Deep Sensor Data Fusion for Environmental Perception of Automated Systems

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät
der Eberhard Karls Universität Tübingen
zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)

vorgelegt von

Fabian Duffhauß

aus Kempen

Tübingen
2023

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	19.02.2024
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Gerhard Neumann
2. Berichterstatter:	Prof. Dr. Martin Butz

Abstract

Automated systems require a comprehensive understanding of their surroundings to safely interact with the environment. By effectively fusing sensory information from multiple sensors or from different points in time, the accuracy and reliability of environmental perception tasks can be enhanced significantly. Besides, learning multiple environmental perception tasks simultaneously can further improve the performance of these tasks due to synergy effects while reducing memory requirements. However, developing deep data fusion networks targeting multiple tasks concurrently is very challenging, leaving substantial room for research and further advancements.

The primary objective of this dissertation is to advance the development of novel data fusion techniques for enhancing the environmental perception of automated systems. It contains a comprehensive investigation of deep learning-based data fusion techniques including extensive experimentation with a focus on the domains of automated driving and industrial robotics. Furthermore, this thesis explores multi-task learning approaches for exploiting synergy effects, reducing computational effort, and lowering memory requirements.

Firstly, this dissertation investigates the encoding and fusion of 3D point clouds for LiDAR-based environmental perception of automated vehicles. In particular, we study novel approaches for simultaneous 3D object detection and scene flow estimation in dynamic scenarios. Our research efforts have yielded a novel deep multi-task learning approach that outperforms previous methods in terms of accuracy and runtime, establishing it as the first real-time solution for this task combination.

Secondly, this dissertation involves research about novel fusion strategies specifically designed to handle multi-modal input data within the field of industrial robotics. In the course of this research, we have developed two novel RGB-D data fusion networks for multi-view 6D object pose estimation. Furthermore, we examine the ambiguity issues associated with object symmetries and propose a novel symmetry-aware training procedure to effectively handle these issues. To comprehensively assess the performance of our proposed methods, we conduct rigorous experiments on challenging real-world and self-generated photorealistic synthetic datasets, revealing significant improvements over previous methods. Moreover, we demonstrate the robustness of our approaches towards imprecise camera calibration and variable camera setups.

Finally, this dissertation explores novel fusion methodologies to integrate multiple imperfect visual data streams, taking into account uncertainty and prior knowledge associated with the data. In the context of this exploration, we have devised a novel deep

hierarchical variational autoencoder that can be utilized as a fundamental framework for various fusion tasks. In addition to leveraging prior knowledge acquired during training, our method can generate diverse high-quality image samples, which are conditioned on multiple input images, even in the presence of strong occlusions, noise, or partial visibility. Furthermore, we have conducted extensive experiments demonstrating a substantial superiority of our method in comparison to conventional approaches.

Kurzfassung

Automatisierte Systeme benötigen ein umfangreiches Verständnis ihrer Umgebung, um sicher mit ihr interagieren zu können. Durch die effektive Fusion von sensorischen Informationen von mehreren Sensoren oder von verschiedenen Zeitpunkten kann die Genauigkeit und Zuverlässigkeit von Aufgaben zur Umgebungserfassung erheblich verbessert werden. Außerdem kann das gleichzeitige Erlernen mehrerer Umgebungserfassungsaufgaben die Leistung dieser Aufgaben aufgrund von Synergieeffekten weiter verbessern und gleichzeitig den Speicherbedarf verringern. Die Entwicklung von tiefen Datenfusionsnetzwerken, welche mehrere Aufgaben gleichzeitig erfüllen, ist jedoch eine große Herausforderung, die noch viel Raum für Forschung und weitere Fortschritte lässt.

Das Hauptziel dieser Dissertation ist es, die Entwicklung neuartiger Datenfusionstechniken zur Verbesserung der Umgebungserfassung von automatisierten Systemen voranzutreiben. Sie enthält eine umfassende Untersuchung von Deep-Learning-basierten Datenfusionstechniken einschließlich umfangreicher Experimente mit einem Fokus auf die Bereiche automatisiertes Fahren und Industrierobotik. Darüber hinaus werden in dieser Arbeit Multi-Task-Learning-Ansätze zur Nutzung von Synergieeffekten, zur Reduzierung des Rechenaufwands und zur Verringerung des Speicherbedarfs untersucht.

In dieser Dissertation wird zunächst die Kodierung und Fusion von 3D-Punktwolken für die LiDAR-basierte Umgebungserfassung von automatisierten Fahrzeugen erforscht. Insbesondere werden neuartige Ansätze für die gleichzeitige 3D-Objekterkennung und die Schätzung des Szenenflusses in dynamischen Szenarien untersucht. Daraus resultiert ein neuartiger Deep-Multitask-Learning-Ansatz, der bisherige Methoden in Bezug auf Genauigkeit und Laufzeit übertrifft und damit die erste Echtzeitlösung für diese Aufgabenkombination darstellt.

Der zweite Teil dieser Dissertation thematisiert die Erforschung neuartiger Fusionsstrategien für die Verarbeitung multimodaler Eingabedaten im Bereich der Industrierobotik. Im Zuge dieser Forschung wurden zwei neuartige RGB-D-Datenfusionsnetzwerke für die 6D-Objektposenschätzung auf Basis von mehreren Kameraperspektiven entwickelt. Darüber hinaus wurden Mehrdeutigkeitsprobleme im Zusammenhang mit Objektsymmetrien untersucht und ein neuartiges Symmetrie-berücksichtigendes Trainingsverfahren konzipiert, welches diese Mehrdeutigkeitsprobleme effektiv reduziert. Zur Evaluierung der präsentierten Methoden wurden umfangreiche Experimente mit herausfordernden realen und selbst generierten fotorealistischen synthetischen Datensätzen durchgeführt, welche eine signifikante Verbesserung gegenüber früheren Methoden zeigen. Darüber hinaus wird

die Robustheit der entwickelten Ansätze gegenüber ungenauer Kamerakalibrierung und variablen Anordnungen der Kameras demonstriert.

Im dritten Teil dieser Dissertation werden neuartige Methoden zur Fusion mehrerer fehlerbehafteter visueller Datenströme unter Berücksichtigung von Unsicherheiten und Vorwissen über die Daten erforscht. Im Rahmen dieser Untersuchung wurde ein neuartiger tiefer hierarchischer Variational Autoencoder konzipiert, der als Grundlage für verschiedene Fusionsaufgaben genutzt werden kann. Die entwickelte Methode nutzt das beim Training erworbene Vorwissen und kann verschiedene qualitativ hochwertige Bilder generieren, welche auf mehreren Eingabebildern beruhen, selbst wenn diese starke Verdeckungen aufweisen, verrauscht sind oder nur teilweise sichtbar sind. Darüber hinaus wurden umfangreiche Experimente durchgeführt, die eine deutliche Überlegenheit der entwickelten Methode im Vergleich zu herkömmlichen Ansätzen belegen.

Acknowledgments

This dissertation has emerged from my doctoral study at the University of Tübingen, my research at the Bosch Center for Artificial Intelligence (BCAI) in Tübingen and Renningen, as well as the collaboration with the Chair for Autonomous Learning Robots (ALR) at the Karlsruhe Institute of Technology (KIT). Many wonderful people from all three institutions supported me during this time.

First and foremost, I would like to express my deepest gratitude towards my doctoral advisor, Prof. Dr. Gerhard Neumann, head of the ALR and formerly honorary professor at the University of Tübingen. I thank him for the countless meetings we have had to discuss my research, to develop novel ideas, and for giving me valuable advice.

I would also like to thank Prof. Dr. Martin Butz, head of the Neuro-Cognitive Modeling Group at the University of Tübingen, for his interest in my research and for serving as the second examiner of my thesis.

Furthermore, I would like to thank Dr. Ngo Ahn Vien and Dr. Hanna Ziesche, who are Research Scientists at the BCAI and my supervisors from the company side. I sincerely appreciate their dedication to my research, their guidance, and the fruitful discussions we have had over the last few years.

I am also grateful to Stefan Baur, doctoral student at the environment perception research division of the Mercedes-Benz AG, for the great collaboration in the context of my first publication at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2020.

In addition, I would like to thank all the students I supervised during my doctoral study. This includes the former master students Lennard Bodden, Lukas Krauch, and Sebastian Koch from the University of Tübingen, Tobias Demmler from the University of Freiburg, and the former bachelor student Jens Beißwenger from the KIT. I am very thankful for the productive conversations and the synergistic collaboration.

Moreover, I would like to thank all the doctoral students and colleagues from my research group at the BCAI and from the ALR for many enriching discussions and the valuable exchange of ideas.

Contents

Chapter 1

Introduction

Automated systems, such as robots, drones, and autonomous vehicles, are rapidly advancing and transforming various sectors, including manufacturing [Rib+21; AG22], transportation [Yur+20; Gup+21], agriculture [OMS21], and healthcare [AMS20; Kyr+21]. Recent technological progress enabled these systems to perform a variety of complex tasks, thus enhancing efficiency, safety, productivity, and reliability [AG22; Mor+20; AMS20; OMS21]. For example, in manufacturing, automated assembly lines and robotic arms are used to streamline production processes, increasing efficiency and precision while reducing costs. Another example is the road traffic where advanced driving assistance systems such as automatic emergency brake systems, lane-keeping assistants, and blind spot detection systems protect from human mistakes and increase safety [Zie+17].

However, further progress towards more advanced automated systems is limited by the ability to accurately perceive the environment in complex or dynamic scenarios. For example, to enable fully autonomous driving on public roads, automated vehicles are required to detect road users, lane markings, traffic signs, and potential obstacles robustly in real-time, even under adverse weather and lighting conditions, such as rain, snow, fog, darkness, and glare. Also in industry, the environmental perception becomes more challenging as the demand for flexible robotic solutions rises. For instance, there are applications in which robots need to reliably detect and grasp a variety of different objects with texture-less or highly reflective surfaces, even in very complex environments with heavy occlusions [Yan+21; AG22].

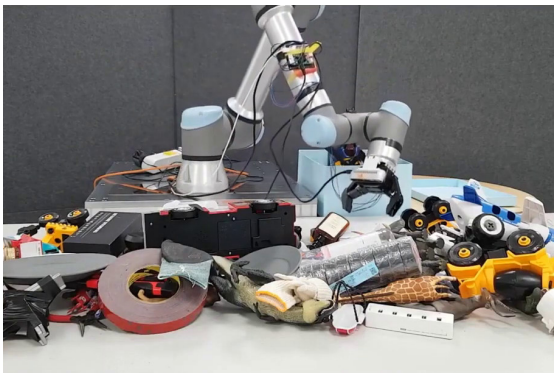
To enable the environmental perception according to all demands, modern automated systems make use of sensor data fusion techniques which can significantly improve the accuracy and reliability of the environmental perception [Sah+20; OB23]. Automated vehicles, for instance, leverage a multitude of different sensors, such as cameras, LiDAR sensors, radar sensors, and ultrasonic sensors, to combine the individual advantages of each sensor [Fay+20a]. Robotic manipulation systems, in contrast, rely more on cameras and depth sensors as well as interactive perception methods [KNK21; Cor+22]. Furthermore, data can be fused over time to enable the understanding of dynamics in a scene or fused over space by collecting data with a single sensor at multiple poses. Independently of what kind of sensory data is fused, there is a trend to use deep learning approaches for perception tasks like object detection, segmentation, and pose estimation

[Zou+23; Zha+21a]. However, despite considerable research efforts, there are still many open research questions to make fusion more data-efficient, more generalizable, and more reliable [Cui+21; OB23]. This motivates the research within this dissertation.

Another trend in deep learning and environmental perception is the utilization of multi-task learning, which aims to learn multiple related tasks jointly in order to improve the generalization performance, to reduce the required computational resources, and to increase the inference speed in comparison to executing multiple independent models [ZY21; Van+21]. For instance, Xu et al. [Xu+18] showed that training semantic segmentation together with depth estimation leads to higher accuracy of both tasks in comparison to individual training processes. Another example is the combination of classification and bounding box regression for object detection [Gir15]. However, designing an accurate and efficient multi-task learning system for environmental perception is still challenging as it requires a careful design and optimization of the system's architecture [Van+21]. The research for this dissertation addresses these challenges in order to explore and exploit the potential benefits of multi-task learning among other techniques.

One main aim of this dissertation is to develop novel deep sensor data fusion techniques for enhancing the environmental perception of automated systems. It includes an extensive investigation of sensor data fusion techniques based on deep learning in two major application domains, namely automated driving and industrial robotics. Figure 1.1 provides two example scenes illustrating typical scenarios in both domains. In addition to

(a) Example scene in automated driving with many different road users moving at various velocities. This open-world domain is characterized by huge diversity of objects, environmental variability due to changes in lighting and weather, and strong occlusions between objects.



(b) Example scene in industrial robotics. Scenes in industrial robotics can contain many different objects of various shapes and textures, whereas objects can severely occlude each other. Image adapted from [Son22].

Figure 1.1: Application domains related to environmental perception of automated systems. Both, the automotive domain (a) and the industrial robotics domain (b) can pose massive challenges to perception systems including heavy object occlusion and diversity.

many domain-specific challenges, both automated driving and robotics can involve very difficult scenes with many objects required to be detected. Noisy sensor data, occlusions, and various lighting conditions among other challenges need to be overcome.

Furthermore, this dissertation presents multiple fusion approaches, which are evaluated on different computer vision tasks, such as 3D object detection, scene flow estimation, 6D object pose estimation, and image generation with noise and occlusion removal. Table 1.1 provides an overview of the three main chapters of this dissertation including the addressed data modalities, fusion types, and application domains. Table 1.2 extends the previous overview with details about the fusion type, application tasks, and challenges which are addressed in the respective chapters. The following three sections introduce the topics of each of these chapters and elaborate on the challenges.

Chapter	RGB data	Range data	Fusion type	Domain
3		✓	Temporal fusion	Automotive
4	✓	✓	Spatial fusion	Robotics
5	✓		Generative fusion	Miscellaneous

Table 1.1: Overview of data modalities, fusion types, and domains addressed in the three main chapters of this dissertation. In chapter 3, range data in form of LiDAR point clouds is used, whereas depth data from an RGB-D camera is considered in chapter 4. Chapter 3 addresses RGB data fusion on multiple datasets related to miscellaneous domains.

1.1 Deep Point Cloud Fusion and Multi-task Learning for Automated Driving Perception

Chapter 3 of this dissertation deals with the environmental understanding of automated vehicles in the dynamic world. Figure 1.1a shows an example scene in this domain, where many different road users, including cars, pedestrians, cyclists, trucks and trams, can move in different directions with a broad range of velocities. Road users in the back can be fully or partly occluded by road users in the front. Further challenges are changing lighting and weather conditions that can significantly alter the visual appearance of all objects.

In order to operate safely on public roads, automated vehicles require precise knowledge about other road users in their surroundings as well as their respective motions. The fulfillment of this requirement can be achieved by combining 3D object detection with scene flow estimation. 3D object detection involves the prediction and classification of oriented 3D bounding boxes for all objects in a given scene. Scene flow can be defined as a three-dimensional vector field that characterizes the motion at every point within a given scene [Ved+99].

Chapter	Fusion type	Application tasks	Challenges
3	Multiple LiDAR point clouds over time	Scene flow estimation and 3D object detection (including 3D bounding box classification and regression)	• Learning scene flow from sparse point cloud data without point-to-point correspondences • Efficient processing and fusion of large LiDAR point clouds • Learning multiple tasks simultaneously in an end-to-end fashion • Consideration of task-specific uncertainties • Achieving real-time capability • Coping with limited real-world datasets containing a small number of different sequences • Performing 3D object detection based on sparse annotations • Mastering imbalance between static and dynamic scene flow vectors
4	RGB and depth data from multiple perspectives	6D pose estimation (including semantic segmentation and 3D keypoint detection)	• Effective fusion of visual and geometric information obtained from RGB and depth data • Processing an arbitrary number of RGB-D images efficiently • Achieving robustness towards inaccurate camera calibration and diverse camera setups • Estimating 6D object poses in very cluttered scenes with many occlusions • Considering ambiguities due to object symmetries • Learning multiple tasks simultaneously in an end-to-end fashion • Coping with limited real-world datasets containing a small number of different scenes • Generation of photorealistic RGB-D data with domain randomization to overcome domain gaps
5	Multiple RGB images with uncertainty	Image generation (including inpainting, noise suppression, and occlusion removal)	• Effective fusion of an arbitrary number of images • Learning comprehensive prior distributions of datasets • Consider uncertainty in the input data • Deriving an evidence lower bound for the conditional log-likelihood of a fusion-enabling hierarchical variational autoencoder • Overcome convergence issues in large hierarchical variational autoencoders • Creating generative fusion tasks for evaluating the proposed approaches

Table 1.2: Overview of fusion types, application tasks, and challenges addressed in the three main chapters of this dissertation.

Both tasks are associated with individual and shared challenges. 3D object detection requires precise measurements of the positions, extends, and orientations of road users while scene flow estimation requires comprehensive knowledge about changes of geometry in 3D space. This becomes increasingly difficult the further away a road user is and the more it is occluded by other objects. We have decided to conduct research on these tasks using LiDAR sensor data because LiDAR sensors possess a precise distance measurement capability. Fusing consecutive LiDAR point clouds enables also the prediction of motions and velocities in the scene.

Previous LiDAR-based 3D object detectors and scene flow estimators, however, do not achieve the desired accuracy and robustness required for safe automated driving [Wan+18a; LQG19; Lan+19; Beh+19b]. Furthermore, estimating scene flow vectors on consecutively acquired LiDAR point clouds is a challenging task in general as the point clouds do not have point-to-point correspondences. Thus, we focus on end-to-end deep learning solutions for this task that aim to learn motion from changes in geometry. Besides, we try to combine 3D object detection and scene flow estimation into a single deep learning model in order to exploit synergy effects while reducing the computational effort in comparison to two single-task networks.

Depending on the employed LiDAR sensor and the scene, the acquired LiDAR point clouds can encompass huge data volumes. For example, the popular automated driving dataset KITTI [GLU12; Gei+13] contains point clouds with an average size of 1.9 MB including around 120,000 measurement points. Processing such a large amount of data efficiently is challenging. Previous methods often reduce the data quantity, e.g. by projecting points to 2D [Che+17c; Ku+18; Bel+18; YLU18], voxelization [MS15; ZT18], or discarding points by sampling [Qi+17a; Qi+17b; Hu+20]. However, it remains a challenge to find a good trade-off between efficiency and data loss. Furthermore, we have to cope with an increasing sparsity of the point density with increasing distance.

Deep learning methods require a substantial amount of labeled data in order to generalize well to diverse scenarios. However, acquiring and labeling data is time-consuming and expensive. Thus, we have to find solutions based on the limited amount of annotated data in public datasets. Furthermore, automotive datasets often contain long driving sequences whereas major parts of the frames show static backgrounds, and only small parts contain dynamic objects. The resulting data sparsity and class imbalances pose major issues in 3D object detection and scene flow estimation.

Chapter 3 presents our research endeavors addressing the previously stated challenges. We have explored ways to efficiently encode and fuse multiple LiDAR point clouds in order to perform 3D object detection and scene flow estimation simultaneously. In this context, we have developed a novel deep multi-task learning approach, called PillarFlowNet, which solves these tasks precisely. We have optimized the point cloud processing and feature encoding so that our network is the first one for this set of tasks that is real-time capable with an inference time of 87.6 ms. Furthermore, we have conducted extensive experiments

to examine synergy effects and demonstrated the outperformance of our approach in terms of accuracy in both tasks compared to the state-of-the-art.

1.2 Multi-View RGB-D Fusion for 6D Pose Estimation

Chapter 4 explores fusion approaches for multi-modal input data with the goal of robustly estimating the 6D poses of objects within a cluttered scene. 6D pose estimation describes the task of jointly predicting the 3D position and the 3D orientation of objects relative to a reference coordinate system. It is an essential tool for environmental perception and widely used in robotics [CMS11; Cor+22; Xia+22], automated driving [Qi+18; Ku+18; Gu+21], augmented reality [MUS15; Su+19], human machine interaction [Pav+17; ML20], and several other fields.

We focus on 6D object pose estimation on very cluttered scenes in the robotics domain as illustrated in figure 1.1b. This is challenging as objects occlude each other so that a specific object to be grasped by a robot might not be visible from a single perspective. For this reason, we consider combining information from multiple perspectives.

There are just a few previous approaches [Zen+17; LBH18; Lab+20] that target the problem of multi-view 6D pose estimation. However, all of these works built their final prediction based on multiple single-view predictions which suffer from high computational cost due to redundancy and hypothesis matching errors. To the best of our knowledge, we are the first to present a deep multi-view fusion approach for this task.

Objects can be manifold in terms of visual appearance and geometric shape. On the one hand, there are objects with identical shapes that can be distinguished only by their texture. This inevitably requires visual information, for example, acquired by RGB cameras. On the other hand, there are texture-less objects whereas the geometry provides the most relevant information. Therefore, we consider fusing both RGB and depth data to combine the individual benefits of each modality. Related literature [Wan+21b; Fen+21; Zhu+22] provides a variety of concepts to fuse different modalities whereas the performance of each concept is highly dependent on the task, the input modalities, and the data. Thus, it is an open research question to conceive a suitable architecture for the fusion of RGB and depth data from multiple perspectives.

Most previous 6D pose estimation methods including [Wan+19; He+20; He+21] do not explicitly consider object symmetries. As symmetric objects have multiple orientations resulting in the same visual appearance and the same geometry, there are multiple correct 6D poses. These ambiguities often result in bad predictions for symmetric objects. Therefore, we conduct research on approaches for overcoming this issue.

Similar to the automotive domain, available data with annotations is limited in robotics. Furthermore, there are no large-scale real-world datasets for multi-view 6D object pose estimation. Thus, we aim to get along with annotated video datasets that provide multiple

perspectives of static scenes over time. This is challenging as only a few different sequences are available showcasing just as few scenes.

Our research has resulted in two novel deep learning approaches for multi-view RGB-D image fusion for 6D object pose estimation. To overcome ambiguity issues due to object symmetries, we propose a novel symmetry-aware training procedure including a novel objective function. For exhaustively evaluating the performance of our proposed methods, we consider challenging real-world datasets and generate several photorealistic synthetic datasets. Our comprehensive experiments indicate vast improvements compared to the state-of-the-art in single-view and multi-view 6D object pose estimation. Moreover, we demonstrate the robustness of our approaches towards dynamic camera arrangements and inaccurate camera calibration.

1.3 Deep Hierarchical Variational Autoencoding for RGB Image Fusion

A major motivation for fusing input data from multiple sources is to gather more information in total compared to single-modal approaches. However, for most applications, the reliability of each data source can vary. For example, if one camera has dirt on its lens, parts of the image might be occluded or blurry. Thus, it would be helpful to integrate an approach measuring the uncertainty of the input data and weight its influence on the outcome dependent on that measurement. However, most sensor fusion approaches in automated driving and robotics do not specifically consider uncertainty in the input data [Che+17c; Ku+18; Wan+19; PMR20; He+20; Zha+21b; He+21]. Therefore, in chapter 5, we investigate and discuss ways to aggregate data from multiple imperfect sources while considering uncertainty.

Furthermore, conventional approaches to fusion for environmental perception tasks tend to prioritize developing methods that merge multiple modalities as optimally as possible. In this context, the endeavor of teaching the model to gather profound prior knowledge about the used dataset often receives less attention. However, having a comprehensive prior knowledge about the relevant objects in a specific application can be very useful in perception, for instance, if parts of an object are not visible due to occlusion, or if the sensor data describing the object is noisy. In these cases, a generative approach can provide supplementing data whenever the relevant input data is missing or inaccurate. This can also be beneficial for subsequent tasks such as robot grasping as the prior knowledge might enable the robot to select grasp poses which are not directly visible.

Over the last few years, many different types of generative models have been conceived such as Variational Auto-Encoders (VAEs), Generative Adversarial Networks (GANs), and diffusion models. GANs are well-known for their high sample quality enabling the generation of realistic images with high resolutions. However, they do not explicitly model the data likelihood and suffer from mode collapse resulting in an incomplete

representation of the data distribution [Zho+19]. VAEs, in contrast, explicitly learn a probabilistic mapping from the data space to a latent space resulting in a well-defined lower-dimensional representation of the data. Furthermore, their reconstruction loss term incentivizes the model to precisely reconstruct the input data, which can be useful for image fusion, denoising, or inpainting. However, working with VAEs is challenging as they tend to generate blurry images, are often limited to low resolutions, and suffer from posterior collapse [Cin+21]. Please note that we do not consider diffusion models in our experiments as their outstanding image generation abilities reached widespread popularity [DN21; Rom+22; Cro+23] after having performed the majority of experiments related to this topic.

In Chapter 5, we present a solution combining the two previously described abilities to consider uncertainty while providing a strong prior knowledge about the used data. Specifically, we have developed a novel deep hierarchical variational autoencoder, called FusionVAE, that can serve as a basis for many fusion tasks. It can generate diverse image samples that are conditioned on multiple noisy, occluded, or only partially visible input images while supplementing it with prior knowledge about the data gained while training.

To the best of our knowledge, we are the first to describe this task combination composed of image fusion with image inpainting, noise suppression, and occlusion removal. For this reason, there are no suitable benchmarks for this task combination prompting us to create some to exhaustively assess the fusion capabilities of our model. Out of this motivation, we have created three novel datasets for image fusion based on popular computer vision datasets.

We have conducted extensive experiments demonstrating a significant superiority of FusionVAE in comparison to traditional approaches on our benchmarks. Additionally, we showcase FusionVAE's ability to generate high-quality image samples even when the input is constrained to a few partially observable images. Moreover, we have performed ablation studies showing the benefits of our design choices regarding both the posterior distribution and commonly used aggregation methods.

1.4 Contributions

This dissertation contributes to the field of environmental perception for automated systems by introducing novel deep learning methods for data fusion among other contributions which are summarized in the following.

Firstly, we address the environmental perception of automated vehicles with a point cloud fusion approach:

- We propose a novel end-to-end trainable network for simultaneous LiDAR object detection and scene flow estimation, called PillarFlowNet.
- We extend the KITTI Object Tracking dataset [GLU12] with static and dynamic scene flow annotations and devised effective data augmentation techniques.
- We demonstrate a significant accuracy increase in multi-task LiDAR scene flow estimation and object detection compared to the state-of-the-art.
- PillarFlowNet is the first multi-task LiDAR scene flow and object detection network to achieve real-time performance.

Secondly, we target the problem of fusing multiple imperfect visual data streams while considering uncertainty and prior knowledge about the used data:

- We present a deep hierarchical variational autoencoder called FusionVAE that is able to perform image-based data fusion while employing prior knowledge of the used dataset.
- We create three challenging image fusion tasks for generative models.
- We show that FusionVAE produces high-quality fused output images and outperforms traditional methods by a large margin.
- We perform ablation studies demonstrating the benefits of our design choices regarding both the posterior distribution and commonly used aggregation methods.

