Completion". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2020, pp. 120–136. DOI: 10.1007/978-3-030-58601-0_8.

[162] Nguyen Anh Minh Mai, Pierre Duthon, Louahdi Khoudour, Alain Crouzil, and Sergio A. Velastin. "3D Object Detection with SLS-Fusion Network in Foggy Weather Conditions". In: *Sensors* 21.20 (2021). DOI: 10.3390/s21206711.

[163] Nguyen Anh Minh Mai, Pierre Duthon, Louahdi Khoudour, Alain Crouzil, and Sergio A. Velastin. "Sparse LiDAR and Stereo Fusion (SLS-Fusion) for Depth Estimation and 3D Object Detection". In: *Proceedings of the International Conference of Pattern Recognition Systems (ICPRS)*. Vol. 2021. 2021, pp. 150–156. DOI: 10.1049/icp.2021.1442. arXiv: 2103.03977.

[164] 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: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 3308–3317. DOI: 10.1109/CVPR.2019.00343.

[165] Chenhang He, Hui Zeng, Jianqiang Huang, Xian-Sheng Hua, and Lei Zhang. "Structure Aware Single-Stage 3D Object Detection From Point Cloud". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11870–11879. DOI: 10.1109/CVPR42600.2020.01189.

[166] Peiliang Li, Xiaozhi Chen, and Shaojie Shen. "Stereo R-CNN Based 3D Object Detection for Autonomous Driving". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7636–7644. DOI: 10.1109/CVPR.2019.00783.

[167] Zengyi Qin, Jinglu Wang, and Yan Lu. "Triangulation Learning Network: From Monocular to Stereo 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7607–7615. DOI: 10.1109/CVPR.2019.00780.

[168] Yilun Chen, Shu Liu, Xiaoyong Shen, and Jiaya Jia. "DSGN: Deep Stereo Geometry Network for 3D Object Detection". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 12533–12542. DOI: 10.1109/CVPR42600.2020.01255.

[169] Chengyao Li, Jason Ku, and Steven L. Waslander. "Confidence Guided Stereo 3D Object Detection with Split Depth Estimation". In: *Proceedings*

of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). 2020, pp. 5776–5783. DOI: 10.1109/IROS45743.2020.9341188.

[170] Tsun-Hsuan Wang, Hou-Ning Hu, Chieh Hubert Lin, Yi-Hsuan Tsai, Wei-Chen Chiu, and Min Sun. "3D LiDAR and Stereo Fusion using Stereo Matching Network with Conditional Cost Volume Normalization". In: *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5895–5902. DOI: 10.1109/IROS40897.2019.8968170.

[171] Maxim Shevtsov, Alexei Soupikov, and Alexander Kapustin. "Highly Parallel Fast KD-tree Construction for Interactive Ray Tracing of Dynamic Scenes". In: *Computer Graphics Forum (CGF)* 26.3 (2007), pp. 395–404. DOI: 10.1111/j.1467-8659.2007.01062.x.

[172] Bin Yang, Wenjie Luo, and Raquel Urtasun. "PIXOR: Real-time 3D Object Detection from Point Clouds". In: *Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 7652–7660. DOI: 10.1109/CVPR.2018.00798.

[173] Alex D. Pon, Jason Ku, Chengyao Li, and Steven L. Waslander. "Object-Centric Stereo Matching for 3D Object Detection". In: *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 8383–8389. DOI: 10.1109/ICRA40945.2020.9196660.

[174] Nguyen Anh Minh Mai, Pierre Duthon, Alain Crouzil, Louahdi Khoudour, and Sergio A. Velastin. "Détection d'obstacles par vision et LiDAR par temps de brouillard pour les véhicules autonomes". In: *Proceedings of journées francophones des jeunes chercheurs en vision par ordinateur (ORASIS)*. 2021. URL: https://hal.archives-ouvertes.fr/hal-03339637.

[175] Nguyen Anh Minh Mai, Pierre Duthon, Pascal Housam Salmane, Louahdi Khoudour, Alain Crouzil, and Sergio A. Velastin. "Camera and LiDAR analysis for 3D object detection in foggy weather conditions". In: *Proceedings of the International Conference on Pattern Recognition Systems (ICPRS)*. 2022, pp. 1–7. DOI: 10.1109/ICPRS54038.2022.9854073. HAL: hal-03820137.