Finally, we address the challenge of fusing multi-modal input data with the goal of reliably predicting the 6D poses of objects within cluttered scenes with strong occlusions:

- We present two novel multi-view fusion frameworks for efficient representation learning of multiple RGB-D frames.
- Based on the frameworks, we propose two novel deep-learning approaches for multi-view 6D object pose estimation.
- We introduce a novel symmetry-aware training procedure for 3D keypoint detection based on a symmetry-aware objective function.
- We present four novel synthetic datasets with photorealistic multi-view RGB-D data and ground truth for both instance semantic segmentation and 6D object pose estimation.
- We demonstrate the superiority of our approaches on challenging real-world and synthetic datasets.

- We prove the robustness of our approaches towards inaccurate camera calibration and dynamic camera setups.
- We reveal significant improvements and synergy effects resulting from the implementation of our symmetry-aware training procedure.

1.5 Thesis Outline

Chapter 2 introduces the fundamental concepts of this dissertation including an introduction to environmental perception, typical sensors, and related data processing approaches.

Chapter 3 deals with multi-task learning for simultaneous 3D object detection and scene flow estimation based on LiDAR data in automated driving. It presents a novel deep-learning approach in this field that achieves very accurate results in real-time.

Chapter 4 covers the field of multi-view RGB-D fusion. It presents two deep multi-task learning approaches for multi-view 6D object pose estimation in very cluttered scenes including a novel training procedure for symmetry-aware keypoint detection.

Chapter 5 addresses the problem of fusing information from multiple sources while supplementing it with prior knowledge about the data gained while training. In particular, it introduces a deep hierarchical variational autoencoder that can generate diverse image samples which are conditioned on multiple noisy, occluded, or partially visible input images.

Chapter 6 concludes this dissertation, thematizes open research questions, and discusses potential future work.

Chapter 2

Background

The following sections provide essential fundamentals which are the basis for this dissertation. Section 2.1 defines environmental perception and describes the most common sensors accompanied by their characteristics. As this dissertation deals mostly with input data in the form of images and point clouds, sections 2.2 and 2.3 present typical methods for processing these two modalities and extracting expressive features. Section 2.4 provides an introduction to data fusion techniques including architecture concepts and fusion operations. In section 2.5, we delve into the fundamentals of generative models focusing on VAEs, GANs, and diffusion models. Finally, section 2.6 is dedicated to presenting the foundations of computer vision and perception tasks which are most relevant to this dissertation, including pertinent related work.

2.1 Environmental Perception and Sensors

Environmental perception describes the process of gathering and interpreting information about the surroundings of an automated system. It involves the acquisition of data about the physical environment, including objects, obstacles, dangerous zones, and further relevant instances. Subsequently, the acquired data is analyzed to obtain a comprehensive understanding of the environment. This procedure can encompass several sub-tasks, such as segmentation, classification, object detection, tracking, and pose estimation [Che+21; Guo+21c]. It is important that these perception algorithms are precise and robust as they are the foundation of subsequent decision-making processes and the execution of actions. [Bi21; Wan+21a].

For gathering sufficient information about the environment, automated systems are typically equipped with a variety of sensors which can be classified into thermal, mechanical (including acoustic), electromagnetic (including optical), nuclear, gravitational, and chemical sensors [Kal13]. However, in order to obtain a high-level understanding of a scene from a distance, automated systems such as autonomous vehicles and industrial robots mostly rely on electromagnetic sensors, such as cameras, depth sensors, laser scanners, radar sensors, and ultrasonic sensors [Fay+20a; Bon+21a; LL19].

Each sensor has individual advantages, limitations, and drawbacks making its suitability for a particular use case dependent on the specific requirements of the application. Figure 2.1 illustrates the sensing capabilities and further properties of the most relevant sensors for environmental perception.

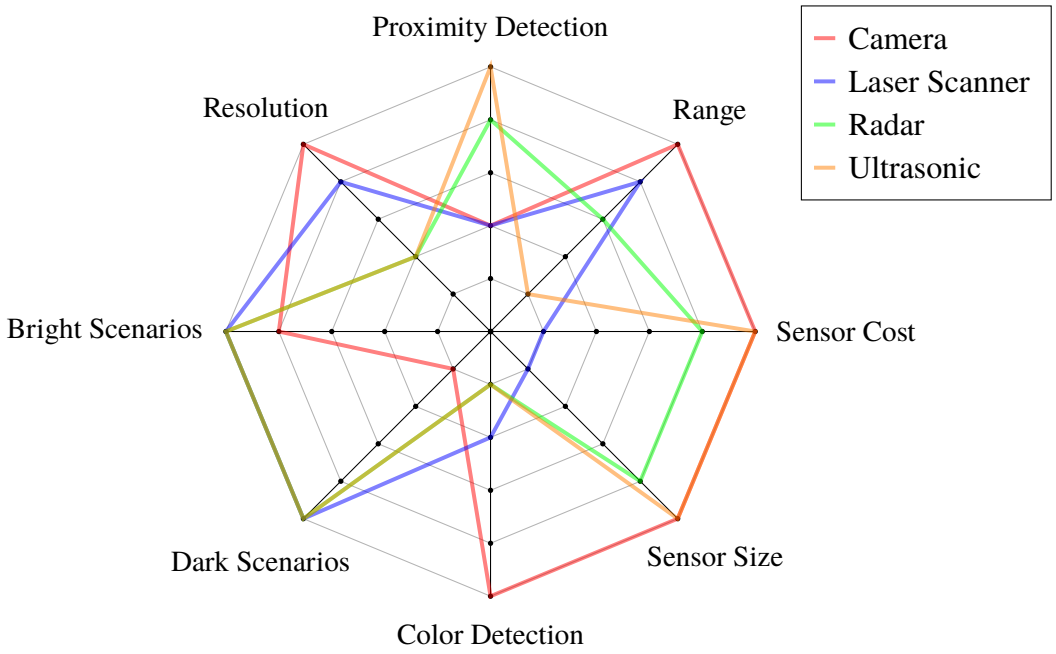


Figure 2.1: Comparison of sensor properties. The farther a line is from the center, the better is the corresponding property.

Cameras

Cameras, and digital RGB (**R**ed **G**reen **B**lue) cameras in particular, are the most common sensors for automated vehicles and robotic applications [LS22]. They record color images by measuring the light entering through the lens with a digital image sensor. As they work without active illumination, cameras are very sensitive to light intensity and require good lighting conditions in order to detect objects. The continual progress in camera hardware and software has enabled high image resolution and quality, enabling a very high range [Che+21]. As cameras are very cheap, small, and lightweight, while having a low power consumption, they offer the versatility to be mounted at multiple locations, enabling the capture of visual data from different perspectives [LS22]. For instance, in the case of a vehicle, cameras can be strategically placed at the front, sides, and rear to enable comprehensive scene monitoring. In robotic applications, cameras can be deployed at various positions surrounding a scene to be manipulated by a robot, facilitating comprehensive visual coverage. Furthermore, cameras can be mounted on robot arms, providing flexibility in capturing scenes from diverse viewpoints.

In addition to RGB cameras, there is a huge variety of other camera types, including infrared cameras, thermal cameras, grayscale cameras, depth cameras, and RGB-D (**R**ed **G**reen **B**lue **D**epth) cameras. In recent years, rapid technological advancements, particularly in the field of depth cameras and RGB-D cameras, have significantly contributed to their increasing popularity and affordability. Depth cameras provide pixel-wise distances to points in the scene. RGB-D cameras enable the joint acquisition of scene appearance and scene geometry in real-time by combining an RGB image with a depth image. The distance measurement can be performed either passively or actively [Ros+19].

Passive depth cameras compute distances without modifying the scene. Predominantly, the computation is based on two monochrome cameras or two RGB cameras, in which case the process is called *stereo reconstruction* or *binocular reconstruction*. The reconstruction process requires corresponding pixels in the two camera views that describe the same location in the scene. Based on each pair of corresponding pixels, the location in the scene can be computed via *triangulation*, i.e., by forming triangles between the camera positions and the observed point in the scene [Ros+19]. However, finding correspondences between two camera views can be challenging.

Active depth cameras modify the scene by emitting light and can be divided into **S**tructured **L**ight (SL) cameras and **T**ime **o**f **F**light (ToF) cameras. SL cameras project an infrared pattern into the scene to simplify the process of finding correspondences. This requires just a single camera, whereas the second camera is replaced by an infrared projector. ToF cameras emit light pulses and can be divided into *Pulsed ToF cameras* and *Modulated ToF cameras*. Pulsed ToF cameras are based on the LiDAR technology and compute distances based on the round-trip times of the light pulses (see section 2.1). Modulated ToF cameras emit time-modulated light pulses and measure the phase shift between emitted and returned pulses [Zan+16; Ros+19].

Laser scanners

Laser scanners are based on the LiDAR (**Li**ght **D**etection **A**nd **R**anging) technology. These devices generate laser beams at near-infrared wavelength (850 – 950 nm) or short-wave infrared wavelength (1550 nm) and emit them within a certain field of view. Objects within this field of view reflect the laser beams and a photodetector measures the intensity of the incoming impulses. The distance d to an object is computed based on the time difference Δt between transmitted and received signals, i.e.

$$d = \frac{c\Delta t}{2n} \tag{2.1}$$

where c is the speed of light in vacuum and n is the refraction index of the propagation medium (approx. 1 for air) [LI20].

By repeating this measurement in multiple directions, a point cloud can be generated where each point represents a specific location in 3D space. Modern laser scanners can

produce very dense point cloud data resulting in a high-resolution geometric representation of the environment. Even though the high resolution of precise distance measurements is a key advantage of laser scanners, processing a large amount of 3D or 4D (when including intensity) data in real-time can be challenging [LS22].

As laser scanners rely on active illumination, they are less affected by external lighting conditions enabling them to work well in dark and bright scenarios. Depending on the laser power, the scanners can have a high operating range. Due to the ability to measure the received intensity, they can detect material properties up to a certain extent. However, laser scanners are expensive and large in comparison to other sensors [Che+21; LS22].

Radar

Radar (**Ra**dio **d**etection **a**nd **r**anging) sensors have a similar functional principle as laser scanners, involving the emission and measurement of electromagnetic waves. However, they differ in the type of electromagnetic waves they emit as radar sensors utilize radio waves within the millimeter or microwave spectrum. For many applications with dynamic objects, such as advanced driver assistance systems and automated driving, radar sensors leverage the Doppler effect to measure the frequency shift of emitted waves, enabling the velocity detection of dynamic objects. The velocity v of a target object relative to the sensor's intrinsic motion can be computed by

$$v = \frac{c\Delta f}{2f_\mathrm{r}} \tag{2.2}$$

where c is the speed of light in vacuum, Δf is the frequency shift due to the Doppler effect, and f_r is the frequency of the emitted radio wave [Yeo+21].

Due to the active emission of radio waves and their propagation properties, radar sensors are not sensitive to lighting conditions and meteorological effects, making them effective in bright and dark scenarios, as well as in adverse weather conditions [Yeo+21]. Furthermore, radar sensors are small and cost-effective enabling versatile mounting similar to cameras. Radar sensors can be deployed for short-range and long-range object recognition, depending on their radiation angle, their emitting frequency, and their delay between emitted pulses [LS22]. However, in comparison to cameras, radar sensors generally provide lower resolution outputs, which limits their effectiveness in accurately detecting objects [Yeo+21; Che+21].

Ultrasonic

Ultrasonic sensors emit mechanical waves with a constant frequency within the inaudible range of humans, i.e. higher than 20 kHz. They measure distances based on the time-of-flight measurement principle as ToF depth cameras, laser scanners, and radar sensors. Since the propagation speed of ultrasonic waves is with around 340 m/s in air much slower than the speed of electromagnetic waves, the sensing rate is much lower. Typical frequencies for automotive applications are in the range of 40 to 70 kHz. Increasing the ultrasonic frequency results in greater absorption, which in turn decreases the maximum range but enables a higher sensing rate [SK16; Bi21; Var+21].

Due to the active emission of waves, ultrasonic sensors perform well in bright and dark environments. They possess the advantages of being affordable, compact, and lightweight, enabling the convenient mounting of multiple sensors in various configurations to achieve comprehensive sensing coverage. However, ultrasonic sensors have a very limited range, cannot provide color information, and have a low resolution. Furthermore, their performance is affected by atmospheric conditions, such as humidity and barometric pressure [Che+21; Var+21].

2.2 Image Processing for Perception

In order to extract information from images that is relevant for environmental perception, a huge variety of methods has been proposed over the last decades. Most traditional approaches without deep learning rely on hand-crafted features, such as edge detection, Histogram of Oriented Gradients (HOG), Scale-Invariant Feature Transform (SIFT), and Local Binary Patterns (LBP) [Jia+19; Ros+19]. However, the performance of these traditional approaches is very limited. This is particularly evident when attempting to detect objects in challenging scenarios, such as highly cluttered scenes and scenes with substantial occlusions [Ros+19].

In recent years, deep learning has experienced rapid progress, largely driven by the availability of huge datasets, powerful computational resources, and algorithmic advancements [Jia+19]. Unlike traditional methods, deep learning approaches extract features in an automated way with low effort and minimal domain expertise [Pou+18].

2.2.1 Convolutional Neural Networks

Especially the occurrence of **C**onvolutional **N**eural **N**etworks (CNN) has led to a rapid improvement of environmental perception methods. As CNNs share weights over entire images, the number of trainable parameters is vastly reduced in comparison to fully connected networks [Jia+19].

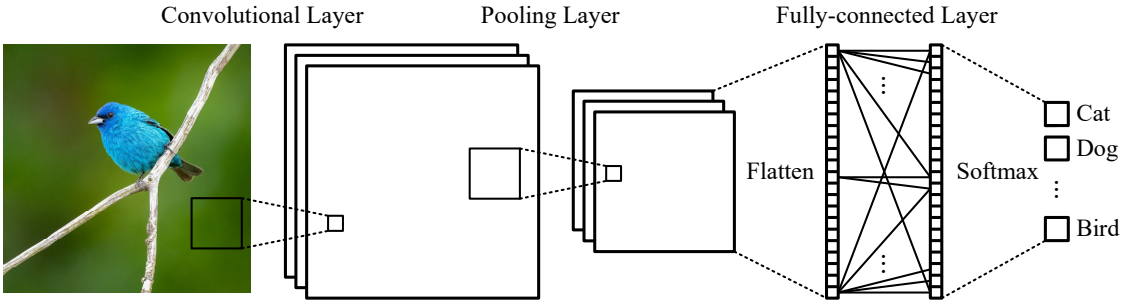


Figure 2.2: Overview of a typical architecture of a CNN for an image classification task.

Figure 2.2 illustrates a typical CNN architecture for an image classification task. The main building blocks are convolutional layers which convolve the input data $\boldsymbol{x}$ at layer l with a set of K learnable kernels $\boldsymbol{W} = \boldsymbol{W}_1, ..., \boldsymbol{W}_K$. These kernels are filter matrices whose size represents the receptive field. After the convolution, a bias value $b = b_1, ..., b_K$ is added to every element in the resulting feature map. Given the feature map of layer l, the k-th feature map of layer $l + 1$ is computed by

$$\boldsymbol{x}_k^{l+1} = \sigma(\boldsymbol{W}_k^l * \boldsymbol{x}^l + b_k^l) \tag{2.3}$$

where σ is an activation function that introduces a non-linearity. Cybenko [Cyb89] demonstrated that a non-linear activation function such as a sigmoid function

$$\sigma = \frac{1}{1 + \exp(-x)} \tag{2.4}$$

enables a neural network to approximate any continuous function [Jia+19].

Another common activation function is the Rectified Linear Unit (ReLU) which is defined as

$$ReLU(x) = \max(0, x) = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}. \tag{2.5}$$

A CNN typically contains a down-sampling mechanism to reduce the size of intermediate network layers. A common choice is a max pooling layer which subdivides the given feature map into non-overlapping rectangles and creates a new feature map based on the maximum values within each rectangle [Jia+19]. As an alternative to pooling layers, convolutional layers with stride can be used. Stride S is a hyperparameter that determines

the step size when moving the convolutional kernels over a feature map. By skipping rows or columns with $S >= 2$, the output feature is reduced in size similar to pooling.

For performing downstream tasks, CNN can be combined with conventional neural network modules such as fully connected layers. For that purpose, a rectangular feature map is flattened to obtain a vector. As illustrated in figure 2.2, a softmax function can be used to compute probabilities for a classification task. Given K classes with input vector $\boldsymbol{z} = [z_1, ..., z_K]^{\mathrm{T}}$ the softmax function is defined as

$$s_i(\boldsymbol{z}) = \frac{\exp(z_i)}{\sum_{j=1}^{K} \exp(z_j)} \quad \text{for } i = 1, ..., K. \tag{2.6}$$

2.2.2 Residual Neural Networks

One type of CNNs, which is rigorously used in this dissertation, is the **Res**idual **Net**work (ResNet), proposed by He et al. [He+16] in 2015. ResNets employ multiple residual blocks as illustrated in figure 2.3. Each building block contains a set of neural network layers resembling the function $F(\boldsymbol{x})$. A parameter-free shortcut connection, also known as a skip connection, is employed which bypasses the neural network layers. After each set of layers, the resulting feature map $\boldsymbol{x}_{\mathrm{res}}$ is computed by the sum of input $\boldsymbol{x}$ and the layer output $F(\boldsymbol{x})$, i.e. [He+16; SG22]

$$\boldsymbol{x}_{\mathrm{res}} = \boldsymbol{x} + F(\boldsymbol{x}). \tag{2.7}$$

The shortcut connections urge the network to learn residual mappings, which capture the differences between the input and the desired output of the corresponding building block. In most cases, residual mappings are easier to optimize than the original mappings and the shortcut connections reduce the issue of vanishing gradients. Thus, the training of very deep neural network architectures is facilitated significantly, resulting in higher accuracy for many environmental perception tasks [He+16; SG22].

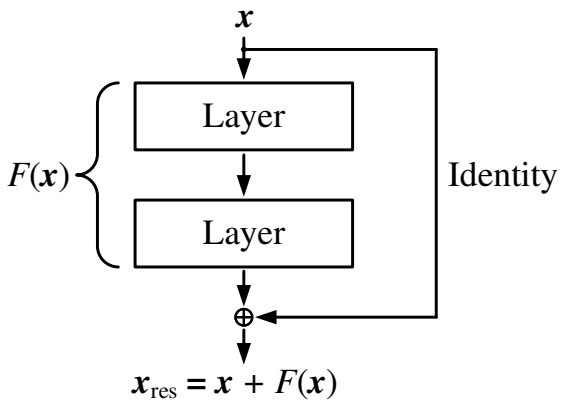


Figure 2.3: Illustration of a residual block. Image adapted from [He+16].

2.2.3 Attention-Based Networks

In recent years, the emergence and advancements of attention mechanisms have considerably contributed to performance improvements in image processing. Attention mechanisms are inspired by the human's perceptual system. Similar to humans who perceive certain parts of the incoming visual information as stronger than others, attention networks have the goal to weight incoming data dependent on their importance [Yan20; Guo+22].

In 2014, Mnih et al. [Mni+14] published pioneering work in visual attention with the introduction of a Recurrent Attention Model (RAM). RAM can recurrently select relevant regions and only processes them at high resolution while processing less important regions with lower resolution. It is trained in an end-to-end manner using a policy gradient method. Jaderberg et al. [Jad+15] proposed a novel Spatial Transformer module which can be employed within neural networks without requiring changes to the loss function. The Spatial Transformer modules contain a localization network that predicts an affine transformation used to select relevant regions in the input [Jad+15; Yan20; Guo+22].

Another important variant of visual attention is the *channel attention*. A milestone in this field is the Squeeze-and-Excitation Network (SENet) by Hu et al. [HSS18]. The authors propose a novel module for CNNs which explicitly models interdependencies between the channels of its convolutional features. This enables a feature recalibration process that fosters the usage of informative features while suppressing less informative ones [HSS18].

Figure 2.4 shows an overview of the previously mentioned module called the Squeeze-and-Excitation (SE) module. Given a feature map $\boldsymbol{X} \in \mathbb{R}^{H' \times W' \times C'}$ the SE module can be applied to any transformation $\mathbf{F}_{tr} : \mathbf{X} \rightarrow \mathbf{U}$ (e.g. a set of convolutional layers) with $\boldsymbol{U} \in \mathbb{R}^{H \times W \times C}$. It employs a squeeze operation on the features $\mathbf{U}$ that aggregates the features along the spatial dimensions H and W by average pooling. The resulting channel descriptor vector with dimensions $1 \times 1 \times C$ is processed by the excitation transformation $\mathbf{F}_{ex}$ which applies two fully connected layers with bottleneck, ReLU, and sigmoid activation function. The outcome is a vector of weights that is used in the scaling transformation $\mathbf{F}_{scale}$ to

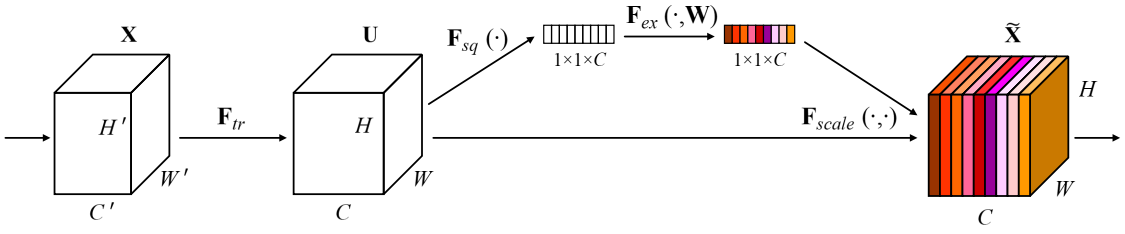


Figure 2.4: Overview of a Squeeze-and-Excitation module [HSS18]. For each channel C of a feature map, it learns and applies a weighting factor corresponding to the importance of the channel. Image from [HSS18].

multiply each of the C feature maps of $\mathbf{U}$ with a scalar according to the relevance of the given channel [HSS18].

In 2017, Vaswani et al. [Vas+17] proposed a *self-attention* mechanism that yielded large success in the domain of natural language processing. Self-attention relates different elements of an input sequence in order to compute a representation of the sequence. It weights these elements according to their relevance and thus adjusts their impact on the output. Wang et al. [Wan+18b] transferred the concept of self-attention to the domain of computer vision and presented the concept of *Non-local Neural Networks*. They employ non-local filtering operations which compute the response at a position as a weighted sum of the features at all positions in the input. The input can be an image, a video, or their features. Thus, non-local operations are a mechanism for capturing long-range dependencies using deep neural networks [Vas+17; Wan+18b; Guo+22].

2.2.4 Transformer-based Networks

Together with the introduction of the self-attention mechanism, Vaswani et al. [Vas+17] proposed a novel network architecture called *Transformer* which opened up a new and successful avenue in deep learning. Transformers constitute an alternative to recurrent neural networks and CNNs while demonstrating an exceptional performance on a broad variety of natural language processing tasks including text summarization [Cai+19], machine translation [Ott+18], and question answering [Sha+19; Zha+20c]. In this field, also the popular Bidirectional Encoder Representations from Transformers (BERT) [Dev+19], and the Generative Pre-trained Transformer (GPT) [RN18; Rad+19; Bro+20] originated from [Kha+22].

Transformers are sequence-to-sequence models with an encoder-decoder architecture as illustrated in figure 2.5. The input is a sequence consisting of a set of tokens with are converted into an embedding. Positional encodings are added to store information about the position of the tokens within the input sequence. The encoder consists of N identical layers performing multi-head attention which relates the feature vectors corresponding to all sequence elements. It further employs residual connections [He+16], layer normalization [BKH16], and a fully-connected feed-forward network [Vas+17].

The Transformer decoder also consists of a stack of N identical layers, but they differ slightly from the encoder. It obtains the Transformer outputs as input shifted right by one position. A masked multi-head attention module relates the different feature vectors whereas a masking ensures that relations can only be created between known outputs. The second multi-head attention module receives the output vectors of the encoder enabling cross-attention between encoder and decoder features [Vas+17].

Motivated by the remarkable achievements of Transformer architectures in natural language processing, the work principle has been successfully transferred to the field of computer vision. Chen et al. [Che+20b] proposed a sequence Transformer called image GPT (iGPT) which does not encode the 2D spatial structure of images. Instead, it resizes

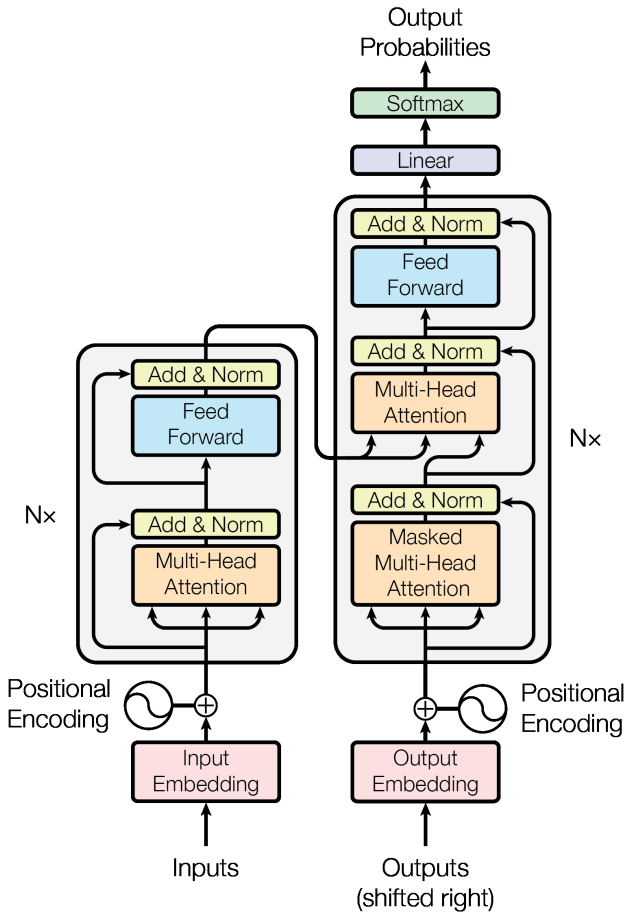


Figure 2.5: Overview of the Transformer architecture consisting of an encoder network stack (left) and a decoder network stack (right). Image taken from [Vas+17].

and flattens the image to a low-resolution 1D sequence that is input to a GPT-2 model [Rad+19] for autoregressive pixel prediction. iGPT achieves competitive results on image classification as CNNs. However, it has a high computational cost enabling the application for low-resolution images only [Che+20b; Liu+23].

In 2020, Dosovitskiy et al. [Dos+21] introduced the Vision Transformer (ViT) which is a pure Transformer [Vas+17] directly applied to a sequence of image patches. Its architecture is illustrated in figure 2.6. First, an input image is reshaped into a sequence of 2D patches. The patches are flattened and processed by a trainable linear projection that maps each flattened patch to the fixed latent vector size D which is used through all Transformer layers. This results in the so-called patch embeddings. [Dos+21; Han+22] Subsequently, a learnable classification token is prepend to the patch embeddings, and position embeddings are added to store positional information. The resulting sequence of embedding vectors is input to a standard Transformer encoder [Vas+17] which consists of

multi-head attention layers, MLPs, layer normalization [BKH16], and residual connections [He+16]. An MLP-based output head is employed for classification [Dos+21].

When pre-trained on large-scale datasets, Vision Transformers achieve outstanding results on multiple image classification tasks including ImageNet [Rus+15] and CIFAR-100 [Kri09]. However, due to the missing inductive bias in comparison to CNNs, they do not generalize well when trained on smaller datasets [Liu+23].

Since its inception, the Transformer was further improved and modified leading to state-of-the-art results in many computer vision tasks including semantic segmentation [Str+21; Gu+22], object detection [Car+20; Li+22d], 6D pose estimation [Zha+22b; Jan+23], and image generation [Cha+22; Zha+22a]. Please refer to section 2.6 for more details about these tasks.

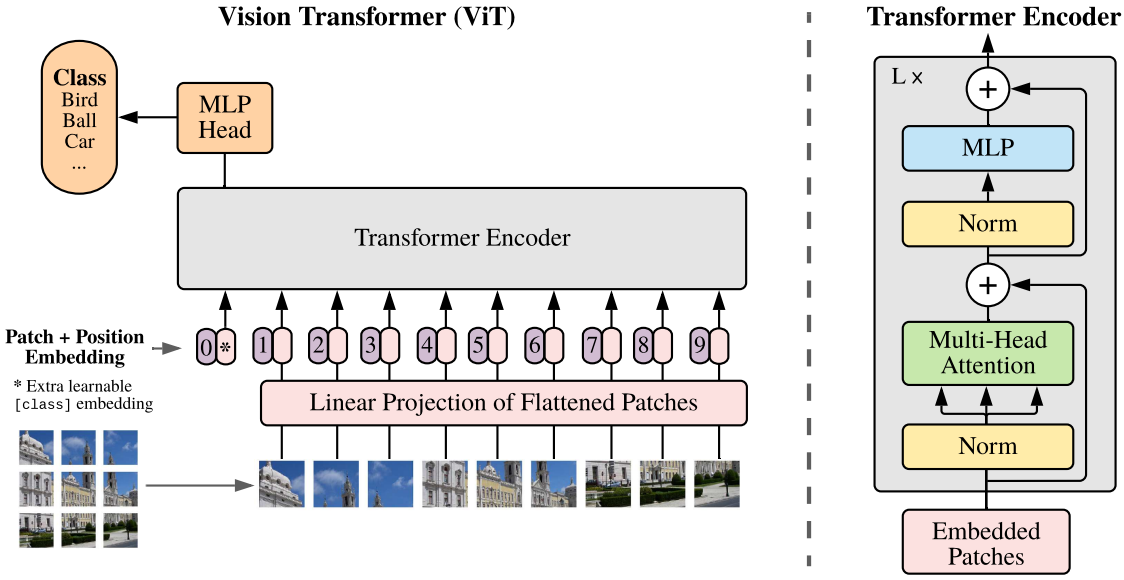


Figure 2.6: Overview of the Vision Transformer architecture [Dos+21]. It divides an input image into patches which are augmented with positional and class information. Based on a Transformer Encoder and an MLP-based output head, it predicts image classification scores. Image from [Dos+21].

2.3 Point Cloud Processing for Perception

Point clouds are sets of three-dimensional points describing the location of objects in 3D space. Thus, they represent the surface of objects and can provide useful information about the geometry, the shape, and the scale of structures. Depending on the type of sensor employed to capture the point cloud, each point can be augmented with supplementary data, such as LiDAR intensity, RGB color values, or surface normals. As typical point clouds in robotics or automated driving are unordered sets of tens to hundreds of thousands of points, it is challenging to infer useful information from point clouds efficiently [Guo+20; Liu+21].

Before the widespread adoption of deep learning, traditional approaches for extracting information from point clouds have mostly relied on hand-crafted features, requiring substantial manual effort and domain knowledge [Jia+19]. Over the last few years, a huge variety of deep learning methods for point cloud processing have been proposed. These methods have the capability to automatically learn feature representations from point cloud data, leading to significant improvements across almost all environmental perception tasks, including point cloud classification, segmentation, object detection, and odometry [Guo+20; Liu+21].

Most deep learning feature extraction methods for point clouds can be classified into projection-based, volumetric-based, and point-based approaches which will be explained in the following [Guo+20].

2.3.1 Projection-based Methods

Projection-based methods project a 3D point cloud onto 2D images, e.g. a panoramic view (cylinder projection) [Shi+15], a front view [LZX16; Che+17c] or a bird's eye view [Che+17c; Ku+18; Bel+18; YLU18]. Thus, the data can be processed as image data, for instance, by using 2D CNNs.

A seminal work in this field is MV3D by Chen et al. [Che+17c] which is illustrated in figure 2.7. It converts a LiDAR point cloud into a bird's eye view and a front view in order to obtain two different compact 2D representations of the point cloud. Using 2D convolutional layers, the bird's eye view, the front view, and an RGB image are processed independently to extract features which are fused subsequently for performing a 3D object detection.

As the projection reduces the amount of data significantly, projection-based methods can scale very well to large point clouds with millions of points. However, the accuracy and robustness of these methods are susceptible to occlusions and viewpoint selection. Furthermore, projection-based methods cannot fully exploit the available geometric information, as the projection inevitably causes a loss of information [Guo+20; Fer+21].

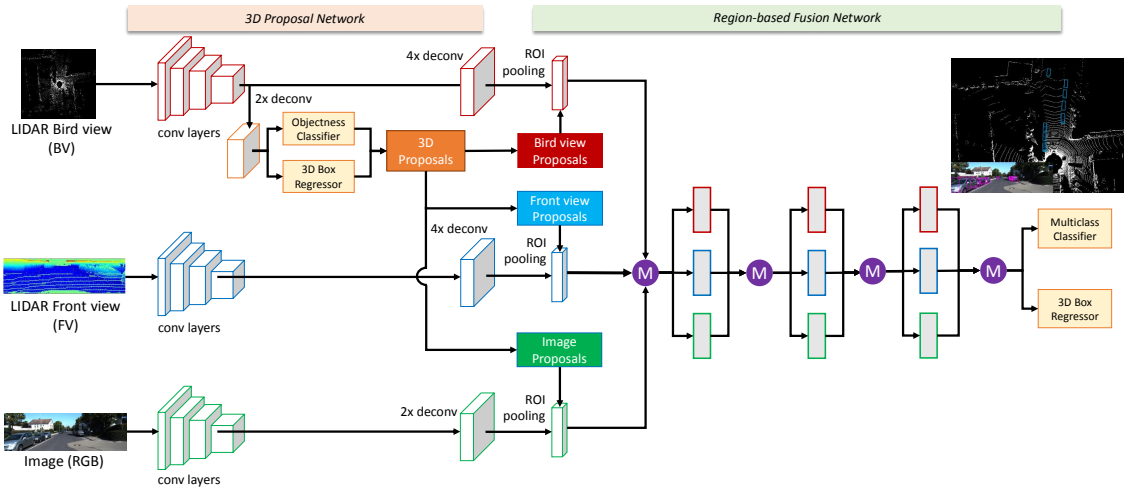


Figure 2.7: Overview of the MV3D [Che+17c] network architecture. MV3D performs 3D object detection based on different 2D projections of a LiDAR point cloud. It converts the given point cloud into a bird's eye view and a front view before extracting features with 2D CNNs. Combining these features with RGB image-based features, MV3D predicts oriented 3D bounding boxes with classification scores. Image from [Che+17c].

2.3.2 Volumetric-based Methods

Volumetric-based methods convert a point cloud into a three-dimensional volumetric representation, such as a voxel grid [MS15; ZT18], frustums [Qi+18; WJ19], or a grid of vertical pillars [Lan+19]. The volumetric representation inherently maintains the neighborhood structure of point clouds, ensuring the preservation of spatial relationships. Voxel-based methods in particular enable point cloud feature extraction with 3D convolutions, learning spatially invariant features across different regions of the point cloud [Guo+20; Fer+21].

Figure 2.8 shows the architecture of VoxNet by Maturana et al. [MS15] as an example for a voxel-based object recognition method. VoxNet constructs a volumetric occupancy grid representation of the input point cloud and feeds it into a 3D CNN exploiting the spatial structure of the data representation.

Volumetric-based methods have the disadvantage that the voxelization causes discretization artifacts and information loss. Besides, it is difficult to select an appropriate grid resolution, as a low resolution increases the information loss while a high resolution requires high memory and computational resources [Guo+20; Fer+21].

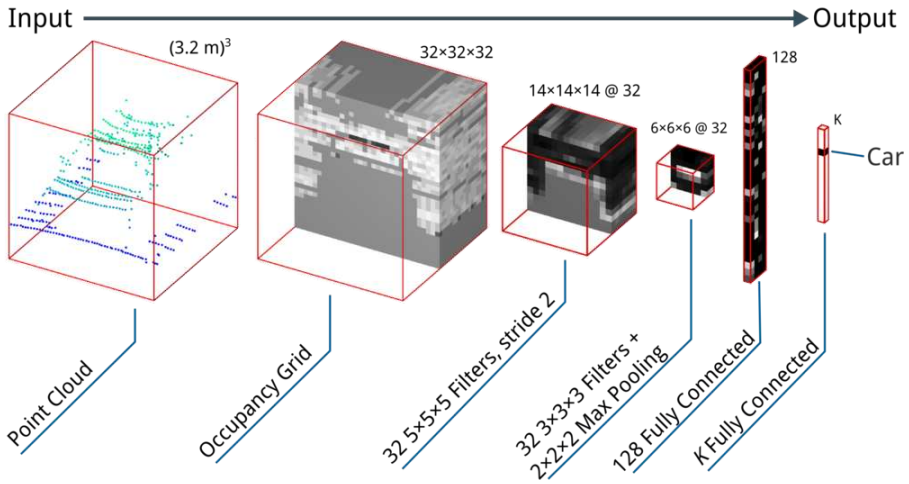


Figure 2.8: Overview of VoxNet [MS15] processing the point cloud of a car. It generates an occupancy grid which is processed by 3D convolutional layers, max pooling, and fully connected layers before predicting classification scores. Image from [MS15].

2.3.3 Point-based Methods

Point-based methods can be applied directly on raw point clouds without projection or voxelization. A seminal work in this area is PointNet by Qi et al. [Qi+17a] which is illustrated in figure 2.9.

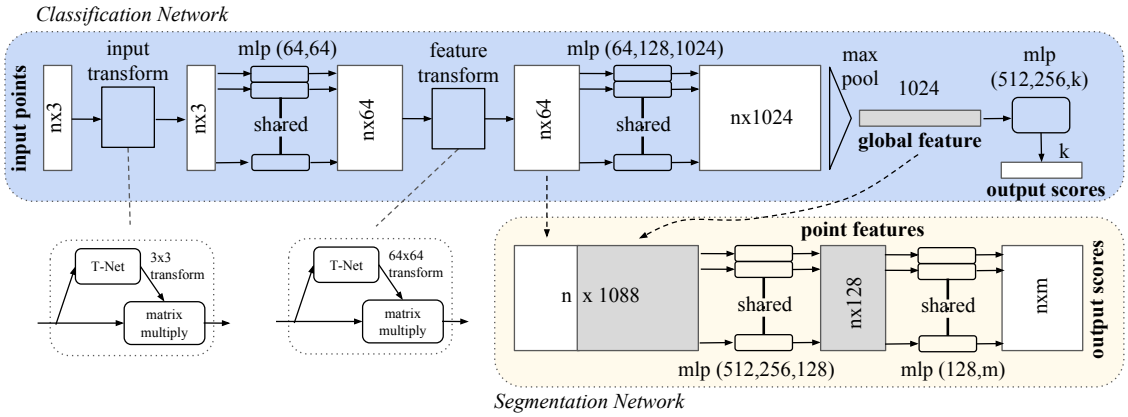


Figure 2.9: Overview of PointNet [Qi+17a]. The classification network operates on a raw point cloud with n points and predicts classification scores for k classes. The segmentation network concatenates the local and global features of the classification networks and predicts point-wise classification scores. Image from [Qi+17a].

PointNet takes the unsorted point cloud with n points and applies an input transformation based on a mini-network, called T-Net, which predicts an affine transformation matrix.

Multiple stages of shared Multi-Layer Perceptrons (MLPs) are utilized to extract point-wise features. PointNet applies a feature transformation based on another T-Net to align features from different input point clouds. Since point clouds are unordered sets of points, the authors propose to use a symmetry function for making the network invariant to input permutation. To achieve that, they employ max pooling for aggregating the point features and obtaining global features [Qi+17a].

As PointNet learns features independently for each point, it does not extract the local structural information between points. This limits the model's ability to recognize fine-grained patterns and to generalize to complex scenes. To address this issue, Qi et al. proposed a hierarchical neural network approach called PointNet++ [Qi+17b] which incrementally aggregates increasingly larger local regions along the hierarchy. Each hierarchical level consists of a sampling layer, a grouping layer, and a PointNet layer. The sampling layer uses the Farthest Point Sampling (FPS) algorithm [Eld+97] to find points in the point cloud that are centroids of local regions. The grouping layer employs a ball query strategy to find adjacent points within each local region. The PointNet layer applies a small PointNet to learn features based on the local regions. In order to better cope with non-uniformly sampled point clouds that vary in point density, Qi et al. introduce Multi-scale grouping (MSG) and Multi-resolution grouping (MRG). These are density-adaptive PointNet layers which enable an intelligent aggregation of multi-scale information according to local point densities.

The utilization of FPS sampling in PointNet++ is the main reason making the method computationally expensive and memory-costly for very large point clouds. Hu et al. [Hu+20] address this issue with a random point sampling strategy and a lightweight neural network architecture called RandLA-Net. As random sampling can discard relevant information, RandLA-Net uses local feature aggregation modules to preserve information captured from adjacent points. Furthermore, it employs attentive pooling to automatically keep the most relevant local features based on a computed attention score.

Following the success of Transformer networks in natural language processing and computer vision, the technology has been also transferred to point cloud processing tasks. Guo et al. [Guo+21a] introduced the Point Cloud Transformer (PCT) which achieves state-of-the-art performance in shape classification and part segmentation. First, they propose a naive approach using the standard Transformer [Vas+17] by treating the entire point cloud as a sentence of points. Second, the authors propose an offset-attention mechanism which calculates the difference between the self-attention features and the input features by element-wise subtraction. Third, they introduce a neighbor embedding inspired by PointNet++ [Qi+17a], which aggregates local neighborhood information of points [Guo+21a; ZWC22].

In recent years, further Transformer-based networks have been proposed, demonstrating state-of-the-art results in point cloud based perception tasks including point cloud classification [EBD21; Lu+22] 3D object detection [Mao+21; Zho+22], 3D semantic

segmentation [Zha+21c; Lai+22], and scene flow estimation [Li+22a]. Please refer to section 2.6 for more details about these tasks.

2.4 Data Fusion

Data fusion describes the combination of data from multiple sources with the aim of improving the quality or amount of information. The data can originate from multiple sensors of the same type, from multiple sensors of different types, or from the same sensor at different points in time. Depending on the application setup, data fusion can increase spatial coverage, temporal coverage, robustness towards sensor failures or algorithmic flaws, noise suppression, and estimation accuracy [Mit12].

The following two subsections present possible network architectures for fusion and different fusion operations.

2.4.1 Fusion Architectures

Fusion techniques can be categorized into early fusion, intermediate fusion, and late fusion techniques according to the stage of the network in which data is merged [Fay+20b; Fen+21]. Figure 2.10 gives an overview of possible fusion network architectures.

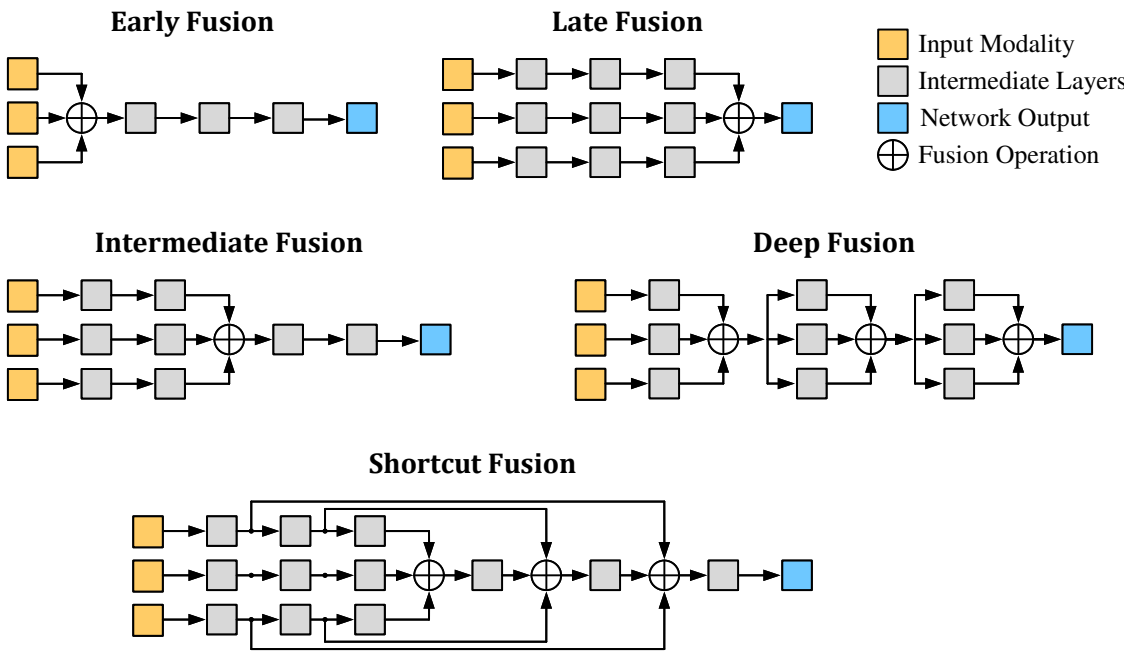


Figure 2.10: Overview of typical architectures for sensor data fusion.

Early fusion. Early fusion, also called data-level fusion, combines raw or pre-processed sensor data. Thus, the entire information from the data can be exploited with low memory usage and low computational expenses. However, early fusion models suffer from limited flexibility since the entire network needs to be retrained when the input modality changes, and varying relevance of different modalities cannot be fully exploited. Another drawback originates from the sensitivity towards spatial-temporal data misalignment between the used sensors e.g. caused by calibration errors, unequal sampling rates, or sensor defects [RT17; Fen+21].

Late fusion. Late fusion, also called decision-level fusion, deploys separate networks for each input modality and combines the network outputs. It enables higher flexibility and modularity compared to early fusion as in case of a new input modality being introduced, only its domain-specific network needs to be trained, without any necessity to modify the other networks. However, late fusion requires high computational and memory resources. Furthermore, it does not exploit the full potential benefits of incorporating intermediate features into the fusion process [RT17; Fen+21].

Intermediate fusion. Intermediate fusion, also called feature-level fusion, is a frequently used compromise between early fusion and late fusion. It utilizes separate feature encoding networks for each input modality to learn independent feature representations. Subsequently, the feature representations are combined into a joint representation for all modalities which can be further processed by a single network. Intermediate fusion offers the greatest flexibility in terms of how and when the fusion takes place. However, it can be very challenging to find an optimal fusion architecture for a specific application. Therefore, there are many variants of intermediate fusion, such as deep fusion and shortcut fusion (see figure 2.10) [RT17; Fen+21].

It should be noted that there is no single fusion technique that is optimal for all applications. The performance of a fusion technique is highly dependent on the type and characteristics of the employed input modalities as well as the task at hand [Fay+20b; Fen+21].

2.4.2 Fusion Operations

The previously presented fusion architectures can be used with a variety of different fusion operations, also called aggregation methods, which will be exhibited in the following. The aim of a fusion operation is to aggregate data, i.e. given a fusion operation G, it combines information from a set of K input tensors $\boldsymbol{f}_1, \boldsymbol{f}_2, ..., \boldsymbol{f}_K$ resulting in a tensor $\boldsymbol{f}_{\text{res}} = G(\boldsymbol{f}_1, \boldsymbol{f}_2 ..., \boldsymbol{f}_K)$.

Concatenation. A very common and simple method to combine multiple tensors is to concatenate them along an existing axis, i.e.

$$\boldsymbol{f}_{\text{res}} = \boldsymbol{f}_1 \oplus \boldsymbol{f}_2 \oplus ... \oplus \boldsymbol{f}_K \tag{2.8}$$

where the operation $\oplus$ denotes concatenation. Please note that the input tensors need to have the same dimensions except for one dimension. The dimension corresponding to the axis along which the concatenation is performed may vary.

Addition. Another very simple fusion operation is the element-wise or pixel-wise addition of the input features, i.e.

$$\boldsymbol{f}_{\text{res}} = \sum_{i=1}^{K} \boldsymbol{f}_i \tag{2.9}$$

Average Pooling. Average pooling corresponds to addition with the exception of an additional scaling factor, i.e.

$$\boldsymbol{f}_{\text{res}} = \frac{1}{K} \sum_{i=1}^{K} \boldsymbol{f}_i \tag{2.10}$$

Maximum Pooling. Maximum pooling, or max pooling for short, selects from all input tensors the element-wise or pixel-wise maxima, i.e.

$$\boldsymbol{f}_{\text{res}} = \max(\boldsymbol{f}_1, \boldsymbol{f}_2, ...\boldsymbol{f}_K). \tag{2.11}$$

Mixture of Experts. The previously presented fusion operations do not explicitly consider the amount of information and uncertainty in the input tensors. However, individual input tensors might contain more relevant information than others, for example, in dark environments an RGB camera provides less useful information than a LiDAR sensor. A mixture of experts tries to overcome that limitation by learning weighting factors w_i for each input tensor so that the resulting tensor is a linear combination of the individually weighted input tensors, i.e.

$$\boldsymbol{f}_{\text{res}} = \sum_{i=1}^{K} w_i \cdot \boldsymbol{f}_i \quad \text{with} \quad \sum_{i=1}^{K} w_i = 1. \tag{2.12}$$

For learning the weighting factors, domain-specific networks called "experts" are employed. This technique was proposed by Jacobs et al. [Jac+91] and further extended in [ERS14; MEB16; Val+17] [Fen+21].

Bayesian Aggregation. Bayesian aggregation [Vol+20] describes a permutation invariant fusion method that considers the uncertainty and the informativeness of the input tensors. It employs two related encoders enc_μ and enc_σ to learn a latent observation $\boldsymbol{\mu}_i = \text{enc}_\mu(\boldsymbol{f}_i)$ with its corresponding variance values $\boldsymbol{\sigma}_i = \text{enc}_\sigma(\boldsymbol{f}_i)$. This represents a factorized Gaussian distribution $\mathcal{N}$ over the latent feature vectors

$$\boldsymbol{f}_i = \mathcal{N}(\boldsymbol{\mu}_i, \text{diag}(\boldsymbol{\sigma}_i)). \tag{2.13}$$

The predicted Gaussian distributions over the latent feature vectors for multiple input tensors are fused iteratively using the Bayes rule [Bec+19]

$$\boldsymbol{q}_i = \boldsymbol{\sigma}_{i-1}^2 \oslash (\boldsymbol{\sigma}_{i-1}^2 + \boldsymbol{\sigma}_i^2) \tag{2.14}$$

$$\text{where } \boldsymbol{\mu}_i = \boldsymbol{\mu}_{i-1} + \boldsymbol{q}_i \odot (\boldsymbol{\mu}_i - \boldsymbol{\mu}_{i-1}) \tag{2.15}$$

$$\text{and } \boldsymbol{\sigma}_i^2 = \boldsymbol{\sigma}_{i-1}^2 \odot (1 - \boldsymbol{q}_i). \tag{2.16}$$

$\oslash$ and $\odot$ denote element-wise division and multiplication respectively.

2.5 Generative Models

Generative models are a class of unsupervised machine learning models which can learn the underlying probability distribution of a given dataset. They allow for generating unseen samples that resemble the original data distribution. Sufficiently trained generative models can be used for many applications, such as image synthesis, text generation, style transfer, data augmentation, audio synthesis, denoising, and inpainting. For most practical applications, however, it is not possible to learn the exact distribution of the given dataset. Therefore, it is common practice to employ deep neural networks to learn an approximation of the target distribution. These particular models are commonly denoted as Deep Generative Models (DGMs) and have the ability to approximate very complicated, high-dimensional probability distributions given enough unlabeled training samples [Tom22; FGH22].

According to Tomczak [Tom22] generative modeling methods can be categorized into four main groups:

- Autoregressive models (e.g., PixelCNN [VKK16])
- Flow-based models (e.g., RealNVP [DSB17])
- Latent variable models (e.g., GANs [Goo+14], VAEs [KW14], and diffusion models [Soh+15])
- Energy-based models (e.g., IGEBM [DM19])

This thesis focuses on the most relevant DGMs for image synthesis, namely GANs, VAEs, and diffusion models. Please refer to [Tom22] for further details about other types of generative models.

2.5.1 Variational Autoencoders

A Variational Auto-Encoder (VAE) is a type of generative latent variable model that was introduced by Kingma et al. [KW14] in 2013. Figure 2.11 illustrates the architecture of a typical VAE. It is built upon an encoder network, a latent space, and a decoder network. The encoder network represents a probabilistic model $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ with trainable parameters ϕ. It compresses the input data $\boldsymbol{x}$ and maps it into the latent space. Thus, the latent variable vector $\boldsymbol{z}$ represents the input data using fewer dimensions. Since the latent space is stochastic, the encoder network has the task of predicting the parameters of the approximate posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. An arbitrary target distribution with an arbitrary number of parameters can be chosen for that task. A common choice is a multivariate Gaussian distribution

$$q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{\mu}_{\boldsymbol{z}|\boldsymbol{x}}, \boldsymbol{\Sigma}_{\boldsymbol{z}|\boldsymbol{x}}) \tag{2.17}$$

with a diagonal covariance matrix $\Sigma_{\boldsymbol{z}|\boldsymbol{x}}$ and mean $\boldsymbol{\mu}_{\boldsymbol{z}|\boldsymbol{x}}$ [Gho+23].

In order to teach the model learning a meaningful approximate posterior distribution, the Kullback-Leibler (KL) divergence [KL51] is utilized. It is a measure for the similarity of two probability distributions $p(x)$ and $q(x)$ and defined as

$$KL(p(x)||q(x)) = \int_{-\infty}^{\infty} p(x) \log\left(\frac{p(x)}{q(x)}\right) \mathrm{d}x. \tag{2.18}$$

The decoder network, also called the generator network, draws samples from the prior distribution $p_\theta(\boldsymbol{z})$ within the latent space and decodes them. Thus, it models the conditional probability distribution $p_\theta(\boldsymbol{x}|\boldsymbol{z})$ where θ are the trainable parameters of the decoder network. Encoder and decoder are trained simultaneously by maximizing the Evidence Lower Bound (ELBO) of the marginal log-likelihood

$$\log p_\theta(\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right] + \mathrm{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z}|\boldsymbol{x})\right) \tag{2.19}$$

$$\geq \underbrace{\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right]}_{L_{\mathrm{rec}}} - \underbrace{\mathrm{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x})||p_\theta(\boldsymbol{z})\right)}_{L_{\mathrm{KL}}} = \mathrm{ELBO}(\boldsymbol{x}), \tag{2.20}$$

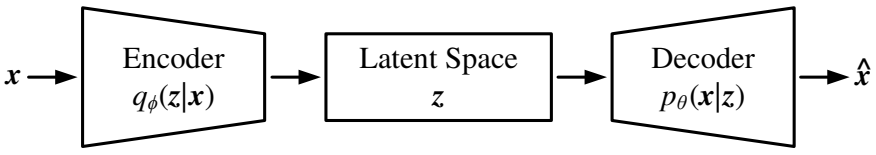


Figure 2.11: Overview of a VAE. Given an input image $\boldsymbol{x}$, a probabilistic encoder q_ϕ learns a distribution in the latent space which is regularised during the training. A probabilistic decoder p_θ can generate new data $\hat{\boldsymbol{x}}$ based on the latent distribution.

where the term L_{rec} represents the likelihood of the reconstructed samples which is maximized. Thus, creating output images deviating from the input image is penalized. The term L_{KL} ensures that the approximate posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ is close to the true prior distribution $p_\theta(\boldsymbol{z})$. It works as a regularization mechanism promoting structure and disentanglement for the latent space [KW14; Cin+21].