[176] Kaiming He, Jian Sun, and Xiaoou Tang. "Single image haze removal using dark channel prior". In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 1956–1963. DOI: 10.1109/CVPR.2009.5206515.

[177] Yuan-Kai Wang and Ching-Tang Fan. "Single Image Defogging by Multiscale Depth Fusion". In: *IEEE Transactions on Image Processing (TIP)* 23.11 (2014), pp. 4826–4837. DOI: 10.1109/TIP.2014.2358076.

[178] Robin Heinzler, Florian Piewak, Philipp Schindler, and Wilhelm Stork. "CNN-Based LiDAR Point Cloud De-Noising in Adverse Weather". In: *IEEE Robotics and Automation Letters (RA-L)* 5.2 (2020), pp. 2514–2521. DOI: 10.1109/LRA.2020.2972865.

[179] Richard Meyes, Melanie Lu, Constantin Waubert de Puiseau, and Tobias Meisen. "Ablation Studies in Artificial Neural Networks". 2019. arXiv: 1901.08644.

[180] Isha Hameed, Samuel Sharpe, Daniel Barcklow, Justin Au-Yeung, Sahil Verma, Jocelyn Huang, Brian Barr, and C Bayan Bruss. "BASED-XAI: Breaking Ablation Studies Down for Explainable Artificial Intelligence". 2022. arXiv: 2207.05566.

[181] Laith Alzubaidi, Jinglan Zhang, Amjad J Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, José Santamarıa, Mohammed A Fadhel, Muthana Al-Amidie, and Laith Farhan. "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions". In: *Journal of Big Data* 8.1 (2021), pp. 1–74. DOI: 10.1186/s40537-021-00444-8.

[182] Neil C Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F Manso. "The computational limits of deep learning". 2020. arXiv: 2007.05558.

[183] *Alpha Prime*. URL: https://velodynelidar.com/products/alpha-prime/.

[184] *AT128 - HESAI*. URL: https://www.hesaitech.com/en/AT128.

[185] *Pandar128 - HESAI*. URL: https://www.hesaitech.com/en/Pandar128.

[186] Jane Maynard. *The HDL-64E LiDAR Sensor Retires*. 2021. URL: https://velodynelidar.com/blog/hdl-64e-lidar-sensor-retires/.

[187] *Pandar64 - HESAI*. URL: https://www.hesaitech.com/en/Pandar64.

[188] *Velodyne's HDL-32E Surround LiDAR Sensor*. URL: https://velodynelidar.com/products/hdl-32e/.

[189] *RS-LiDAR-32-RoboSense LiDAR - Autonomous Driving, Robots, V2X*. URL: https://www.robosense.ai/en/rslidar/RS-LiDAR-32.

[190] *Puck LiDAR Sensor, High-Value Surround LiDAR*. URL: https://velodynelidar.com/products/puck/.

[191] *LS LiDAR product guide*. 2020. URL: https://www.lidarsolutions.com.au/wp-content/uploads/2020/08/LeishenLiDARProductguideV5.2.pdf.

[192] *Valeo Scala LiDAR*. URL: https://www.valeo.com/en/valeo-scala-lidar/.

[193] Margrit Betke and Zheng Wu. "Evaluation Criteria". In: *Data Association for Multi-Object Visual Tracking*. Springer, 2017, pp. 29–35. DOI: 10.1007/978-3-031-01816-9_4.

[194] Mario Bijelic, Tobias Gruber, and Werner Ritter. "A Benchmark for LiDAR Sensors in Fog: Is Detection Breaking Down?" In: *Proceedings of IEEE Intelligent Vehicles Symposium (IV)*. 2018, pp. 760–767. DOI: 10.1109/IVS.2018.8500543.

[195] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. "Point-Voxel CNN for Efficient 3D Deep Learning". In: *Advances in Neural Information Processing Systems (NIPS)*. Vol. 32. 2019. URL: https://proceedings.neurips.cc/paper/2019/hash/5737034557ef5b8c02c0e46513b98f90-Abstract.html.