Conditional VAE

In standard VAEs, there is no control of the data generation process as unconditioned samples are drawn from the latent distribution. Sohn et al. [SLY15] proposed a remedy for this limitation by presenting the Conditional Variational Auto-Encoder (CVAE). It extends the standard VAE by adding a conditioning vector $\boldsymbol{y}$ (e.g. a class label) as input to both the encoder and decoder. Thus, the encoder aims to learn the approximate posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})$ and the decoder tries to learn the conditional distribution $p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})$. The prior of the latent variable vector $\boldsymbol{z}$ is $p_\theta(\boldsymbol{z}|\boldsymbol{x})$ and the variational lower bound of the conditional log-likelihood can be written as follows

$$\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) = \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})}\left[\log p_\theta(\boldsymbol{y}, \boldsymbol{z}|\boldsymbol{x}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})\right] + \text{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})||p_\theta(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})\right) \tag{2.21}$$

$$\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{y})}\left[\log p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})\right] - \text{KL}\left(q_\phi(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})||p_\theta(\boldsymbol{z}|\boldsymbol{x})\right) = \text{ELBO}(\boldsymbol{x}, \boldsymbol{y}). \tag{2.22}$$

Hierarchical VAE

Sønderby et al. [Søn+16] introduced the first hierarchical VAE, called Ladder Variational Autoencoder (LVAE), by splitting the latent variables $\boldsymbol{z}$ into L layers $\boldsymbol{z}_1, ..., \boldsymbol{z}_L$. This enables a recursive correction of the generative distribution $p_\theta(\boldsymbol{x}, \boldsymbol{z}) = p_\theta(\boldsymbol{x}|\boldsymbol{z}) \cdot p_\theta(\boldsymbol{z})$ and increases the expressiveness of both prior and approximate posterior which become

$$p_\theta(\boldsymbol{z}) = \prod_{l=1}^{L} p_\theta(\boldsymbol{z}_l|\boldsymbol{z}_{<l}) \quad \text{and} \quad q_\phi(\boldsymbol{z}|\boldsymbol{x}) = \prod_{l=1}^{L} q_\phi(\boldsymbol{z}_l|\boldsymbol{z}_{<l}, \boldsymbol{x}), \tag{2.23}$$

where $\boldsymbol{z}_{<l}$ denotes the latent variables in all previous hierarchies. All the conditionals in the prior $p_\theta(\boldsymbol{z}_l|\boldsymbol{z}_{<l})$ and in the approximate posterior $q_\phi(\boldsymbol{z}_l|\boldsymbol{z}_{<l}, \boldsymbol{x})$ are modeled by factorial Gaussian distributions. Under this modeling choice, the variational lower bound of the marginal log-likelihood from equation (2.19) turns into

$$\log p_\theta(\boldsymbol{x}) \geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}, \boldsymbol{z}) - \log q_\phi(\boldsymbol{z}|\boldsymbol{x})\right] \tag{2.24}$$

$$= \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] - \sum_{l=1}^{L} \mathbb{E}_{q_\phi(\boldsymbol{z}_{<l}|\boldsymbol{x})}\left[\text{KL}\left(q_\phi(\boldsymbol{z}_l|\boldsymbol{z}_{<l}, \boldsymbol{x})||p_\theta(\boldsymbol{z}_l|\boldsymbol{z}_{<l})\right)\right] \tag{2.25}$$

$$= \text{ELBO}(\boldsymbol{x}). \tag{2.26}$$

An alternative approach resulting in a hierarchical VAE is the usage of Inverse Autoregressive Flows (IAF) proposed by Kingma et al. [Kin+16a]. They employed Normalizing Flows [RM15a] which iteratively improve the latent variables by

$$\boldsymbol{z}_l = \boldsymbol{\mu}_l + \boldsymbol{\sigma}_l \odot \boldsymbol{z}_{l-1} \tag{2.27}$$

where $\boldsymbol{\mu}_l$ and $\boldsymbol{\sigma}_l$ are predicted mean and standard deviation vectors of the l-th layer and $\odot$ denotes element-wise multiplication.

Further Improvements for VAEs

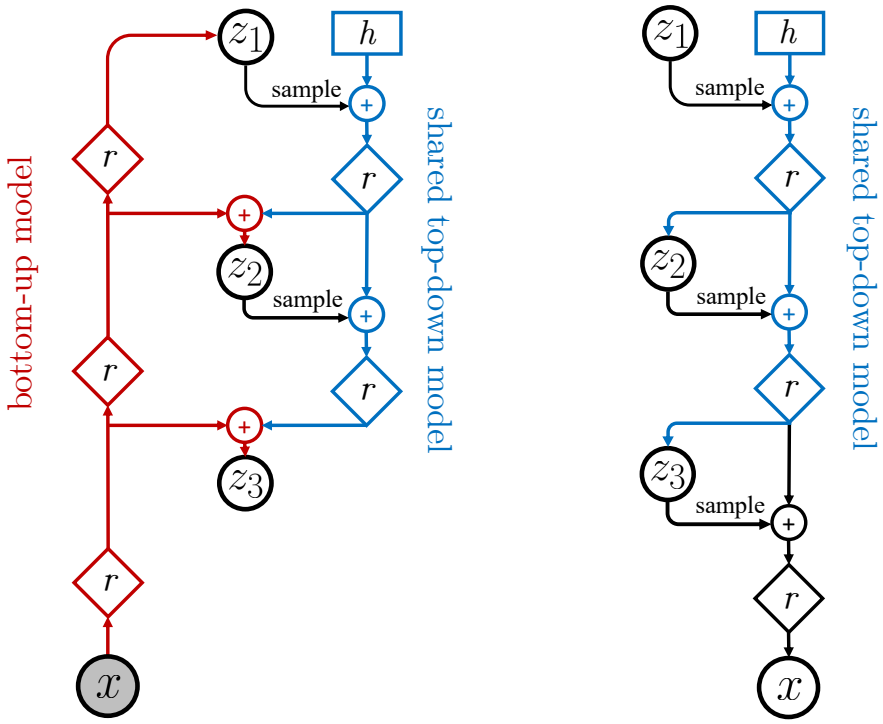
A common issue when maximizing the standard ELBO in equation (2.20) is the variable collapse phenomenon, also called over-pruning [Yeu+17]. This term describes a model's tendency to converge to a sub-optimal solution in which only a small subset of the latent variables is exploited [AT20]. One approach for reducing over-pruning is to vary the balance between the reconstruction loss L_{rec} and the KL loss L_{KL} in equation (2.20) as proposed by Bowman et al. [Bow+16]. They introduced a balancing parameter γ resulting in the training objective

$$L_{\text{total}} = L_{\text{rec}} - \gamma L_{\text{KL}} \tag{2.28}$$

where γ is gradually increased from zero to one. Thus, the training starts as a vanilla autoencoder by learning only the reconstruction. The increment of γ leads to an increasing relevance of the KL loss improving the approximate posterior. As γ reaches one, the training objective L_{total} is equivalent to the ELBO in equation (2.20).

As there is often a mismatch between the aggregated posterior distribution $q_\phi(\boldsymbol{z})$ and the standard Gaussian prior distribution $p(\boldsymbol{z})$, Dai et al. [DW19] have proposed the 2-Stage VAE, which effectively mitigates this issue. The first stage contains a larger VAE for learning a comprehensive data representation $q_\phi(\boldsymbol{z}|\boldsymbol{x})$ within the latent space, though it does not ensure an exact alignment between $q_\phi(\boldsymbol{z})$ and $p(\boldsymbol{z})$. In the second stage, a smaller VAE with independent parameters is employed with the objective of learning to sample from the true distribution $q(\boldsymbol{z})$ without using the prior distribution $p(\boldsymbol{z})$ [DW19; AEL21].

While previous research in the field of VAEs, has predominantly concentrated on statistical challenges, Vahdat et al. [VK20] have shifted their focus on architectural enhancements for hierarchical VAEs. In this context, they have presented the Nouveau VAE (NVAE) which is illustrated in figure 2.12. The bidirectional encoder in figure 2.12a consists of a deterministic bottom-up network which given input $\boldsymbol{x}$ learns L latent variables $\boldsymbol{z}_1, ..., \boldsymbol{z}_L$ representing the approximate posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{x})$. A top-down network with trainable parameter vector h is employed to infer the latent variables group by group. It performs sampling from each latent variable group, combines the samples with deterministic feature maps, and passes the result to the next group. The generative model in figure 2.12b, utilizes the same top-down network with shared parameters to generate data $\boldsymbol{x}$ given the learned latent variables $\boldsymbol{z}$.



(a) Bidirectional Encoder (b) Generative Model

Figure 2.12: Network architecture of the NVAE [VK20] illustrated for three latent variable vectors z_1, z_2, and z_3. It consists of a bidirectional encoder (a), which learns the latent variable vectors from data $\boldsymbol{x}$, and a generative model (b), which decodes samples from the latent probability distributions. $\diamondsuit$ denotes a residual network, $\boldsymbol{h}$ denotes a trainable parameter vector, and $\oplus$ denotes feature aggregation (e.g. concatenation or pixel-wise addition). Images taken from [VK20].

Both encoder and generative model consist of multiple residual cells r including batch normalization [IS15], standard convolutional layers, depth-wise separable convolutional layers [Cho17], swish activation functions [RZL18], and Squeeze-and-Excitation modules [HSS18]. The authors of NVAE propose a novel residual parameterization of the approximate posteriors $q_\phi(z|\boldsymbol{x})$ relative to the prior distribution $p_\theta(z)$. Furthermore, they stabilize the training by spectral norm regularization [YM17].

Please refer to section 5.2.1 for further related work about VAEs for image generation tasks.

2.5.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) were introduced by Goodfellow et al. [Goo+14] in 2014. They consist of two neural networks, a generator network and a discriminator network, which are involved in an adversarial process, i.e. in competition with each other. Figure 2.13 illustrates the architecture of a typical GAN. The generator network G tries to capture the probability distribution of a given dataset and creates new samples $G(\boldsymbol{z})$ from the learned probability distribution. The discriminator network D receives samples from the generator network and samples $\boldsymbol{x}$ from the actual dataset. The goal of the discriminator is to correctly classify the two received samples into the generated one and the real one from the actual dataset [Wan+17].

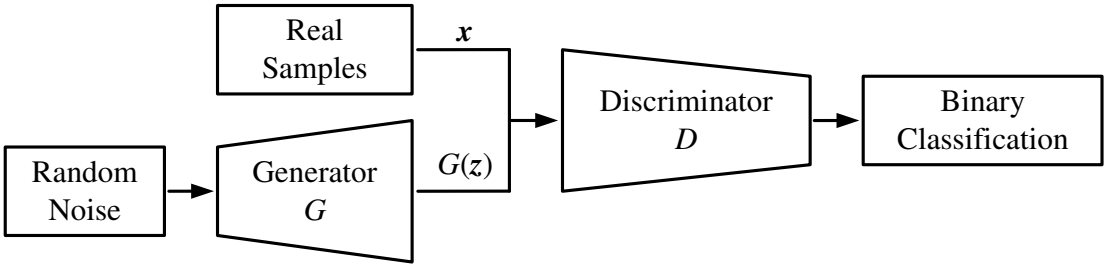


Figure 2.13: Overview of a GAN. The generator network G creates samples while the discriminator network D tries to distinguish between real samples and generated ones.

The optimization of a GAN can be formulated as a minimax problem

$$\min_G \max_D \left(\mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log(D(\boldsymbol{x}))] + \mathbb{E}_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \right) \tag{2.29}$$

where $\boldsymbol{x}$ represents a sample from the real data distribution $p_{\text{data}}(\boldsymbol{x})$, and $\boldsymbol{z}$ represents a sample from a random noise distribution $p_z(\boldsymbol{z})$. Consequently, $D(\boldsymbol{x})$ denotes the predicted probability of $\boldsymbol{x}$ being sampled from the real data distribution rather than being created by the generator [Wan+17; Bon+21b].

Over the last years, many different variants of GANs have been presented that tackle issues and limitations of the original GAN implementation. For example, Mirza et al. proposed a conditional version of GANs (CGAN) [MO14] which enables conditioning both generator and discriminator to auxiliary input data such as a class label. Radford et al. introduced a Deep Convolutional GAN (DCGAN) [RMC16] by employing CNNs in the generator and discriminator. Arjovsky et al. proposed the Wasserstein GAN (WGAN) [ACB17] which reduces the problems of mode collapse by reformulating the minimax problem using the Earth-Mover distance [RTG00]. Mao et al. introduced the Least Squares GAN (LSGAN) [Mao+17] which reduces the vanishing gradient problem by establishing a least squares loss function for the discriminator network. Mejjati et al. [Ala+18] and Zhang et al. [Zha+19] proposed attention-based GANs which further improved the quality

of the generated images. Recently, also Transformer-based GANs [JCW21; Lee+22] have been proposed which achieve competitive results on low-resolution image generation tasks in comparison to CNN-based GANs. However, they do not reach the performance of CNN-based GANs on high-resolution benchmarks.

2.5.3 Diffusion Models

Diffusion models are inspired by diffusion processes in non-equilibrium thermodynamics [Soh+15]. They belong to a class of deep generative models which are based on two processes, a forward diffusion process and a reverse diffusion process. The forward diffusion process impairs the input data incrementally through the iterative addition of noise. In the subsequent reverse process, a neural network tries to reconstruct the original input data progressively by learning to undo the diffusion process [Cro+23].

According to Yang et al. [Yan+22] and Croitoru et al. [Cro+23], diffusion models can be categorized into three main categories: denoising diffusion probabilistic models (DDPM), score-based generative models, and stochastic differential equations. DDPMs are the most relevant category of diffusion models which are explained in the following subsection. Please refer to [Yan+22; Cro+23] for further details about all types of diffusion models.

Denoising Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models (DDPMs) [HJA20; ND21] are based on a forward process and a reverse process as all three forms of diffusion models. Figure 2.14 illustrates the typical architecture of a DDPM. The forward diffusion process gradually perturbs the input data $\boldsymbol{x}_0$ with Gaussian noise. This process can be formulated as a Markov chain, where $\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T$ are the noised versions of the input data after t steps.

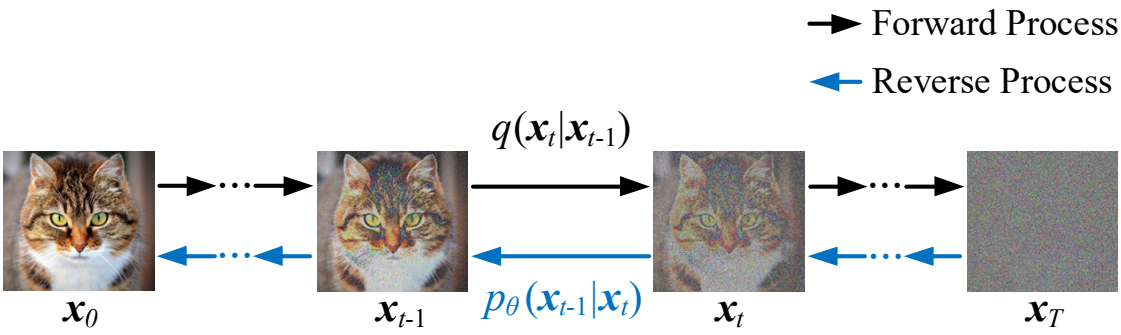


Figure 2.14: Overview of a diffusion model. The forward process gradually adds noise to an input image $\boldsymbol{x}_0$. The reverse process aims to incrementally remove noise.

$q(\boldsymbol{x}_t)$ represents the probability distribution of the data after t steps. The Markov chain is defined as

$$q(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}) = \mathcal{N}\left(\boldsymbol{x}_t|\sqrt{1-\beta_t}\boldsymbol{x}_{t-1}, \beta_t\boldsymbol{I}\right) \tag{2.30}$$

where $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ represents a multivariate Gaussian distribution with mean vector $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ which creates $\boldsymbol{x}$. $\boldsymbol{I}$ is the identity matrix which has the same dimensions as the input data $\boldsymbol{x}_0$. $\beta_t \in [0, 1)$ is a scalar variance parameter that can attain different values for different t [Cro+23].

In the reverse stage, the DDPM can generate new samples from the data distribution $q(\boldsymbol{x}_0)$ by starting with a sample $\boldsymbol{x}_T \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and reversing the process of adding noise. For removing the noise, a neural network $p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t) = \mathcal{N}(\boldsymbol{x}_{t-1}|\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t), \boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t))$ with learnable parameters θ is trained that receives the noisy input $\boldsymbol{x}_t$ and predicts mean $\boldsymbol{\mu}_\theta(\boldsymbol{x}_t, t)$ and variance $\boldsymbol{\Sigma}_\theta(\boldsymbol{x}_t, t)$). The training objective is to minimize an evidence lower-bound of the negative log-likelihood

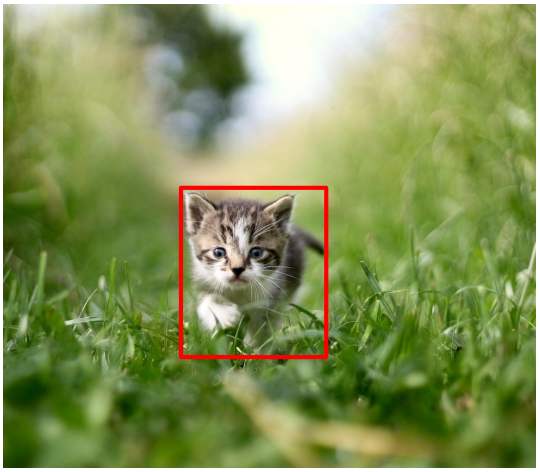
$$\begin{aligned} ELBO = \mathbb{E}_q\Bigg[& \underbrace{KL(q(\boldsymbol{x}_T|\boldsymbol{x}_0)||p(\boldsymbol{x}_T))}_{L_T} + \sum_{t=2}^{T} \underbrace{KL(q(\boldsymbol{x}_{t-1}|\boldsymbol{x}_t, \boldsymbol{x}_0||p_\theta(\boldsymbol{x}_{t-1}|\boldsymbol{x})))}_{L_{t-1}} \\ & \underbrace{-\log p_\theta(\boldsymbol{x}_0|\boldsymbol{x}_1)}_{L_0} \Bigg] \end{aligned} \tag{2.31}$$

where each term is a KL divergence between two Gaussian distributions that can be evaluated in closed form. Please note that the first term L_T of equation (2.31) can be ignored during optimization as it does not depend on the neural network parameters θ. The last term L_0 represents a reconstruction loss similar to the ELBO of the variational autoencoder in equation (2.20).

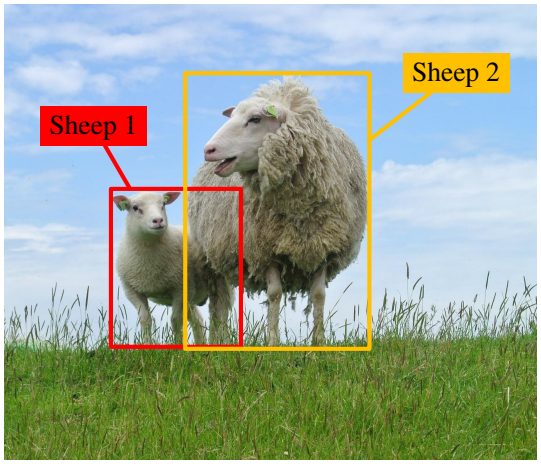
2.6 Computer Vision Tasks

This dissertation deals with a large variety of different computer vision tasks including perception tasks, such as classification, 2D object detection, 3D object detection, semantic segmentation, instance segmentation, 6D pose estimation, optical flow estimation, and scene flow estimation. Furthermore, this work concerns generative tasks, such as image generation and image inpainting. The following subsections thematize each of these tasks, giving a definition, presenting solution approaches, and pointing out challenges. Figures 2.15, 2.17 and 2.21 to 2.24 illustrate these tasks with different example images.

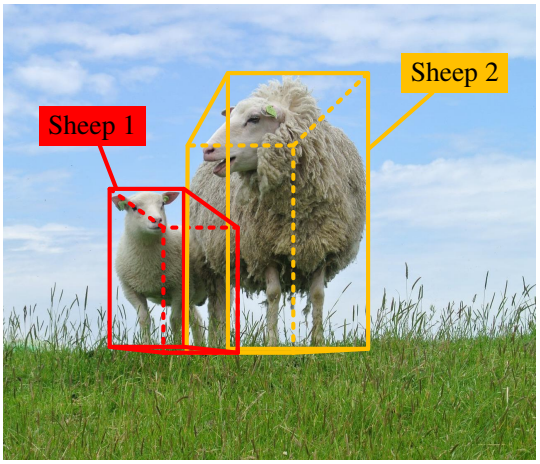
(a) Classification with label “cat”



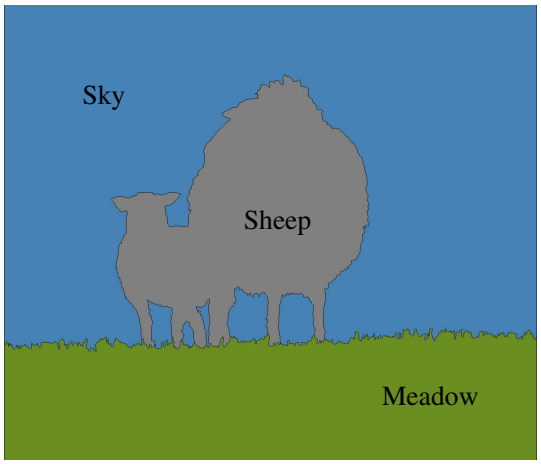
(b) Localization with label “cat”



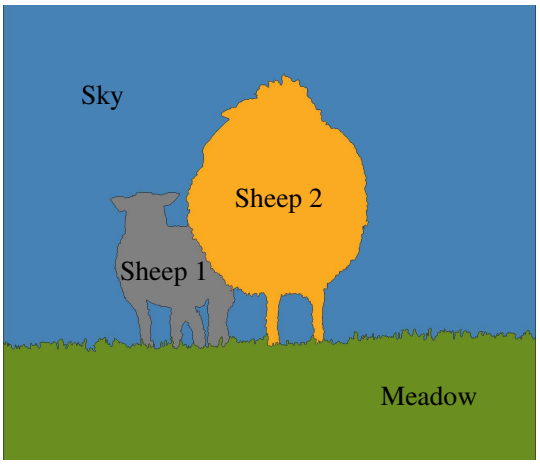
(c) 2D object detection



(d) 3D object detection



(e) Semantic segmentation



(f) Instance/panoptic segmentation

Figure 2.15: Overview of different computer vision perception tasks.

2.6.1 Classification

Classification describes the task of categorizing the given input data into a limited set of classes. In computer vision, the input data can be for example an RGB image, a point cloud, or a video. The output of a classification model is typically a classification score vector

$$s = (s_1, ..., s_K) \in \mathbb{R}^K \tag{2.32}$$

where each element s_i is the predicted probability, that the input data contains an object of class i. The maximum value in $\boldsymbol{s}$ determines the most likely class. Assuming the existence of K classes, an additional class score s_0 can be added, indicating the probability that none of the K classes is in the data. Figure 2.15a shows an example RGB image of a cat whereas the classification categorizes the entire image as "cat" without localizing the cat in the image.

Image classification performance has faced tremendous improvements due to the emergence of CNNs. Starting LeNet-5 [LeC+98], CNNs became incrementally more powerful with the development of AlexNet [KSH12], VGGNet [SZ15], GoogLeNet [Sze+15], and ResNet [He+16] (see sections 2.2.1 and 2.2.2). Concurrently, many computer vision datasets have been published accelerating the advancements in image classification and other perception tasks. These include MNIST [LeC98], CIFAR-10 and CIFAR-100 [Kri09], the PASCAL Visual Object Classes (VOC) Challenge [Eve+10], the Scene Understanding (SUN) database [Xia+10] and the ImageNet dataset [Den+09] with the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [Rus+15]. Recently, image classification has been further improved by attention mechanisms [Jad+15; HSS18] and Transformer architectures [Che+20b; Dos+21] (see sections 2.2.3 and 2.2.4).

Point cloud classification in recent years has been mainly driven by 3D CNNs [MS15; Wu+15], projection-based methods [Shi+15; Su+15a], point-based methods [Qi+17a; Qi+17b], and Transformer-based methods [Zha+21c; EBD21; Lu+22]. A large number of point cloud based classification benchmarks has been proposed including the Sydney Urban Objects Dataset [De +13], the ModelNet shape classification benchmark [Wu+15], the Semantic3D Large-Scale Point Cloud Classification Benchmark [Hac+17], the ScanNet dataset [Dai+17], and the ScanObjectNN dataset [Uy+19].

2.6.2 Localization

Localization is a regression task with the goal of finding instances of a given object class in the data and predicting a 2D bounding box around the object. Figure 2.15b shows a localization example where a cat is localized in an RGB image and framed with a 2D bounding box. The regression output is typically a 4D vector

$$(x_c, y_c, w, h) \in \mathbb{R}^4 \tag{2.33}$$

with center coordinate (x_c, y_c), width w, and height h or the corner coordinates

$$(x_0, y_0, x_1, y_1) \in \mathbb{R}^4 \tag{2.34}$$

of the bounding box. Most related work directly combines the localization task with a classification task, which is known as 2D object detection (see section 2.6.3).

2.6.3 2D Object Detection

2D object detection describes the task of finding desired objects in the given input data, framing them with a 2D bounding box, and classifying each object. The output is a set of bounding boxes with individual class labels or score vectors, using encodings from equations (2.32) to (2.34). Unlike, a pure classification or localization task, object detection enables the recognition of multiple objects of different classes in the same input data. Figure 2.15c shows an example of a 2D object detection in an RGB image, where two sheep are detected using individual bounding boxes with class labels.

Traditional object detection methods [PP00; VJ01; SK04; Fel+09] rely mostly on hand-crafted features which were subsequently processed by bounding box regressors and classifiers. With the occurrence of deep learning, the development of object detectors is split into two-stage and one-stage approaches. *Two-stage object detectors* [Gir+14; Gir15; Ren+17; Lin+17a] have a region proposal network in the first stage which predicts areas in the image that potentially contain objects. The second stage classifies the region proposals, removes duplicate detections, and adjusts the bounding boxes [WSH20].

One-stage object detectors [Ser+14; Red+16; Liu+16; Lin+17b; LD18; Dua+19], in contrast, are end-to-end trainable networks that directly predict bounding box coordinates and classification scores without separate stages. During the first years of the coexistence of both object detector types, the two-stage detectors were considered as more precise but slower than the one-stage detectors. However, Lin et al. [Lin+17b] were the first who overcome the limited precision of one-stage detectors by introducing the focal loss. This is a dynamically scaled variant of the cross-entropy loss that considers class imbalances. Using the focal loss, their proposed RetinaNet outperformed all previous two-stage detectors while matching the high speed of previous one-stage detectors [Lin+17b; WSH20].

After a long period in which most object detectors were based on CNNs [Gir+14; Ser+14; Gir15; Red+16; Liu+16; Ren+17; Lin+17a; Lin+17b; LD18; Dua+19], the introduction of the Transformer [Vas+17] has brought a turnaround. Many recent 2D object detectors [Car+20; Sun+21; Li+22d; Fan+23] have been developed using the Transformer as a building block and outperforming previous CNN-based approaches.

2.6.4 3D Object Detection

3D object detection is the task of regressing and classifying 3D bounding boxes for specific objects in the input data. A 3D bounding box is defined by the vector

$$\boldsymbol{b} = (x_c, y_c, z_c, l, w, h, \theta, \phi, \psi) \in \mathbb{R}^9. \tag{2.35}$$

with center point (x_c, y_c, z_c), length l, width w, height h and the 3D orientation of the bounding box (θ, ϕ, ψ). 3D object detectors for automotive applications [ZT18; YML18; Qi+18; Lan+19] often assume the rotation around the x-axis ψ and the rotation around the y-axis ϕ to be zero so that the regression problem reduces to $\boldsymbol{b} \in \mathbb{R}^7$.

3D object detection can be based on single RGB images [Che+16; Mou+17; BL19; Ma+19b; LWT20], stereo RGB images [Che+17b; QWL19; Pon+20; Guo+21b], single RGB-D images [SX16; LG17; Qi+18], LiDAR point clouds [ZT18; YLU18; YML18; Lan+19] or a fusion of RGB data with LiDAR point clouds [Che+17c; Ku+18; Pai+21]. Unlike 2D object detection which does not require estimating object sizes or distances, 3D object detection can benefit from depth information in the form of stereo input or point cloud data [Wan+23]. Due to the very accurate distance measurement of LiDAR sensors and the high-quality texture acquisition ability of modern RGB cameras, methods combining these two modalities [Cai+23; Wu+23; Hu+23] achieve currently the highest accuracies in the 3D object detection challenges of KITTI [GLU12] and nuScenes [Cae+20].

The KITTI dataset [GLU12; Gei+13], named after the Karlsruhe Institute of Technology (KIT) and the Toyota Technological Institute (TTI), is one of the most influential datasets in the field of autonomous driving. It contains six hours of driving data in and around Karlsruhe including RGB front camera images and LiDAR point clouds. Annotations are provided for a variety of tasks, such as 3D object detection, object tracking, and optical flow estimation. Figure 2.16 illustrates an example scene of the KITTI dataset consisting of a front camera image and the corresponding LiDAR point cloud with 3D bounding box annotations. Please note that only the objects within the field of view of the front camera are annotated.

Further relevant datasets and challenges for 3D object detection are the ScanNet dataset [Dai+17], the SUNRGB-D 3D Object Detection Challenge [SLX15], the Argoverse datasets [Cha+19; Wil+21], and the Waymo Open Dataset [Sun+20]. All of them provide LiDAR point clouds with annotations for oriented 3D bounding boxes including class labels. The recently published Cityscapes 3D dataset and benchmark [Gäh+20], however, provides only RGB data as input for performing 3D object detection in the automotive domain.

Please refer to section 3.2.2 for more details about LiDAR-based 3D object detection methods.

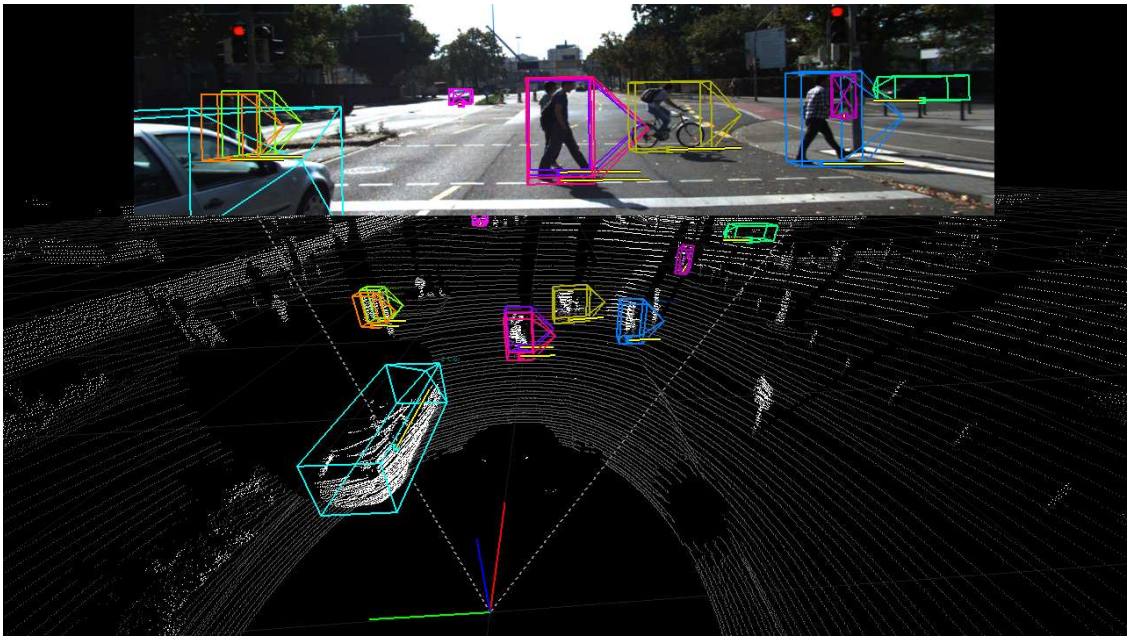


Figure 2.16: Front camera image (top) and LiDAR point cloud (bottom) of the KITTI dataset with annotated 3D bounding boxes. Image adapted from [GLU12].

2.6.5 Semantic Segmentation

The goal of semantic segmentation is to classify each pixel of a given image. Figure 2.15e shows an example of an image containing three different classes, namely Sky, Sheep, and Meadow. Please note, that semantic segmentation does not differentiate between different instances belonging to the same class. The principle of semantic segmentation can also be extended to the three-dimensional space, specifically referred to as 3D semantic segmentation. In this case, given a point cloud, each point is assigned a semantic label.

2.6.6 Instance/Panoptic Segmentation

Instance segmentation assigns each pixel in a given image an instance identifier. That allows for differentiating between multiple objects of the same class as illustrated in figure 2.15f. The process of assigning each pixel a class label together with an instance label is denoted as panoptic segmentation. Analog to 3D semantic segmentation, also instance segmentation and panoptic segmentation can be extended to the three-dimensional space.

2.6.7 6D Pose Estimation

6D pose estimation aims to obtain an accurate prediction of the pose of an object, i.e. its position and orientation in the three-dimensional space. Unlike 3D object detection, no bounding box or extent parameters are estimated. Instead, all objects are known and their 3D models or meshes are given.

A 6D pose is defined by the vector

$$\boldsymbol{b} = (x_c, y_c, z_c, \theta, \phi, \psi) \in \mathbb{R}^6 \tag{2.36}$$

with the center point (x_c, y_c, z_c) and the 3D orientation of the object (θ, ϕ, ψ). Most pose estimators, however, predict the pose in the form of a rigid transformation

$$\boldsymbol{p} = [\boldsymbol{R}|\boldsymbol{t}] \in SE(3) \tag{2.37}$$

where $\boldsymbol{R} \in SO(3)$ is a 3D rotation matrix and $\boldsymbol{t} \in \mathbb{R}^3$ is a 3D translation vector. The 6D pose transformation $\boldsymbol{p}$ converts an object's coordinates, defined in its own coordinate system, into a global reference coordinate system. Figure 2.17 shows an example of an RGB input image (2.17a) and its corresponding 6D pose estimation (2.17b).

(a) RGB input (b) 6D pose estimation

Figure 2.17: Visualization of the prediction of all 6D object poses (b) in a given RGB image (a). For each object, an oriented 3D coordinate system is drawn along with the rendered object transformed by the estimated 6D poses. Images adapted from [HFS22].

Before the emergence of deep learning methods, 6D pose estimators [Low99; Low04; RD06; BTV06; Rot+06] have mostly used a single RGB image as input, extracted hand-crafted features, and matched them to corresponding features in the given 3D models. By leveraging the established 2D-3D correspondences belonging to one object, a Perspective-n-Point (PnP) algorithm [FB81] can be employed to estimate the 6d pose of that object.

With the occurrence of deep learning, CNN-based methods [Xia+18; Keh+17; TSF18] have been proposed which directly regress the 6D object poses. A very popular method among them is PoseCNN by Xiang et al. [Xia+18] which is illustrated in figure 2.18. Given an RGB image, the first stage extracts features by convolutional layers and down-samples them by max pooling. The second stage uses further convolutions to create task-specific embeddings. In the third stage, PoseCNN predicts pixel-wise semantic labels, estimates the 2D pixel coordinates of the object centers, and regresses their distances to the camera. Using the semantic labels, PoseCNN generates 2D bounding boxes for all objects. The 3D translation of an object is computed based on its center and distance, assuming known camera intrinsics. Finally, the 3D rotation in the form of a quaternion is estimated for each object by applying fully connected layers on the 2D bounding box regions.

There are also 6D pose estimation methods, using a single depth image or a single point cloud as input [Hin+16; Vid+18; HB20; Gao+20; Gao+21]. Since the input modality offers useful geometric information, it enables precise pose estimation on datasets where texture information is not required, such as the T-LESS dataset [Hod+17]. However, as methods relying only on depth or point cloud data cannot exploit texture information, they are not suitable for pose estimation tasks where the texture is required to differentiate between otherwise ambiguous object poses, e.g. in the YCB-Video dataset [Xia+18] or in the HOPE dataset [Tyr+22].

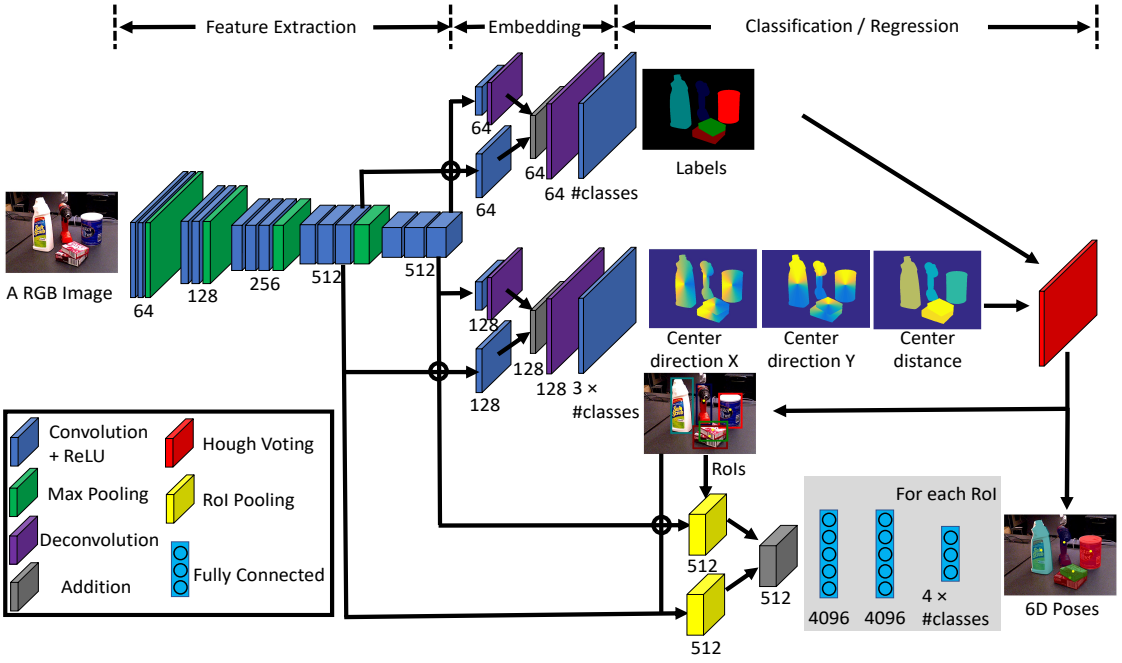


Figure 2.18: Overview of PoseCNN. Image from [Xia+18].

Methods combining RGB and depth data [Bra+14; BI15; Wan+19; He+20; He+21] are currently the most precise methods as they allow the combination of visual and geometric information. A seminal work in this field was the PVN3D network by He et al. [He+20] which performs 6D pose estimation based on predicted 3D keypoints and a semantic segmentation. Their architecture is illustrated in figure 2.19. Given an RGB-D image, PVN3D extracts features from the RGB channels with a CNN pre-trained on ImageNet [Den+09]. The depth image is used to extract features with a PointNet++ [Qi+17b]. A DenseFusion network [Wan+19] is employed to combine the visual with the geometric features resulting in a single feature tensor. This representation is used as input for three MLPs regressing point-wise 3D keypoint offsets, semantic labels, and center point offsets. He et al. utilize eight target keypoints for each object selected by the Farthest Point Sampling (FPS) algorithm [Eld+97] using the known object meshes. They apply a clustering algorithm to differentiate between different object instances and let all points belonging to one instance vote for their target keypoints. Finally, the 6D pose estimates are computed using a least-square fitting algorithm [AHB87].

FFB6D [He+21] is the successor of PVN3D [He+20] with a significant improvement of the feature extraction network. Instead of independently processing RGB and depth input data, the authors propose a deep fusion architecture with multiple bidirectional fusion modules. Figure 2.20 illustrates the network architecture. Similar to PVN3D, FFB6D uses an encoder-decoder CNN for visual feature extraction and a PointNet-based [Qi+17a] point cloud network for geometric feature extraction. However, in each latent hierarchy, they apply a point-to-pixel fusion module and a pixel-to-point fusion module. These modules fuse visual and geometric features by exploiting the pixel-to-point correspondences in the original RGB-D input images. He et al. justify the better performance of this approach by the hypothesis that RGB information can help in the geometric feature extraction process and point cloud information can improve the visual feature extraction. Furthermore, FFB6D uses the Scale Invariant Feature Transform Farthest Point Sampling (SIFT-FPS) algorithm [Low99] to select eight keypoints per object, which are more salient than keypoints selected by FPS.

Over the last few years, many datasets and challenges related to 6D pose estimation have been proposed. Among the most important challenges in this field are the BOP challenges for 6D object pose estimation [Hod+18; Hod+20; Sun+23] aiming to compare 6D pose estimators on a variety of different datasets and metrics. The datasets include YCB-Video [Xia+18], T-LESS [Hod+17], LineMOD [Hin+11b], Occlusion LineMOD [Bra+14], and HomebrewedDB [Kas+19], and HOPE [Tyr+22]. Please refer to chapter 4 for more details about 6D pose estimation methods and datasets.

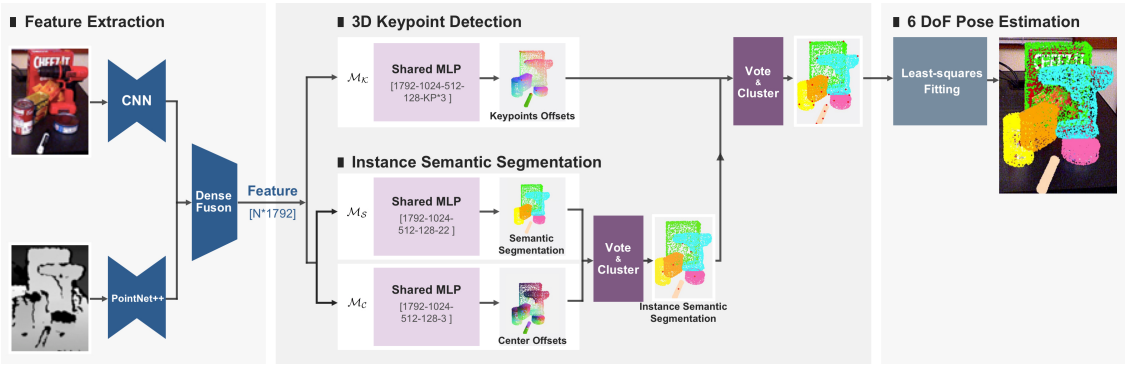


Figure 2.19: Overview of PVN3D. Image from [He+20].

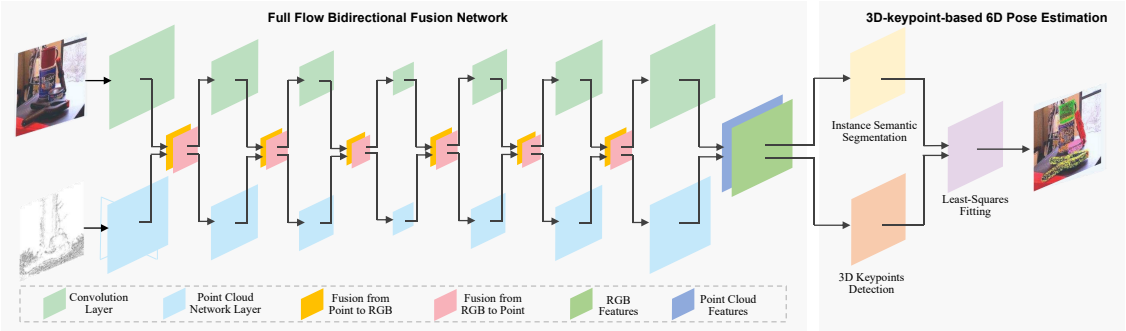


Figure 2.20: Overview of FFB6D. Image from [He+21].

2.6.8 Optical Flow Estimation

Optical flow estimation describes the task of predicting the motions of objects within an image. It provides valuable insights about dynamics within a scene, useful for many computer vision applications including object tracking, visual surveillance, and navigation [Tu+19]. As motions in our world encompass three dimensions, they have to be projected onto the two-dimensional image plane in order to obtain optical flow.

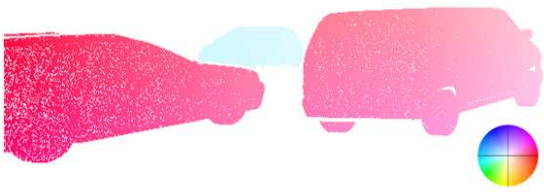
Formally, optical flow is defined as a two-dimensional vector field

$$\boldsymbol{f}(x, y, t) = [f_x, f_y]^{\mathrm{T}} \in \mathbb{R}^2 \tag{2.38}$$

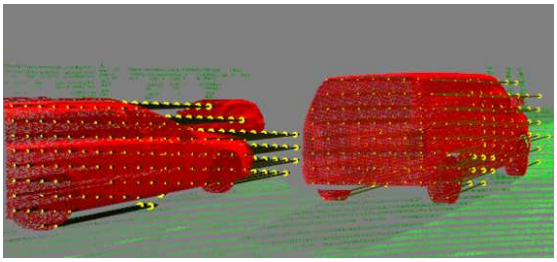
representing the velocity and direction of each pixel (x, y) at time t. Figure 2.21c provides an example of estimated optical flow given two RGB input images at time t_0 (figure 2.21a) and time $t_1 < t_0$ (figure 2.21b).

(a) RGB image at time t_0

(b) RGB image at time $t_1 > t_0$



(c) Optical flow



(d) Scene flow

Figure 2.21: Optical flow and scene flow ground truth based on two consecutively recorded RGB images at time t_0 and t_1. optical flow (bottom left) displays motion parallel to the image plane only. The optical flow (HSV color-coded by direction) describes only the motion projected on the image plane. The scene flow (visualized with 3D vectors in a point cloud) describes the unmitigated 3D motion. Images from [Sch+18].

2.6.9 Scene Flow Estimation

Scene flow is the extension of optical flow from two to three dimensions. It describes the motion of objects in the three-dimensional space. Scene flow is defined as a time-dependent vector field of three-dimensional vectors

$$\boldsymbol{f}(x, y, z, t) = [f_x, f_y, f_z]^{\mathrm{T}} \in \mathbb{R}^3 \tag{2.39}$$

pointing in the direction of movement. The absolute value of the scene flow vectors $|\boldsymbol{f}(x, y, z, t)| = \sqrt{f_x^2 + f_y^2 + f_z^2}$ corresponds to the velocity of the movement. Figure 2.21d depicts the 3D scene flow vectors representing the motion from the given RGB images (figures 2.21a and 2.21b).

As scene flow estimation requires point-wise vector annotations, labeling scene flow datasets is very time-consuming and expensive. Thus, previous work [Dew+16; Ush+17; UE18; VSM18; Bau+19] relies on parts of the KITTI dataset [GLU12] extending it for scene flow estimation based on computations. Another common approach is, to train with the large-scale synthetic scene flow dataset FlyingThings3D [May+16] and fine-tune the model with the KITTI Scene Flow (KITTI-SF) dataset [MHG15; MHG18] as done by [Gu+19; LQG19; LYB19]. However, both FlyingThings3D and KITTI-SF contain one-to-one correspondences between consecutive point clouds which is not the case in raw LiDAR point clouds. The recently published large-scale real-world automotive datasets nuScenes [Cae+20], Argoverse [Cha+19], Argoverse 2 [Wil+21] and the Waymo Open Dataset [Sun+20] do not contain scene flow annotations. However, based on annotated 3D object tracks and ego-motion, a rough estimate for scene flow ground truth annotations can be computed.

Please refer to chapter 3 for related work and more details about scene flow estimation.

2.6.10 Image Generation

Image generation refers to the task of creating new images based on an existing image dataset. The goal is to create images which are not in the dataset but indistinguishable from existing images. Image generation can be either unconditional or conditional. *Unconditional image generation* describes the task of generating images without any specific constraint. In this case, the model generates image samples which are solely based on its knowledge about the data extracted during training. In contrast, *conditional image generation* refers to generating samples constrained to an additional input, for example, a class label, a textual description, or another image. This additional input serves as a condition for the model guiding the generation process and enabling greater control.

Figure 2.22 shows four example images, generated with an unconditional generative model trained on the FFHQ 1024 × 1024 dataset [KLA19]. Figure 2.23 depicts example images, created by a generative model conditioned on different classes of the ImageNet dataset [Den+09].

The most common models for image generation are VAEs, GANs, and diffusion models. Please refer to section 2.5 for more details about these generative models.

Figure 2.22: Results from an unconditional image generation task on the FFHQ 1024 × 1024 dataset. Images taken from [Zha+22a].

(a) Class “maltese dog” (b) Class “volcano” (c) Class “cup”

Figure 2.23: Results from a conditional image generation task on the ImageNet dataset. The generated images are conditioned on the class labels “maltese dog” (a), “volcano” (b), and “cup” (c). Images taken from [SST23].

2.6.11 Image Inpainting

Image inpainting, also called image completion, is a generative computer vision task where missing parts of an input image are filled with plausible content. It can be considered as a form of conditional image generation where the creation of missing image regions is conditioned on the data from the existing regions. The goal is to achieve visual coherence between the existing and the generated regions resulting in an image indistinguishable from unimpaired images of the reference dataset.

Figure 2.24 shows an image inpainting example with a partly masked input image (a), the resulting inpainted image (b), and the corresponding ground truth image (c). It

becomes evident that the unmasked area of the input image corresponds to the ground truth image while the masked area is filled with new data. Even though the filled region deviates notably from the ground truth image, it looks realistic to a certain extent.

(a) RGB input with mask (b) Inpainting result (c) Ground truth

Figure 2.24: Visualization of an example for image inpainting. Given an RGB input image with a masked area (a) an inpainting model generates a new image (b) where the masked area is filled with new content maintaining consistency between the filled region and the surrounding region. The image in (c) shows the corresponding ground truth image which differs slightly in the masked area from the inpainting result. Images taken from [LJ21].

Traditional image inpainting methods can be mainly divided into diffusion-based methods and patch-based methods spreading pixel information gradually in the missing area or copying useful image patches to fill the missing area [Elh+20; Qin+21]. Significant progress has been achieved with CNN-based inpainting methods being able to learn high-level features of the given dataset and thus enabling the generation of more realistic image reconstructions [Zha+23].

The introduction of generative models, such as VAEs [KW14] and GANs [Goo+14], has led to a substantial enhancement in the quality of image inpainting techniques [Qin+21]. As these models can generate multiple reasonable solutions for an inpainting problem, a task variation, called *Pluralistic Image Completion*, has been defined by Zheng et al. [ZCC19]. Pluralistic image completion aims to meet the possible uncertainty and variability in real-world data by generating multiple and diverse plausible solutions.

Furthermore, various attention mechanisms and Transformer architectures have been beneficially leveraged for image inpainting enhancing the quality and the diversity of reconstructions [Xie+19; Zen+20a; QBZ21; Li+22c; SZG23]. With the recent advancements of diffusion models [HJA20] for many different computer vision tasks, also image inpainting has been targeted successfully based on this technology outperforming most VAE-based and GAN-based methods [Lug+22; Xie+23; YCL23].

Please refer to section 5.2.3 for more specific related work about image inpainting relevant to this dissertation.

Chapter 3

Deep Temporal Point Cloud Fusion and Multi-task Learning for Automated Driving Perception

This chapter targets the problem of deep LiDAR point cloud fusion and multi-task learning for 3D object detection and scene flow estimation in autonomous driving scenarios as depicted in figure 3.1. Parts of this chapter are based on my publication "PillarFlowNet: A Real-time Deep Multitask Network for LiDAR-based 3D Object Detection and Scene Flow Estimation" [DB20], written by Fabian Duffhauss and Stefan A. Baur, which has been published in the Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2020. Please refer to the first main row of tables 1.1 and 1.2 to see the fusion modalities and most important challenges addressed in this chapter in comparison to the other main chapters.

3.1 Introduction

Automated vehicles require a robust and precise understanding of their environment in order to drive safely on public roads. In order to fulfill this requirement, it is crucial to obtain accurate knowledge about other road users within the surrounding area including their position, expanse, and motion.

The position and the expanse of road users can be attained by performing a 3D object detection, which predicts oriented 3D bounding boxes for each object in a scene. For inferring information about the motion of road users in dynamic scenes, a 3D scene flow estimation can be conducted. 3D scene flow associates a displacement vector to each point in a point cloud, propagating it forward to its corresponding location in the consecutive point cloud [Ved+99]. Estimating 3D scene flow using LiDAR data is no trivial task: The inherent sparsity of measured points in 3D space renders the problem of scene flow estimation ill-posed, as there are practically no point-wise one-to-one correspondences present in two consecutive point clouds. Instead, scene flow must be inferred from the underlying motion of objects in a scene, irrespective of whether a displacement between

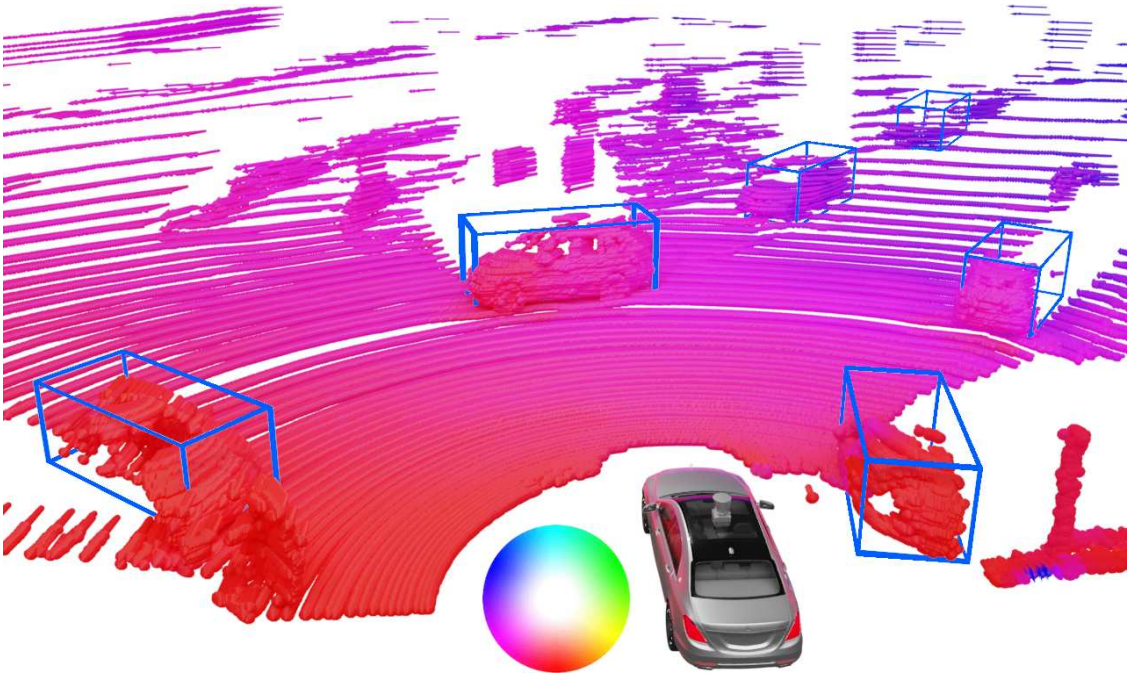


Figure 3.1: Work principle of our multi-task learning network PillarFlowNet. Based on two consecutively recorded LiDAR point clouds, PillarFlowNet detects objects as 3D bounding boxes and predicts scene flow at the same time. This scene shows the ego vehicle performing a sharp right turn. For color coding, the predicted flow vectors are projected onto the ground plane and colored according to the standard HSV (Hue Saturation Value) wheel next to the ego vehicle.

two point clouds is caused by the sensor's ego-motion or whether it is caused by a moving agent.

Recently, the research community's interest in LiDAR object detection and LiDAR scene flow estimation has grown but little work has been conducted towards unifying those tasks, i.e. solving them with a single network. To the best of our knowledge, there exists only one previous method for multi-task learning for these tasks using LiDAR data, which is the PointFlowNet method by Behl et al. [Beh+19b]. Their network relies on computationally expensive 3D convolutions in order to find correspondences over the space and time domain, rendering their network architecture too slow for real-time applications. We propose a novel network utilizing a different representation, which is not only significantly faster but also much more accurate in both tasks, outperforming their network in predicting LiDAR scene flow and objects at the same time. It relies on a pillar feature representation in combination with efficient 2D convolutions. This gain in speed makes our network the first LiDAR object detection and scene flow multi-task network suitable for systems with real-time constraints.

Point-wise scene flow in combination with object detection is more generic than object tracking approaches, as the motion of objects that were not explicitly detected can still be reasoned about. Traditional approaches using object detection in combination with tracking have advantages when it comes to the stability of detections and vehicle tracks. However, tracking cannot capture motion in cases where no vehicle has been detected. Additionally, tracking frameworks need time to initialize and update their motion model in order to predict the meaningful state of a tracked object. In contrast to that, our network computes meaningful motion estimates even for objects it has not been specifically trained for as shown in figure 3.2.

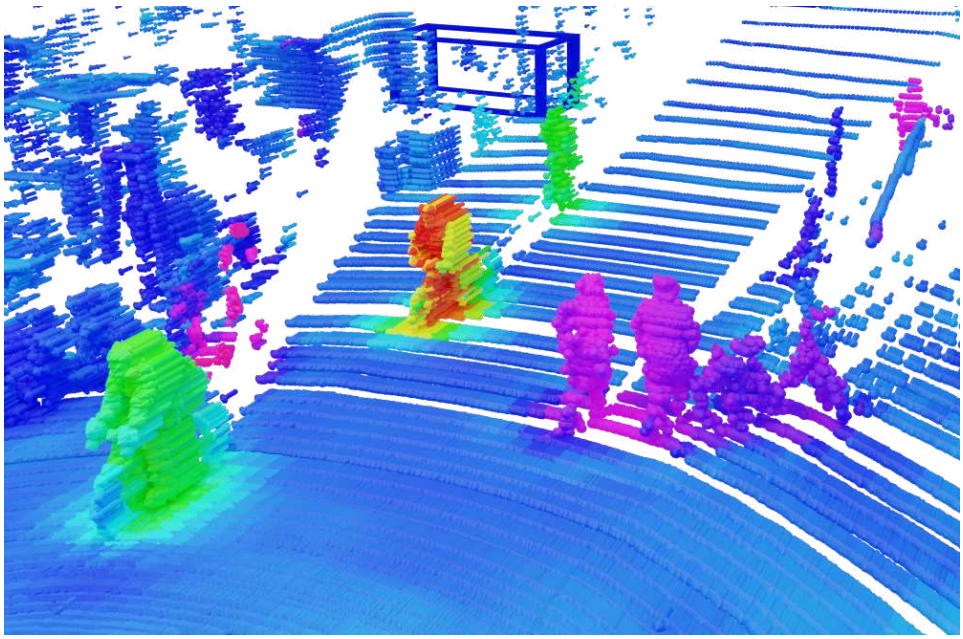


Figure 3.2: PillarFlowNet captures motion at a very detailed level. Color-coding the predicted flow vectors by length makes pedestrians and cyclists easily distinguishable from vehicles and static objects.

Our main contributions concerning the environmental perception of automated vehicles are:

- We propose PillarFlowNet, a novel end-to-end trainable network for simultaneous LiDAR object detection and scene flow estimation.
- Our method significantly improves multi-task LiDAR scene flow estimation and object detection accuracy compared to the state-of-the-art.
- PillarFlowNet is the first multi-task LiDAR scene flow and object detection network to achieve real-time performance.

3.2 Related Work

Our method lies at the intersection of LiDAR scene flow estimation, LiDAR object detection, and multi-task learning. The following paragraphs present relevant literature for all three categories.

3.2.1 LiDAR-based Scene Flow Estimation

Over the years, multiple data representations for LiDAR point clouds have been established for estimating scene flow. The most prominent methods are based on graphs, point sets, and grids.

Graph-based Methods

Dewan et al. [Dew+16] were among the first to research estimating LiDAR scene flow using energy minimization on a factor graph representation of consecutive point clouds. Their approach assumes geometric constancy and parameterizes local geometry with **S**ignature of **H**istograms of **O**rien**T**ations (SHOT) descriptors [TSD10] to find correspondences. Recently, Gu et al. have proposed HPLFlowNet [Gu+19], capable of estimating LiDAR scene flow on two consecutive point cloud frames with up to 86k points per frame. They transform the unordered point cloud onto a permutohedral lattice [ABD10] and employ Bilateral Convolutional Layers (BCL) [KJG15; JKG16] to predict scene flow in an end-to-end fashion.

Point Set Based Methods

FlowNet3D by Liu et al. [LQG19] operates on pre-filtered sets of points. It simultaneously learns hierarchical features based on a PointNet++ [Qi+17b] architecture and flow embeddings representing point motions before predicting point-wise 3D scene flow vectors.

The follow-up work by Liu et al. [LYB19], called MeteorNet, introduces a *Meteor* module to compute point-wise features for points and their neighbors. Features and spatiotemporal differences of neighboring points are passed into a feature encoder inspired by PointNet++ [Qi+17b], aggregated recursively, and shared using max pooling and feature concatenation.

Graph-based and point set based methods have in common that they tend to work well on small point sets. The method reporting the largest point cloud to be processed is HPLFlowNet [Gu+19] with 86k points, while FlowNet3D [LQG19] and MeteorNet [LYB19] report experiments with around 8k points.

Even in the case of reducing the point cloud size to 8k points, FlowNet3D has a runtime of 325.9 ms which is more than three times larger than acceptable for real-time applications.

For the sake of comparison, a Velodyne HDL64 LiDAR sensor records around 128k points per revolution at 10 Hz. All of these methods circumvent this problem by removing ground points in a preprocessing step, which has several disadvantages: Besides costing precious runtime, information is lost which can be fatal, especially in cases where the ground segmentation gives false results.

Grid-based Methods

Ushani et al. [Ush+17; UE18] were the first to apply machine learning to LiDAR scene flow estimation, inserting point clouds with removed static background into 3D occupancy grids. Scene flow is estimated using binary classifiers in order to find matches of feature columns between consecutive occupancy grids. More recent work tackles LiDAR scene flow estimation using Deep Learning. Vaquero et al. [VSM18] and Baur et al. [Bau+19] use range images as data representation, projecting point clouds into 2D grids and applying 2D convolutions to estimate flow. Wang et al. [Wan+18a] use a ResNet-50 architecture [He+16] applying 3D convolutions on an occupancy grid predicting a voxel flow map. They introduced Deep Parametric Continuous Convolutional (DPCC) layers which are then used as refinement on the 3D occupancy flow map. DPCC layers employ the Euclidean space directly as a support domain with MLPs as kernel functions. Compared to other publications such as [Beh+19a; LYB19; LQG19; Gu+19; VSM18; Bau+19; Wu+20] they report exceptionally good scene flow precision, which is why we chose to implement their approach as a baseline in section 3.6.

3.2.2 LiDAR-based 3D Object Detection

Recent methods for LiDAR-based 3D object detection are mostly projection-based, volumetric-based, or point-based [AB22]. Please refer to sections 2.3.1 to 2.3.3 for more background about these point cloud processing principles.

Most relevant for our work are **volumetric-based methods**. A seminal work in this field has been the single-stage, end-to-end trainable 3D object detector VoxelNet proposed by Zhou et al. [ZT18]. It first subdivides the LiDAR point cloud into a voxel grid and then applies Voxel Feature Encoding (VFE) layers to each voxel. The VFE layers perform hierarchical feature encoding based on PointNet [Qi+17a]. A fully connected network with maximum pooling creates voxel-wise features which are collected in a sparse 4D tensor. After some 3D convolutional middle layers, the resulting tensors are ultimately fed into a Region Proposal Network (RPN) [Ren+17] which outputs a map of objectness probability scores and a regression map for bounding box parameters.

As VoxelNet has a huge computational cost due to the large 3D convolutional backbone network, Yan et al. have improved VoxelNet by introducing more efficient sparse convolutions in their network SECOND [YML18]. Additionally, they adopted the focal loss [Lin+17b] for bounding box classification to tackle the problem of imbalances between

the foreground and background classes. Lang et al. [Lan+19] have further improved SECOND by splitting the input point cloud into vertical 3D columns (pillars) instead of voxels and by introducing a pillar feature network. The pillar feature network utilizes PointNet [Qi+17a] to learn feature vectors from all or a subset of points within that pillar. The features are encoded in a pseudo image enabling the use of an efficient 2D object detection backbone network without requiring 3D convolutions. This allows PointPillars to predict 3D bounding boxes at very high speed.

3.2.3 Multi-Task Learning

Multi-task learning attempts to learn multiple tasks simultaneously with the goal of obtaining more general models. While there is a multitude of works that deal with multi-task learning in general [Tei+18; SK18; Zam+18; WFU15; KGC18] or multi-task learning specifically for LiDAR applications [Ren+18; Qi+17a; LYU18; Lia+19] there exists only one paper that deals with simultaneous scene flow estimation and object detection using LiDAR point clouds: PointFlowNet by Behl et al. [Beh+19a] subdivides two point clouds into separate voxel grids and uses Siamese VFE layers [ZT18] as feature encoder to compute independent feature maps for each one of the consecutive point clouds. These two feature maps are then stacked and fed into a context encoder, which applies vertical downsampling using three strided 3D convolution layers, enforcing a common data representation for all tasks. After this, the network is split into different branches, predicting 3D scene flow for each voxel, ego-motion, and object classification scores along with residuals for the box proposals. They evaluate the scene flow accuracy of their network on a subset of the KITTI object detection dataset [GLU12], comparing it to multiple baseline methods.

The advantages of our method over the method by Behl et al. [Beh+19a] are mainly twofold: First, our network is significantly more accurate in both object detection and scene flow estimation. Second, our network is more than twice as fast as theirs, making it the first LiDAR scene flow and object detection multi-task network capable of running in real-time. This is mainly due to using a different, more efficient data representation, enabling us to rely on 2D convolutions instead of computationally expensive 3D convolutions.

3.3 PillarFlowNet Architecture

PillarFlowNet takes two consecutively recorded LiDAR point clouds as input, each accumulated over 360°. Using a deep neural network, it predicts oriented 3D bounding boxes for cars and vans as well as a 3D scene flow vector for each point in the first point cloud. Figure 3.3 shows the overall structure of the network. The network architecture is subdivided into three parts: Firstly, two feature encoding networks create individual feature representations of the two input point clouds. Secondly, a convolutional backbone

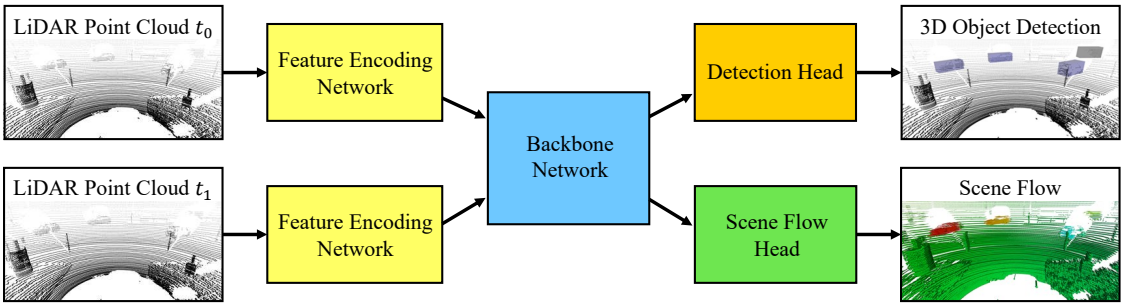


Figure 3.3: Structure of our proposed multi-task architecture PillarFlowNet. The network works directly on two consecutive raw point clouds and predicts 3D bounding boxes as well as scene flow for all points in the scene.

network fuses the point cloud representations and learns a shared feature representation. Finally, two output heads are employed. The first one is dedicated to 3D bounding box classification and regression, while the second one performs the estimation of 3D scene flow vectors.

3.3.1 Feature Encoding Network

In order to encode both input point clouds to be processed efficiently, we use two identical learning-based feature encoding networks whose structure is illustrated in figure 3.4. It is based on the pillar feature network introduced in PointPillars by Lang et al. [Lan+19].

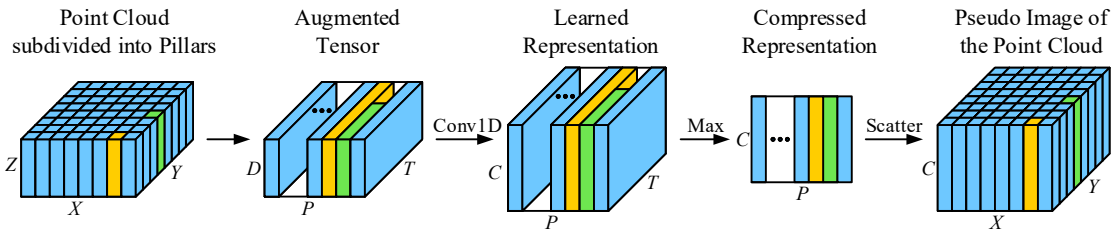


Figure 3.4: Pillar-based feature encoding network of our method PillarFlowNet. It divides a raw point cloud into a grid of vertical pillars and learns a feature representation that can be further processed efficiently without 3D convolutions. The figure shows how two example pillars containing $T < 100$ points (green) and $T \geq 100$ points (orange) are processed by the network. Zero padding is applied to fill the tensors with $T < 100$ points.

The feature encoder subdivides the input point cloud into equally spaced vertical pillars. Empty pillars are discarded, so that a variable number of pillars P is further processed. If a pillar comprises more than $T = 100$ points, a random subset is kept. All remaining points in each pillar are augmented using Cartesian coordinates relative to the pillar's centroid and relative to the pillar's center, creating a tensor with $D = 9$ dimensions as

described in [Lan+19]. Using a linear layer, a batch normalization layer, and a rectified linear activation function (ReLU), a feature representation of the augmented point cloud tensor is learned. C denotes the number of channels. A maximum operation is applied over the T point features in each pillar in order to extract only the most dominant features.

The elements of the resulting tensor with shape (C, P) are scattered into a 3D tensor at their original pillar locations. This yields a tensor with dimensions (X, Y, C) that serves as a compact feature representation of the input point cloud which can be efficiently processed by 2D convolutions exploiting spatial relations between the feature vectors.

3.3.2 Backbone Network

The two pseudo images of the feature encoders are further processed by a convolutional backbone network which is illustrated in figure 3.5. The backbone network concatenates the pseudo images and applies strided 2D convolutions to create a compact shared representation. Features from differently sized tensors are upsampled using transposed 2D convolutional layers, combining high resolution local features with coarser, more globally aggregated features in a subsequent concatenation. This combines semantic with positional information. While the full pillar resolution is maintained for scene flow estimation, for object detection only half of the resolution proved to be sufficient. For more details regarding number of layers and tensor sizes refer to figure 3.5.

3.3.3 Output Heads

The concatenated feature tensors are further processed by 2D convolutions for object classification, bounding box regression, and 3D scene flow estimation as shown on the right side of figure 3.5. For scene flow estimation, a 3D velocity vector is regressed for each pillar directly. For 3D object detection, a region proposal network with fixed-size anchor boxes of two orientations as in [ZT18] is applied. For each anchor box, the regression head predicts the probability that an object is located within it. The regression head predicts the deviations for each anchor box to the actual object using the same encoding as [ZT18; YML18]:

$$b = (\Delta x^{\mathrm{p}}, \Delta y^{\mathrm{p}}, \Delta z^{\mathrm{p}}, \Delta l^{\mathrm{p}}, \Delta w^{\mathrm{p}}, \Delta h^{\mathrm{p}}, \Delta \theta^{\mathrm{p}}) \in \mathbb{R}^7. \tag{3.1}$$

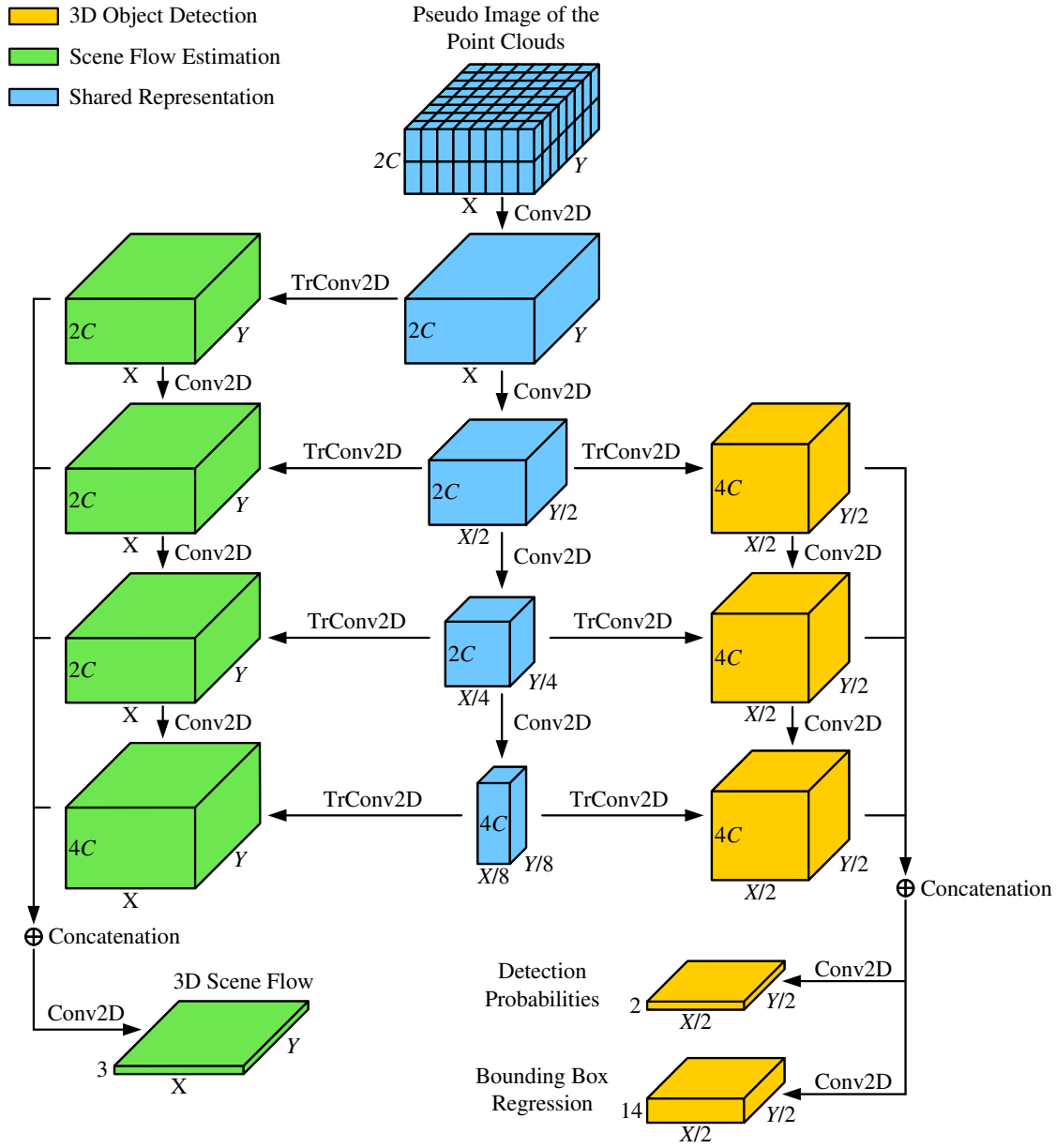


Figure 3.5: Convolutional backbone network with output heads. Given the learned 3D representations of both input point clouds, the backbone network initially creates shared feature tensors applying 2D convolutions. Using transposed convolutions, features from different aggregation stages are taken into account to finally predict 3D bounding boxes with detection probabilities and 3D scene flow vectors.

3.4 Implementation Details

In this section, we present details about our implementation, such as hyperparameters and the loss function.

3.4.1 Network Details

For object detection, we use anchor boxes with 3.9 m in length, 1.6 m in width, and 1.56 m in height centered at $z = -1$ m. We use pillars with a square base of 0.2 m in width and a maximum number of points within a pillar of $T = 100$. The linear layer of our feature encoding network has $C = 64$ units. We train our network on a single GPU with a mini-batch size of one. We use an Adam optimizer [KB15] with initial learning rate 1×10^{-3} and an exponential decay factor of 0.85 applied every ten epochs.

3.4.2 Loss

Our proposed network is end-to-end trainable using a multi-task loss function that considers the homoscedastic (task-dependent) uncertainties, inspired by Kendall et al. [KGC18]. However, instead of combining multiple pixel-based computer vision tasks, we apply the principle to scene flow estimation and object detection with the total loss

$$\mathcal{L}_{\text{total}} = \frac{1}{2\sigma_{\text{OD}}^2}\mathcal{L}_{\text{OD}} + \frac{1}{2\sigma_{\text{SF}}^2}\mathcal{L}_{\text{SF}} + \log \sigma_{\text{OD}}^2\sigma_{\text{SF}}^2, \tag{3.2}$$

where σ_{OD} and σ_{SF} are learnable parameters for object detection (OD) and scene flow (SF) respectively. We initialize these values with one and let them optimize during training.

Object Detection Loss

The object detection loss $\mathcal{L}_{\text{OD}}$ is comprised of a classification loss $\mathcal{L}_{\text{cls}}$ and a bounding box regression loss $\mathcal{L}_{\text{reg}}$:

$$\mathcal{L}_{\text{OD}} = \lambda_{\text{cls}}\mathcal{L}_{\text{cls}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}. \tag{3.3}$$

λ_{cls} and λ_{reg} are manually chosen weighting factors, which we set to $\lambda_{\text{cls}} = 1$ and $\lambda_{\text{reg}} = 2$.

For classification, the focal loss [Lin+17b] is used, whereby all anchor boxes are declared as positive or negative like in Faster R-CNN [Ren+17] with positive and negative IoU thresholds of 0.6 and 0.45 respectively:

$$\mathcal{L}_{\text{cls}} = -\frac{\lambda_{\text{cls}}^{\text{pos}}}{N^{\text{pos}}}\sum_i (1 - p_i^{\text{pos}})^\gamma \log(p_i^{\text{pos}}) - \frac{1 - \lambda_{\text{cls}}^{\text{pos}}}{N^{\text{neg}}}\sum_i (p_i^{\text{neg}})^\gamma \log(1 - p_i^{\text{neg}}). \tag{3.4}$$

$\lambda_{\text{cls}}^{\text{pos}}$ is the balancing factor and γ the focusing parameter. We set them to $\lambda_{\text{cls}}^{\text{pos}} = 0.25$ and $\gamma = 2$. p_i^{pos} and p_i^{neg} are the classification probabilities of the i-th positive and negative anchor box respectively. N^{pos} and N^{neg} are the numbers of positive and negative anchor boxes respectively.

For bounding box regression, the smooth $\mathcal{L}_1$ loss of Girshick et al. [Gir15] and the angle loss of Yan et al. [YML18] are combined:

$$\mathcal{L}_{\text{reg}} = \frac{1}{N^{\text{pos}}} \left(\mathcal{L}_{1,\,\text{smooth}}(\sin(\Delta\theta^{\text{g}} - \Delta\theta^{\text{p}})) + \sum_{i \in \{x,y,z,l,w,h\}} \mathcal{L}_{1,\,\text{smooth}}(\Delta i^{\text{g}} - \Delta i^{\text{p}}) \right) \tag{3.5}$$

Scene Flow Loss

As most points are static in the world, there usually is an imbalance between static and dynamic points in real-world datasets. This imbalance impairs the network to generalize well for both types of points. Therefore, we use a weighted $\mathcal{L}_1$ loss for scene flow estimation

$$\mathcal{L}_{\text{SF}} = \frac{1}{K} \sum_{k=1}^{K} \delta_k \left\| \boldsymbol{f}_k^{\text{g}} - \boldsymbol{f}_k^{\text{p}} \right\|_1, \tag{3.6}$$

where $\boldsymbol{f}_k^{\text{g}}$ and $\boldsymbol{f}_k^{\text{p}}$ are the k-th ground truth and the corresponding predicted flow vector respectively. δ_k is a weighting factor for dynamic points.

3.5 Experimental Setup

For evaluating our approach and comparing it with the state-of-the-art, we have conducted extensive experiments which are explained and presented in the following.

3.5.1 Dataset

For our experiments, we have employed the KITTI Object Tracking dataset [GLU12], which was recorded using a Velodyne HDL64 LiDAR sensor at a frame rate of 10 Hz. We have used the ego vehicle's odometry to annotate the static LiDAR points with ground truth scene flow labels by applying its inverse transformation to every point.

For dynamic objects, point-wise scene flow ground truth is computed using the annotated bounding boxes and their corresponding transformation between consecutive frames. To cope with the high imbalance between static and dynamic points in the KITTI dataset, we have chosen the scene flow loss weighting factor δ_k in equation (3.6) as $\delta_k = 1$ for static points and $\delta_k = 10$ for dynamic ones.

For training and testing of the proposed network and all benchmarks, we have split the KITTI Object Tracking dataset into two sets according to table 3.1. The sequences

are selected in such a way that both sets have on average a similar number of static and dynamic points. Please refer to appendix A.1 for a more detailed analysis of the KITTI Object Tracking dataset.

	Training data	Test data
Selected sequences	0 – 2, 4, 5, 7, 8, 10, 11, 13 – 18, 20	3, 6, 9, 12, 19
Number of frames	5 633	2 349
Total ratio	70.6 %	29.4 %

Table 3.1: Sequence allocation for training and test data of the extended KITTI Object Tracking dataset.

3.5.2 Data Augmentation and Preprocessing

We apply a set of data augmentation methods to the LiDAR point clouds of our extended KITTI Object Tracking dataset, in order to improve the generalization performance on unseen driving sequences. For each of these augmentations, we transform the ground truth annotations of scene flow vectors and 3D bounding boxes accordingly to ensure correctness.

Firstly, we mirror the given training data sample with a 50 % chance across the xz-plane. Secondly, we rotate the data around the z-axis by an angle drawn from the uniform distribution $[-45°, 45°]$. Thirdly, we insert additional moving objects sampled from the training set into each pair of training frames to improve the detection and scene flow estimation for dynamic objects. When doing so, we paste the objects to their original position rotated randomly around the z-axis with the restriction to the area in from of the ego vehicle. Thus, the objects keep their original distance to the LiDAR sensor ensuring a realistic point density.

Even though the LiDAR point clouds of the KITTI dataset [GLU12] contain points accumulated over $360°$, we follow the common practice [ZT18; YML18; Lan+19] to crop the point cloud to $[0, 70.4]\,\mathrm{m} \times [-40, 40]\,\mathrm{m} \times [-3, 1]\,\mathrm{m}$ for x, y, and z respectively, as objects are only annotated in the field of view of the front cameras. Figure 3.6 shows an example scene from the KITTI Object Tracking dataset after data augmentation and cropping.

3.5.3 Evaluation Metrics

We evaluate our network and the benchmarks on the KITTI Object Tracking test set presented in table 3.1. For object detection, we use the Average Precision (AP) score at 70 % intersection over union. For scene flow estimation, we use the Average End-point

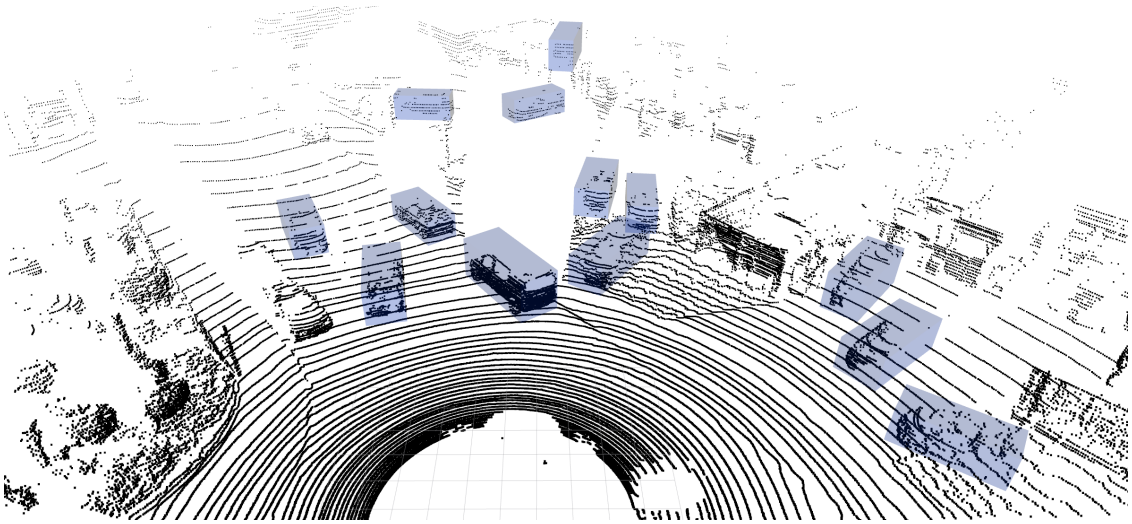


Figure 3.6: Augmented LiDAR point cloud from the KITTI Object Tracking dataset with annotated 3D bounding boxes. Ten additional objects are inserted in this scene.

Error (AEE), a modified version of the Average Cosine Distance (ACD), the percentage of inliers (endpoint error $EE < 10\,\text{cm}$), and the percentage of outliers ($EE > 30\,\text{cm}$).

Since static objects dominate the scene in most autonomous driving scenarios, the accuracy of scene flow vectors belonging to moving objects has only a small impact on metrics that consider all points equally. This diminishing effect on metrics is in stark contrast to the actual relevance of correctly estimating dynamic objects' motions on the safety of operation for an autonomous vehicle. Therefore, we additionally report each metric for dynamic and static points separately in the scene. We define dynamic points as all points in the first point cloud which are inside ground truth bounding boxes.

Regarding the average cosine distance, we propose an adaption which does not penalize the prediction of small scene flow vectors where the direction of the ground truth can be noisy. This is achieved by averaging the cosine distance of K_ϵ ground truth flow vectors, where $\left\|\boldsymbol{f}_k^{\mathrm{g}}\right\|_2 > \epsilon$, i.e.

$$\mathrm{ACD} = \frac{1}{K_\epsilon} \sum_{k=1}^{K_\epsilon} 1 - S_{\mathrm{c}} \left(\boldsymbol{f}_k^{\mathrm{g}}, \boldsymbol{f}_k^{\mathrm{p}} \right). \tag{3.7}$$

S_{c} is the cosine similarity

$$S_{\mathrm{c}} \left(\boldsymbol{f}_k^{\mathrm{g}}, \boldsymbol{f}_k^{\mathrm{p}} \right) = \frac{\boldsymbol{f}_k^{\mathrm{g}} \boldsymbol{f}_k^{\mathrm{p}}}{\left\|\boldsymbol{f}_k^{\mathrm{g}}\right\|_2 \left\|\boldsymbol{f}_k^{\mathrm{p}}\right\|_2} \tag{3.8}$$

We set $\epsilon = 0.05$ as threshold.

3.5.4 Baseline Methods

We compare the performance of our proposed method to multiple baseline methods:

For **simultaneous scene flow & object detection** we use PointFlowNet by Behl et al. [Beh+19a] as a baseline which is to the best of our knowledge the only previous method for this task combination based on LiDAR data.

For **scene flow**, we choose the Deep Parametric Continuous Convolutional Network (DPCCN) as proposed by Wang et al. [Wan+18a] as a baseline. In their study conducted on their non-public dataset, they report an exceptionally small average endpoint error of just 7.8 cm. However, for adapting their network to our experimental setup, we exchange the ResNet-50 with a ResNet-34 which improves the generalization on the KITTI Object Tracking dataset significantly. Due to its excessive memory requirements, it was not possible to use FlowNet3D [LQG19] for the full KITTI point clouds. In addition, we employ PointFlowNet [Beh+19a] focused on scene flow, i.e. with the object detection loss set to zero.

For **object detection**, we use PointPillars [Lan+19] as a baseline, as it represents the previous state of the art in single shot LiDAR 3D object detection. In order to get the best possible object detection performance, we do not limit the maximum number of pillars. Additionally, we consider PointFlowNet [Beh+19a] focused on object detection, i.e. with the scene flow loss set to zero.

Since the baseline methods as reported in their corresponding papers are not adapted to the KITTI Object Tracking dataset, we have performed an extensive parameter search in order to find the best-suited hyperparameters and report only the best results achieved by each method. For all object detection approaches, the data augmentation methods as described in section 3.5.2 are used. For pure scene flow estimation, augmentation is not necessary to achieve improved results.

3.6 Results

The results of the experiments described in section 3.5 are presented and discussed in the following. The quantitative results for the scene flow estimation are shown in tables 3.2 and 3.3. The quantitative object detection results are shown in table 3.4. Results obtained in a multi-task setting (i.e. simultaneous object detection and scene flow prediction) are labeled as MT in the tables. Results obtained in a single-task setup (i.e. either object detection or scene flow estimation) are labeled as ST.

Qualitative results, in form of a visualization of simultaneous predictions of objects and scene flow of PillarFlowNet are shown in figure 3.7.

	Mode	Aug	Average endpoint error			Average cosine distance		
			All	Static	Dynamic	All	Static	Dynamic
PointFlowNet	MT	✓	11.7 cm	11.2 cm	39.2 cm	0.267	0.079	0.571
PillarFlowNet	MT	✓	9.2 cm	8.7 cm	29.9 cm	0.216	0.054	0.451
PointFlowNet	ST	×	10.4 cm	10.1 cm	21.4 cm	0.145	0.090	0.245
DPCCN	ST	×	9.2 cm	8.6 cm	48.0 cm	0.275	0.049	0.641
PillarFlowNet	ST	×	**6.9 cm**	**6.7 cm**	**17.9 cm**	**0.091**	**0.046**	**0.186**

Table 3.2: Quantitative results for the scene flow estimation on the average endpoint error and the average cosine distance metrics. Data augmentation (Aug) is only applied during training of the multi-task (MT) methods. The best results are printed in bold.

	Mode	Aug	Inlier rates			Outlier rates		
			All	Static	Dynamic	All	Static	Dynamic
PointFlowNet	MT	✓	58.3 %	59.2 %	34.2 %	5.4 %	4.8 %	19.2 %
PillarFlowNet	MT	✓	79.4 %	80.4 %	52.5 %	4.9 %	4.4 %	18.4 %
PointFlowNet	ST	×	73.0 %	73.5 %	60.2 %	5.6 %	5.3 %	14.0 %
DPCCN	ST	×	70.4 %	71.4 %	42.6 %	2.6 %	2.0 %	18.2 %
PillarFlowNet	ST	×	**81.2 %**	**81.7 %**	**68.7 %**	**1.5 %**	**1.1 %**	**12.1 %**

Table 3.3: Inlier and outlier rates for the scene flow estimation. The best results are printed in bold.

3.6.1 Multi-Task Performance

While the object detection performance of PillarFlowNet does not reach the object detection performance of the single-task network PointPillars [Lan+19], it outperforms the multi-task baseline PointFlowNet [Beh+19a] for object detection significantly. In terms of scene flow, PillarFlowNet's average endpoint errors are significantly lower than PointFlowNet's. Inlier rates for both static and dynamic points are on average 20 percentage points higher than PointFlowNet's. Also regarding outlier rates, PillarFlowNet is slightly better than PointFlowNet. Remarkably, PillarFlowNet trained in a multi-task setup matches the state-of-the-art DPCCN [Wan+18a] for scene flow estimation in terms of average endpoint error for static points while being over 37.7 % more accurate on dynamic points.

It is well known that certain tasks in certain scenarios can have synergetic relationships [Zam+18]. In order to investigate synergy effects when performing multi-task learning, we have additionally trained PointFlowNet [Beh+19a] and our PillarFlowNet for both

Method	Mode	AP@0.7
PointFlowNet	MT	38.4 ‰
PillarFlowNet	MT	54.7 ‰
PointFlowNet	ST	45.3 ‰
PointPillars	ST	**66.3 ‰**
PillarFlowNet	ST	65.4 ‰

Table 3.4: Results of the 3D object detection. The best result is printed in bold.

tasks individually by setting either the scene flow loss or the object detection loss to zero. The results of these single-task experiments are discussed in the following section.

3.6.2 Single-task Performance

Our experiments suggest that for both PillarFlowNet and PointFlowNet [Beh+19a] the synergetic effects from training two seemingly supportive tasks are smaller than the effects from training on a single task individually. Both networks perform better in either object detection or scene flow estimation when being trained to perform a single task exclusively. This demonstrates that for this combination of architectures and tasks, the beneficial effect of focusing on a single task outweighs the synergetic effects that may exist for scene flow and object detection.

Scene Flow

As can be seen in table 3.2, in a single-task setting, PillarFlowNet surpasses state-of-the-art scene flow estimation baselines. PillarFlowNet is also much more robust than previous approaches, achieving significantly higher inlier rates and significantly lower outlier rates than the baseline methods. As it is to be expected, all methods perform much better on static points than on points belonging to dynamic objects. This underlines the importance of evaluating the performance for static and dynamic objects separately.

3D Object Detection

LiDAR-based 3D object detection is a more thoroughly researched field compared to LiDAR-based scene flow estimation. The dedicated object detection network outperforms our architecture which is designed to perform well on multiple tasks. Noticeably, the performance gap of our multi-task network PillarFlowNet towards the state-of-the-art network PointPillars [Lan+19] is much smaller than the gap of PointFlowNet [Beh+19a] towards PointPillars.

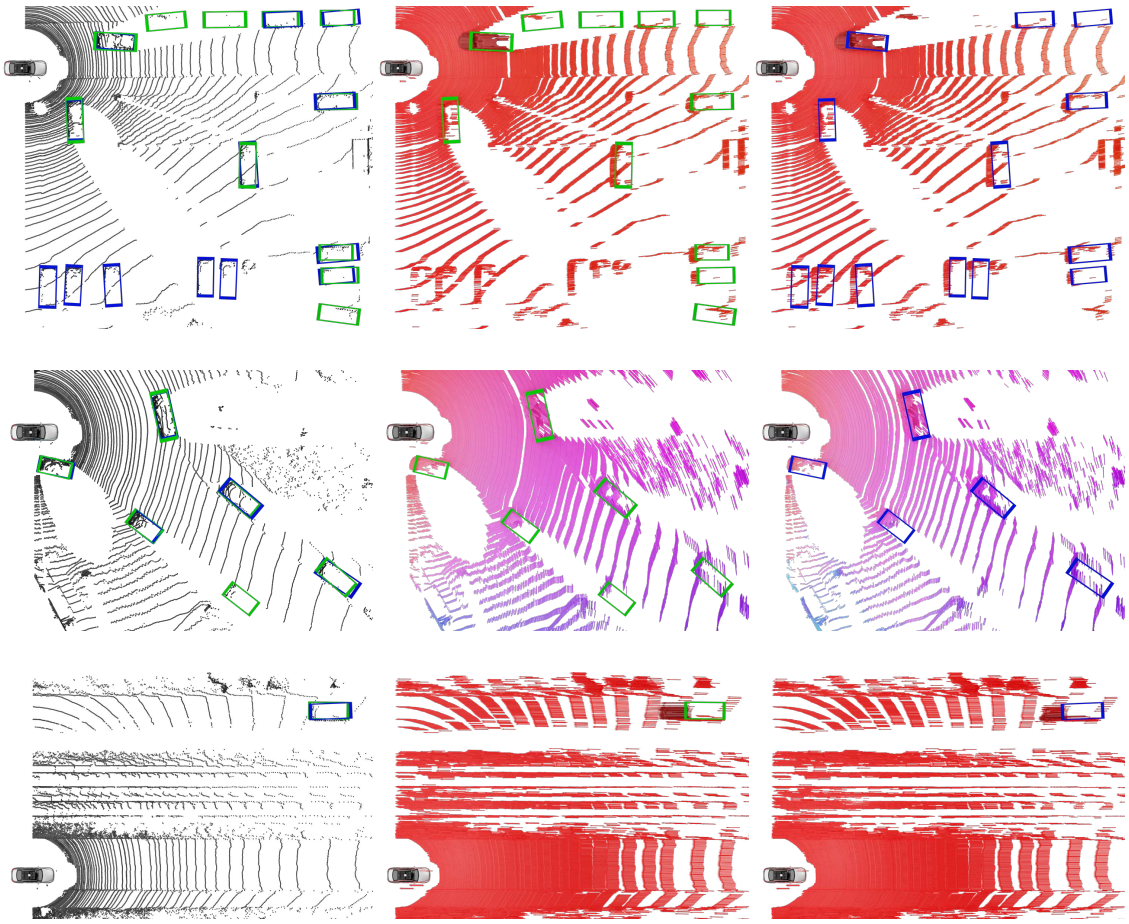


Figure 3.7: Visualization of the prediction results of PillarFlowNet. The first column of images shows PillarFlowNet's object detection prediction (blue) overlayed with ground truth objects (green). Please note that PillarFlowNet detects objects outside of the space that objects were labeled in. The KITTI Object Tracking dataset only contains annotations for objects in the field of view of the front camera. In the middle column, scene flow and object ground truth are shown. In the right image column, PillarFlowNet's predicted scene flow and objects are shown. Flow vectors are HSV color-coded according to their direction.

3.6.3 Runtime

The runtimes were evaluated on a desktop computer with an AMD Ryzen 9 3900X CPU with 3.8 GHz and an NVIDIA GeForce RTX 2080 Ti GPU using a TensorFlow [Aba+16] implementation. The inference of our multi-task network takes 87.6 ms. Object detection without scene flow estimation takes 71.7 ms and scene flow estimation with deactivated object detection takes 86.6 ms. In comparison, PointFlowNet [Beh+19a] requires with 197.8 ms 2.3 times more for the same inference.

3.7 Conclusion

This chapter has introduced PillarFlowNet, a novel end-to-end trainable network for simultaneous LiDAR object detection and scene flow estimation capable of real-time performance. We have employed a pillar-based feature encoding network for efficient LiDAR point cloud compression enabling our subsequent backbone network to learn a shared feature representation of two consecutive point clouds. Based on that representation, 3D scene flow vectors and oriented 3D bounding boxes including detection probabilities are inferred. In comprehensive experiments on the KITTI object tracking dataset extended for scene flow estimation, we have compared PillarFlowNet to multiple strong baselines and demonstrated that for simultaneous LiDAR object detection and scene flow estimation, it significantly outperforms the state-of-the-art, increasing the average precision score by 16.3 percentage points and reducing the average endpoint error by 21.4 %. Furthermore, we showed that in a single-task scene flow training setup PillarFlowNet outperforms the current state-of-the-art by a large margin in terms of endpoint error, while achieving both significantly higher inlier rates as well as significantly lower outlier rates.

Chapter 4

Multi-View RGB-D Fusion for 6D Pose Estimation

This chapter explores fusion methodologies tailored for the combination of RGB-D input data acquired from multiple different perspectives, with the primary objective of accurately and robustly estimating the 6D poses of objects within very cluttered scenes. Parts of this chapter are taken from two of my publications. The first publication is "MV6D: Multi-view 6D Pose Estimation on RGB-D Frames Using a Deep Point-wise Voting Network" [DDN22], written by Fabian Duffhauss, Tobias Demmler, and Gerhard Neumann, which has been presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2022. The second publication is "SyMFM6D: Symmetry-aware Multi-directional Fusion for Multi-View 6D Object Pose Estimation" [Duf+23], written by Fabian Duffhauss, Sebastian Koch, Hanna Ziesche, Ngo Anh Vien, and Gerhard Neumann, which has been published in the IEEE Robotics and Automation Letters (RA-L) 2023. Please refer to the second main row of tables 1.1 and 1.2 to see the fusion modalities and most important challenges addressed in this chapter in comparison to the other main chapters.

4.1 Introduction

Estimating the 6D poses of objects is an essential computer vision task which is widely used in robotics [CMS11; Xia+18; He+20; He+21], automated driving [Ku+18; XAJ18; Gu+21], augmented reality [MUS15; Su+19], human-machine interaction [Pav+17; ML20], and several other fields. 6D object pose estimation describes the prediction of the position and the orientation of objects in 3D space. This task can be very challenging, depending on manifold factors such as the objects' appearances, their geometry, the presence of neighboring objects in close proximity, or inaccurate sensor data.

In recent years, 6D pose estimators have made significant progress based on deep neural network architectures which rely on a single RGB image [Xia+18; Di+21; Su+22], on a single point cloud [HB20; HSS22] or fuse both [Wan+19; He+20; He+21]. Especially methods based on single RGB-D cameras became increasingly popular due to

technological advancements and decreasing costs of these sensors [Wad+20; Xu+20a; Wu+22]. Single-view methods, however, have problems detecting objects which are occluded by other objects. These problems can be overcome by considering data from multiple perspectives. Fusing multi-view data can significantly improve the accuracy and robustness of environmental understanding in complex scenarios, which can enable more flexible production and assembly processes, among other applications. Especially in robotic applications, a multi-view camera setup can be installed very easily, or even a setup with just a single moving camera.

There are already a few methods that consider multi-view data [Zen+17; LBH18; Lab+20] which are, however, computationally expensive and not designed for scenes with strong occlusions. Moreover, most methods suffer from symmetric objects as they have multiple 6D poses with the same visual and geometric appearance, causing most learning-based estimators to average over these multiple solutions. In the following, we present two novel multi-view RGB-D fusion approaches for 6D object pose estimation that tackle the previously stated challenges. An overview illustration of both approaches is depicted in figure 4.1.

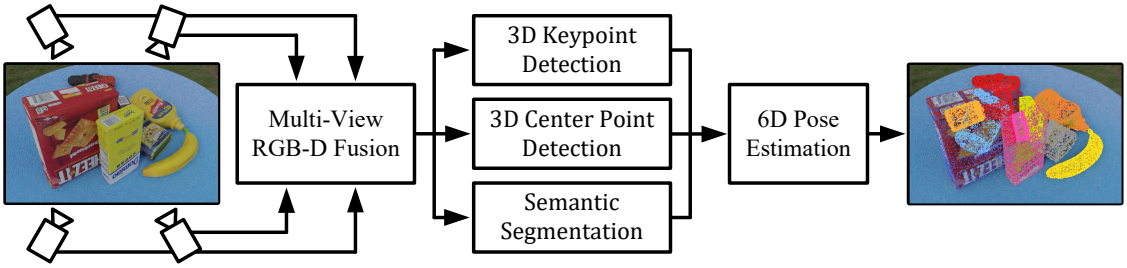


Figure 4.1: Overview of our proposed multi-view 6D object pose estimation approaches. We present two novel deep multi-view fusion networks which merge RGB-D data from multiple cameras. Both methods first predict 3D keypoints, 3D center points, and semantic labels, before estimating the 6D poses of all objects in the scene.

Our first approach, called MV6D (**M**ulti-**V**iew **6D** object pose estimation), takes multiple RGB-D images depicting a cluttered scene from different viewpoints which are ideally very distinct. Whereas the RGB images are processed individually, we fuse all depth images to a joint point cloud. Similar to PVN3D [He+20], our approach predicts pre-defined 3D keypoints for each object using independent feature encoding networks for both modalities. In a final least-squares fitting [AHB87] step, MV6D predicts the 6D poses of all objects in the scene. To the best of our knowledge, MV6D is the first approach that combines multiple RGB-D frames in a single network for the task of 6D pose estimation.

Our second approach is called SyMFM6D (**Sy**mmetry-aware **M**ulti-directional **F**usion approach for **M**ulti-view **6D** object pose estimation). It further improves the fusion of multi-view RGB-D data using a novel deep multi-directional fusion network. This network

exploits the visual information from the RGB data and geometric information from the depth data in multiple hierarchies and learns a compact representation of the entire scene. Like MV6D, SyMFM6D predicts the 6D poses of all objects in the scene simultaneously based on keypoint detection, semantic segmentation, and least-squares fitting. However, we further improve the keypoint detection by presenting a novel symmetry-aware training procedure including a novel objective function.

Since established 6D pose estimation datasets, such as YCB-Video [Xia+18], LineMOD [Hin+11b], and T-LESS [Hod+17] do not provide many frames from very distinct perspectives, we create four challenging photorealistic datasets with cluttered scenes using YCB objects [Cal+15]. These datasets provide multi-view RGB-D data and ground truth for instance semantic segmentation and 6D object pose estimation.

We conduct extensive experiments with our two novel approaches and related work on our photorealistic datasets and the YCB-Video dataset [Xia+18]. They show that our multi-view approaches outperform all single-view methods significantly. We further demonstrate that our approaches work accurately in both fixed and dynamic camera settings. Besides, our methods are robust towards inaccurate camera calibration by compensating for imprecise camera pose information when using multiple views. Moreover, our experiments demonstrate a large benefit of the proposed symmetry-aware training procedure, improving the accuracy of both symmetric and non-symmetric objects due to synergy effects. Thus, SyMFM6D outperforms the state-of-the-art in single-view and multi-view 6D pose estimation while being computationally more efficient.

Our main contributions in this chapter are:

- We propose two novel multi-view fusion networks for efficient representation learning of multiple RGB-D frames and present two novel multi-view 6D pose estimation methods based on it.
- We present a novel symmetry-aware training procedure for 3D keypoint detection based on a symmetry-aware objective function.
- We create four challenging synthetic datasets, each comprising photorealistic multi-view RGB-D data and ground truth annotations for instance semantic segmentation and 6D pose estimation.
- We perform comprehensive experiments demonstrating the superiority and limitations of our approaches on challenging real-world and synthetic datasets.
- We show significant improvements and synergy effects due to our symmetry-aware training procedure.
- We demonstrate the robustness of our approaches towards inaccurate camera calibration and variable camera arrangements.

4.2 Related Work

Over the last few years, there has been significant progress in the area of 6D pose estimation and related tasks like 3D object detection. This section presents the most important milestones subdivided into single-view methods, multi-view methods, and symmetry-aware methods. Please refer to section 2.6.7 for further details about 6D pose estimation fundamentals including related work.

4.2.1 Single-View 6D Pose Estimation

Single-view 6D Pose estimation methods require only a single input modality, which can be a single RGB image, a single RGB-D image, or a single point cloud.

Pose Estimation on Single RGB Images

Traditional pose estimation methods using a *single RGB image* [Low99; Low04; RD06; BTV06; Rot+06; CMS11; Col+09; Pen+19] extract local features from the given RGB image and match them to the corresponding features in its 3D model. Based on the 2D-3D-correspondences, a Perspective-n-Point (PnP) algorithm [FB81] can be applied to estimate the object's pose. Even though feature-based methods can handle occlusions up to a certain degree, the detection of 2D keypoints does not work well on objects without a distinctive texture [Hod+17].

In contrast, methods based on template-matching [GR10; Hin+11a; CSB16] can cope with textureless objects. Templates of an object can be generated by rendering its corresponding 3D model from diverse views. Finding a match between a rendered template and a part of the input image provides the 6D pose of the corresponding object. However, template-based methods suffer from occlusions as the matching becomes inaccurate.

There are also end-to-end trainable neural networks [TSF18; GPH19; Wan+21d; Di+21; Su+22] which directly predict the objects' poses based on a single RGB image without requiring multiple stages. These methods share similar ideas to exploit differentiable PnP or differentiable rendering techniques. However, often the generalization of direct methods is an issue due to the non-linearity of the rotation space [Pen+19]. [TM15; Su+15b; Sun+18] overcome this issue by discretizing the rotation space. Another common procedure is to refine the predicted poses, for example by applying the iterative closest point (ICP) algorithm [BM92] using additional depth data as in PoseCNN [Xia+18] or SSD-6D [Keh+17]. Alternatively, deep learning-based pose refining networks like DeepIM [Li+18] or DPOD [ZSI19] are proposed for faster and more accurate refinement without the requirement for depth data.

Pose Estimation on Single Point Clouds

Recently, due to the rapid technological progress of depth and LiDAR sensors, many pose estimation methods were developed based on a single depth image or a single point cloud [Che+20a; Fer+21]. In this area, there are methods [SX14; Li17] that directly predict oriented 3D bounding boxes using 3D convolutions. However, 3D convolutions are computationally expensive which leads to high inference times. To reduce the computational complexity, it is common even in modern approaches [ZT18; YML18; Lan+19] to apply feature encoding networks based on PointNet [Qi+17a] or PointNet++ [Qi+17b] which are able to extract geometric features. To do that, the point cloud is either divided into voxels [ZT18; YML18] or into vertical pillars [Lan+19].

Recently, Qi et al. [Qi+19] introduced deep Hough voting for end-to-end 3D object detection. Their network VoteNet generates votes to object centers which are fused to obtain object proposals. Building upon that, Xie et al. [Xie+21] further improved the feature encoding of seed points by an attentive multi-layer perceptron, a vote attraction loss, and vote weighting.

However, as methods based on depth images or point clouds cannot exploit texture, their performance is limited to applications where textures are not relevant.

Pose Estimation on Single RGB-D Images

RGB-D-based approaches try to combine the advantage of both color and depth modalities. AVOD [Ku+18] and MV3D [Che+17c] use convolutional feature extractor networks followed by a 3D object proposal network for fusing RGB images and LiDAR point clouds. The latter is compressed into a bird's eye view and in the case of MV3D, an additional front view projection is used. However, these approaches are based on the assumption that all objects of interest are located on a plane.

PointFusion [XAJ18] proposes the usage of PointNet [Qi+17a] for point cloud feature extraction and introduces a dense fusion module for combining point cloud features and RGB features which were created by a CNN. DenseFusion [Wan+19] transfers that concept from 3D object detection in autonomous driving to 6D pose estimation for robotics and introduces a neural network for iterative pose refinement.

PVN3D [He+20] further improves DenseFusion [Wan+19] by applying PointNet++ [Qi+17b] and by introducing a deep Hough voting network for 3D keypoint detection. The 6D poses are estimated by a least-squares fitting algorithm [AHB87]. FFB6D [He+21] enhances PVN3D [He+20] with a bidirectional fusion module that combines the features representing texture and geometric information in each encoding and decoding layer. Furthermore, they replaced PointNet++ [Qi+17b] with a RandLA-Net [Hu+20] for point cloud feature encoding. However, most previous methods including [Wan+19; He+20; He+21] do not explicitly consider object symmetries and suffer from strong occlusions.

4.2.2 Multi-View 6D Pose Estimation

Multi-view pose estimators consider multiple RGB(-D) frames showing the same scene from different perspectives in order to reduce the effect of occlusions and to improve the 6D pose estimation accuracy.

Zeng et al. [Zen+17] present a 6D pose estimation approach based on 15 to 18 RGB-D images that are recorded by a robot arm from very similar perspectives. They use a fully convolutional neural network to perform 2D object segmentation on each RGB image individually. Afterwards, the segmentation results are fused with the depth images into a single segmented point cloud. The 6D poses are estimated using an ICP algorithm [BM92].

Sock et al. [Soc+17] propose an active multi-view framework with next-best-view prediction and hypothesis accumulation. Based on previous single-shot pose hypotheses they predict the next best camera perspectives and select the most likely object poses.

Li et al. [LBH18] introduce an end-to-end trainable CNN-based architecture integrating known class labels into the learning process of convolutional filters for single-view 6D pose estimation. Based on an arbitrary object detector providing regions of interest in RGB or RGB-D data, they perform the single-view pose estimation multiple times with images from different viewpoints before selecting the best hypothesis using a voting score that suppresses outliers.

Recently, Labbé et al. [Lab+20] have presented a three-stage approach for 6D pose estimation based on multiple RGB images. Their method CosyPose employs DeepIM [Li+18] to generate object candidate proposals for each view independently. Secondly, they conduct a candidate matching considering the predictions of all views which belong to the same object instance. Finally, CosyPose performs a refinement procedure based on object-level bundle adjustment [Tri+00]. However, the approach fails if an object is detected in just a single view.

All previously named multi-view pose estimation methods apply deep neural networks independently on each view which leads to high computational effort due to redundancy and sub-optimal use of information as there is no prediction that can use all information. To the best of our knowledge, our approaches MV6D and SyMFM6D are the first approaches that directly fuse the features from multiple RGB-D views before performing the pose estimation based on that.

4.2.3 Symmetry-aware 6D Pose Estimation

Symmetric objects are known to be a challenge for 6D pose estimation approaches due to ambiguities [Pit+19]. Different techniques have been proposed to address this issue. The authors of [Pit+19] and [RL17] propose to utilize an additional output channel to classify the type of symmetry and its domain range. In [PPV19], a loss is introduced that is the smallest error among symmetric pose proposals in a finite pool of symmetric poses.

In [HBM20] the authors propose to use compact surface fragments as a compositional way to represent objects. As a result, this representation can easily allow the handling of symmetries. The authors of [Zha+20b] employ an additional symmetry prediction as output, and an extra refining step of predicted symmetry via an optimization function. A novel output space representation for CNNs is presented in [RF21] where symmetrical equivalent poses are mapped to the same values. In [Mo+22] the authors introduce a compact shape representation based on grouped primitives to handle symmetries. However, none of these methods outperforms the keypoint-based methods PVN3D [He+20] and FFB6D [He+21], even though they do not explicitly consider object symmetries. In contrast, our SyMFM6D method extends current keypoint-based methods to consider object symmetries and consequently outperforms all previous methods on both single-view and multi-view 6D object pose estimation.

4.3 6D Pose Estimation Problem Definition

6D object pose estimation describes the task of predicting a rigid transformation $\boldsymbol{p} = [\boldsymbol{R}|\boldsymbol{t}] \in SE(3)$ which transforms the coordinates of an observed object from the object coordinate system into the camera coordinate system. This transformation is called a 6D object pose because it is composed of a 3D rotation $\boldsymbol{R} \in SO(3)$ and a 3D translation $\boldsymbol{t} \in \mathbb{R}^3$. The designated aim of our approach is to jointly estimate the 6D poses of all objects in a given cluttered scene using multiple RGB-D images which depict the scene from multiple perspectives. We assume that the 3D models of the objects and the camera poses are known and not more than a single instance per object class occurs in each scene.

4.4 Dense Multi-View Fusion Method

This section presents MV6D, the first of the two proposed deep learning approaches for multi-view 6D pose estimation based on RGB-D data. Figure 4.2 illustrates the network architecture of MV6D which is composed of three stages that are inspired by the single-view network PVN3D [He+20]. The first stage accepts a variable number of RGB-D frames, extracts relevant features, and fuses them to a joint feature representation of the entire input scene. The second stage contains network heads for 3D keypoint detection, 3D center point detection, and instance semantic segmentation. The third stage estimates the 6D poses of all objects in the scene in a least-squares fitting manner. While PVN3D [He+20] uses only single-view input data, we propose a new mechanism to fuse the depth data as well as the RGB data of several RGB-D views into a single consistent feature representation. To accomplish that, we employ a modified DenseFusion module [Wan+19] which is illustrated in figure 4.3 and elucidated in the following section.

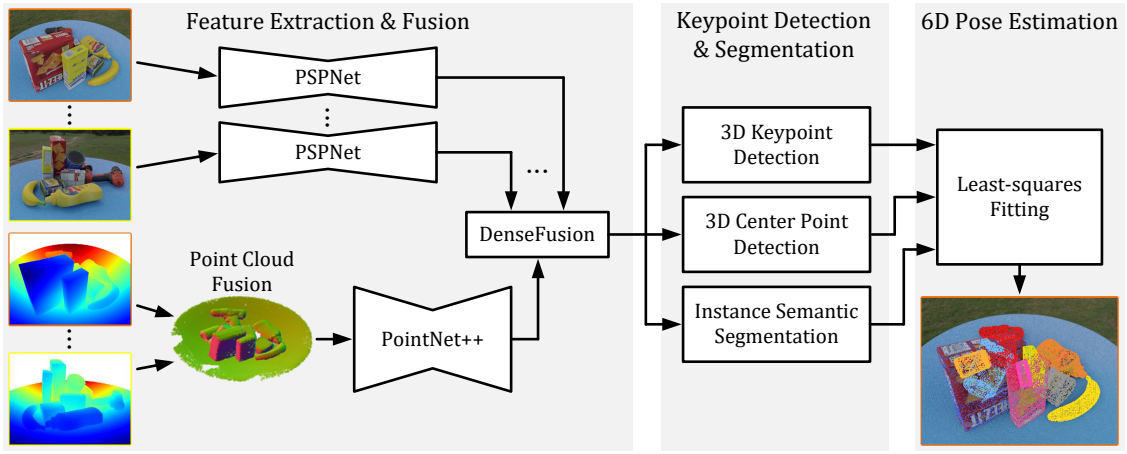


Figure 4.2: Architecture of the proposed MV6D network. Given multiple RGB-D images, MV6D extracts visual features from the RGB images and geometric features from a point cloud which is created by fusing all depth images. A modified DenseFusion network [Wan+19] fuses visual and geometric features. Based on a 3D keypoint detection, a 3D center point detection, and an instance semantic segmentation module, we predict 6D poses using least-squares fitting [AHB87].

4.4.1 Multi-view Fusion Architecture

This section describes the first stage of our proposed MV6D network, depicted in figure 4.2, which is responsible for feature extraction and multi-view RGB-D data fusion.

Geometric Feature Extraction and Fusion

To extract geometric features from the depth images, we first convert all depth images into point clouds and combine them into a single point cloud using the known camera poses. We use the camera coordinate system of the first camera as a reference coordinate system for the resulting point cloud and all further keypoint predictions. As previous methods [He+21; He+20], we further process only a subset of points selected by random sampling and attach the corresponding RGB value as well as the surface normal to each remaining point. For feature extraction, we apply a PointNet++ [Qi+17b] with multi-scale grouping.

Visual Feature Extraction and Fusion

For each RGB image, we independently extract pixel-wise visual features using modified PSPNets [Zha+17b] which contain a ResNet-34 [He+16] pre-trained on ImageNet [Den+09] as the backbone. All PSPNets share the same parameters. Each point in the processed point cloud is associated with a corresponding pixel of the RGB image from the view that generated the point. Similar to a DenseFusion network [Wan+19], we

concatenate to each point in the point cloud the PSPNet feature vector of the associated pixel from the associated view as shown in figure 4.3. Hence, even if points are spatially close to each other, they can obtain the visual features from different RGB images if they have been generated from different views. Subsequently, we compute a global feature vector by an MLP followed by an average pooling layer that aggregates the information from the whole point cloud. This feature vector is then again appended to the geometric and RGB features of each point in the point cloud. Through extensive training of the entire network, the resulting point-wise feature vectors can contain the most relevant information of the input data and thus represent the entire scene in a compact tensor.

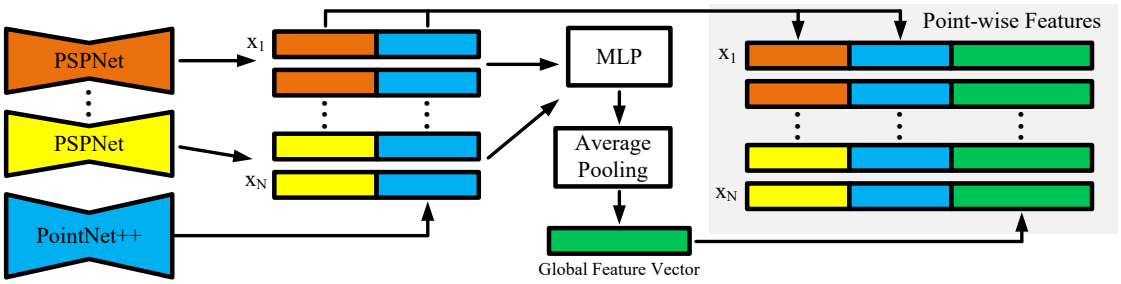


Figure 4.3: DenseFusion module of our MV6D network architecture. While keeping a mapping between the visual features of the PSPNets [Zha+17b] and the corresponding geometric features of the PointNet++ [Qi+17b], we compute a global feature vector and concatenate the results to obtain point-wise feature vectors.

4.4.2 Modules for Segmentation and Keypoint Detection

The output tensor of the feature extraction and fusion network is used as input for the instance semantic segmentation module, the 3D center point detection module, and the 3D keypoint detection module, which are presented in the following.

Instance Semantic Segmentation Module and 3D Center Point Detection Module

The instance semantic segmentation module in figure 4.2 consists of a semantic segmentation module and a center offset module as in PVN3D [He+20]. Both submodules take the point-wise feature vectors (consisting of visual, geometric, and global features) and process them with shared MLPs. The semantic segmentation module predicts an object class for each point. The center offset module estimates the translation offset from the given point to the center of the object that it belongs to. Following PVN3D [He+20], we apply mean shift clustering [Che95] to obtain the final object center predictions. These are then used to further refine the segmentation map by rejecting points which are too far away from the object center.

3D Keypoint Detection Module

In advance of the training, we select eight target keypoints from the mesh of each object using the farthest point sampling (FPS) algorithm [Eld+97] as in [He+20]. Using a shared MLP, we predict the translation offset from each point to each target keypoint of the class that the object belongs to. Adding the predicted offsets to the corresponding points from the point cloud results in a set of keypoint predictions. All keypoint predictions belonging to an instance are clustered by mean shift clustering [Che95] in order to obtain the final 3D keypoint predictions as in [He+20].

4.4.3 Multi-Task Objective Function

We train our MV6D network as in PVN3D [He+20] by minimizing the multi-task loss function

$$L_{\text{multi-task}} = \lambda_1 L_{\text{keypoints}} + \lambda_2 L_{\text{semantic}} + \lambda_3 L_{\text{center}}, \tag{4.1}$$

where the 3D keypoints detection loss $L_{\text{keypoints}}$ and the center voting loss L_{center} are L1 losses. The loss function for the instance semantic segmentation L_{semantic} is a Focal loss [Lin+17b]. $\lambda_1 = 2$, $\lambda_2 = 1$, and $\lambda_3 = 1$ are the weights for the individual loss functions as in PVN3D [He+20].

6D Pose Computation via Least-squares Fitting

Based on the target keypoints and the corresponding keypoint predictions, we use a least-squares fitting algorithm [AHB87] to compute the rotation $\boldsymbol{R}$ and the translation $\boldsymbol{t}$ of each object following PVN3D [He+20]. The employed least-squares fitting algorithm minimizes the squared loss

$$L_{\text{Least-squares}} = \sum_{i=1}^{M} \left\| \hat{\boldsymbol{k}}_i - (\boldsymbol{R}\boldsymbol{k}_i + \boldsymbol{t}) \right\|^2, \tag{4.2}$$

where $M = 8$ is the number of keypoints per object, $\boldsymbol{k}_i$ are the target keypoints in the object coordinate system, and $\hat{\boldsymbol{k}}_i$ are the predicted keypoints in the camera coordinate system.

4.5 Symmetry-aware Multi-View Fusion Method

The previously presented MV6D network contains a computationally expensive feature extraction and fusion network which does not explicitly consider object symmetries. To overcome these two drawbacks, we propose a second multi-view 6D pose estimation method called SyMFM6D. SyMFM6D incorporates a novel deep multi-directional fusion

approach for combining multi-view RGB-D data more effectively and more efficiently. We further introduce a novel training procedure which explicitly considers ambiguities due to object symmetries.

4.5.1 Network Overview

Our symmetry-aware multi-view network consists of three stages which are visualized in figure 4.4. The first stage receives one or multiple RGB-D images and extracts visual features as well as geometric features which are fused to a joint representation of the scene. The second stage performs an instance semantic segmentation and detects predefined 3D keypoints as well as 3D center points. Based on the keypoints and the information to which object the keypoints belong, we compute the 6D object poses with a least-squares fitting algorithm [AHB87] in the third stage.

4.5.2 Multi-View Feature Extraction

To efficiently predict 3D keypoints, 3D center points, and semantic labels, the first stage of our approach learns a compact representation of the given scene by extracting and merging features from all available RGB-D images in a deep multi-directional fusion manner. For that, we first separate the set of RGB images RGB_1, ..., RGB_N from their corresponding depth images Dpt_1, ..., Dpt_N. The N depth images are converted into point clouds, transformed into the coordinate system of the first camera, and merged to a single point cloud using the known camera poses as in MV6D [DDN22]. Unlike [DDN22], we employ a point cloud network based on RandLA-Net [Hu+20] with an encoder-decoder architecture using skip connections. The point cloud network learns geometric features from the fused point cloud and considers visual features from the multi-directional point-to-pixel fusion modules as described in section 4.5.3.

The N RGB images are independently processed by a CNN with encoder-decoder architecture using the same weights for all N views. The CNN learns visual features while considering geometric features from the multi-directional pixel-to-point fusion modules. We build the encoder upon a ResNet-34 [He+16] pretrained on ImageNet [Den+09] and the decoder upon a PSPNet [Zha+17b] similar to FFB6D [He+21].

After the encoding and decoding procedures including several multi-view feature fusions, we collect the visual features from each view corresponding to the final geometric feature map and concatenate them. The output is a compact feature tensor containing the relevant information about the entire scene which is used for 3D keypoint detection, 3D center point detection, and instance semantic segmentation as in MV6D (see section 4.4.2).

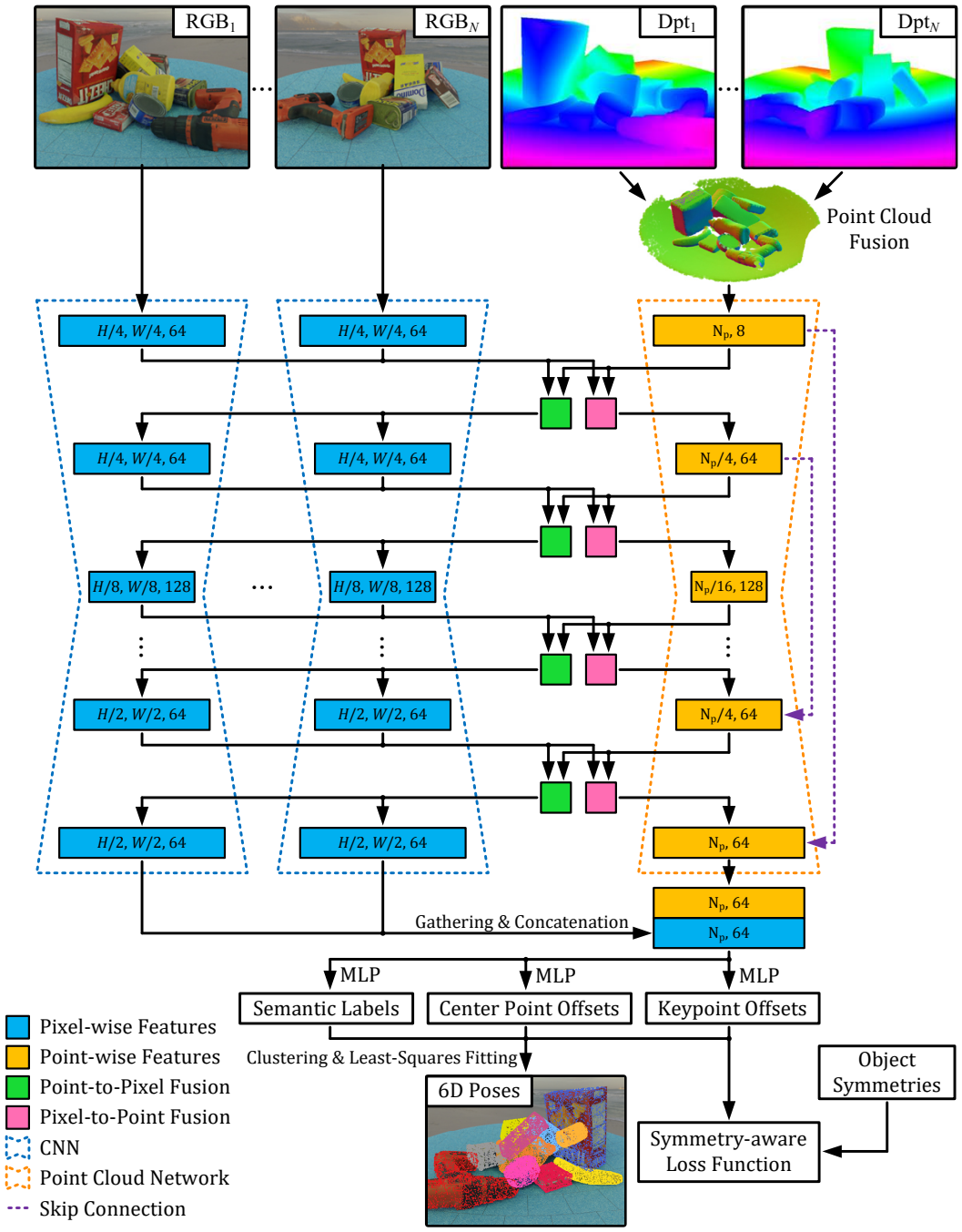


Figure 4.4: Network architecture of our SyMFM6D method. An encoder-decoder CNN processes the N RGB images with height H and width W. The N depth images are converted into a single point cloud with N_p points which is processed by an encoder-decoder point cloud network. Every hierarchy contains two multi-directional fusion modules. We utilize three shared MLPs to regress 3D keypoint offsets, 3D center point offsets, and semantic labels. The 6D poses are computed based on mean shift clustering and least-squares fitting. We train our network by minimizing our proposed symmetry-aware multi-task loss function using pre-computed object symmetries.

4.5.3 Multi-View Feature Fusion

In order to efficiently fuse the visual and geometric features from multiple views, we extend the fusion modules of FFB6D [He+21] from bi-directional fusion to *multi-directional fusion*. We present two types of multi-directional fusion modules which are illustrated in figure 4.5. Both types of fusion modules take the pixel-wise visual feature maps and the point-wise geometric feature maps from each view, combine them, and compute a new feature map. This process requires a correspondence between pixel-wise and point-wise features which we obtain by computing an XYZ map for each RGB feature map based on the depth data of each pixel using the camera intrinsic matrix as in FFB6D [He+21]. To deal with the changing dimensions at different layers, we use the centers of the convolutional kernels as new coordinates of the feature maps and resize the XYZ map to the same size using nearest interpolation as proposed in [He+21].

The *point-to-pixel* fusion module in figure 4.5a computes a fused feature map $\boldsymbol{F}_{\mathrm{f}}$ based on the image features $\boldsymbol{F}_{\mathrm{i}}(v)$ of all views $v \in \{1, \ldots, N\}$. We collect the K_{p} nearest point features $\boldsymbol{F}_{\mathrm{p}_k}(v)$ with $k \in \{1, \ldots, K_{\mathrm{p}}\}$ from the point cloud for each pixel-wise feature and each view independently by computing the nearest neighbors according to the Euclidean distance in the XYZ map. Subsequently, we process them by a shared MLP before aggregating them by max pooling, i.e.,

$$\widetilde{\boldsymbol{F}}_{\mathrm{p}}(v) = \max_{k \in \{1,\ldots,K_{\mathrm{p}}\}} \Big(\mathrm{MLP}_{\mathrm{p}}(\boldsymbol{F}_{\mathrm{p}_k}(v)) \Big). \tag{4.3}$$

Finally, we apply a second shared MLP to fuse all features $\boldsymbol{F}_{\mathrm{i}}$ and $\widetilde{\boldsymbol{F}}_{\mathrm{p}}$ as $\boldsymbol{F}_{\mathrm{f}} = \mathrm{MLP}_{\mathrm{fp}}(\widetilde{\boldsymbol{F}}_{\mathrm{p}} \oplus \boldsymbol{F}_{\mathrm{i}})$ where $\oplus$ denotes the concatenate operation.

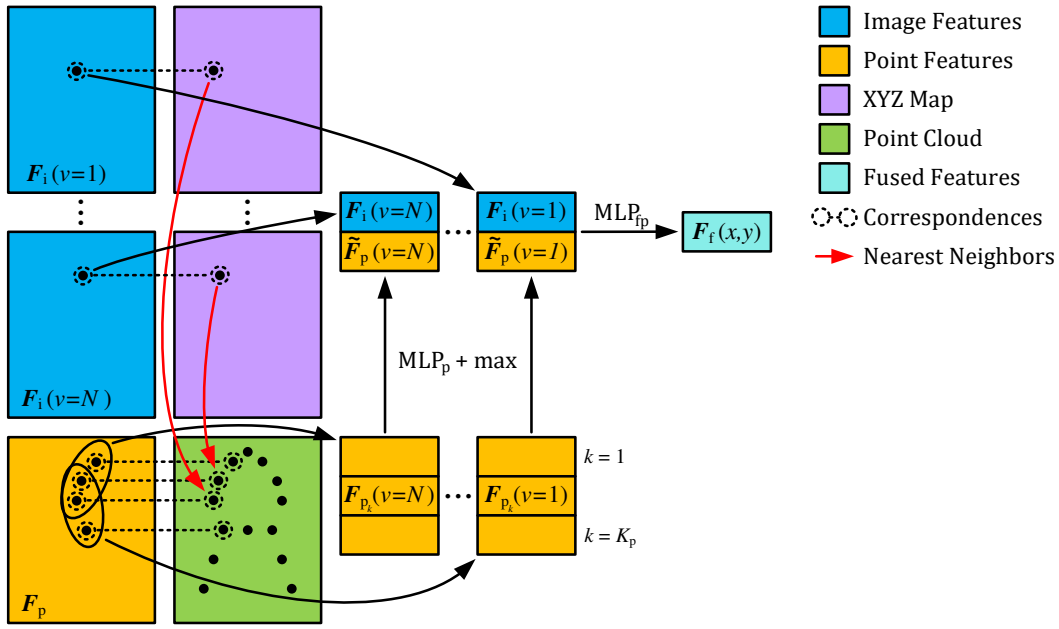
The *pixel-to-point* fusion module in figure 4.5b collects the K_{i} nearest image features $\boldsymbol{F}_{\mathrm{i}_k}(\mathrm{i2v}(i_k))$ with $k \in \{1, ..., K_{\mathrm{i}}\}$. $\mathrm{i2v}(i_k)$ is a mapping that maps the index of an image feature to its corresponding view. This procedure is performed for each point feature vector $\boldsymbol{F}_{\mathrm{p}}(n)$. We aggregate the collected image features by max pooling and apply a shared MLP, i.e.,

$$\widetilde{\boldsymbol{F}}_{\mathrm{i}} = \mathrm{MLP}_{\mathrm{i}} \left(\max_{k \in \{1,\ldots,K_{\mathrm{i}}\}} \Big(\boldsymbol{F}_{\mathrm{i}_k}(\mathrm{i2v}(i_k)) \Big) \right). \tag{4.4}$$

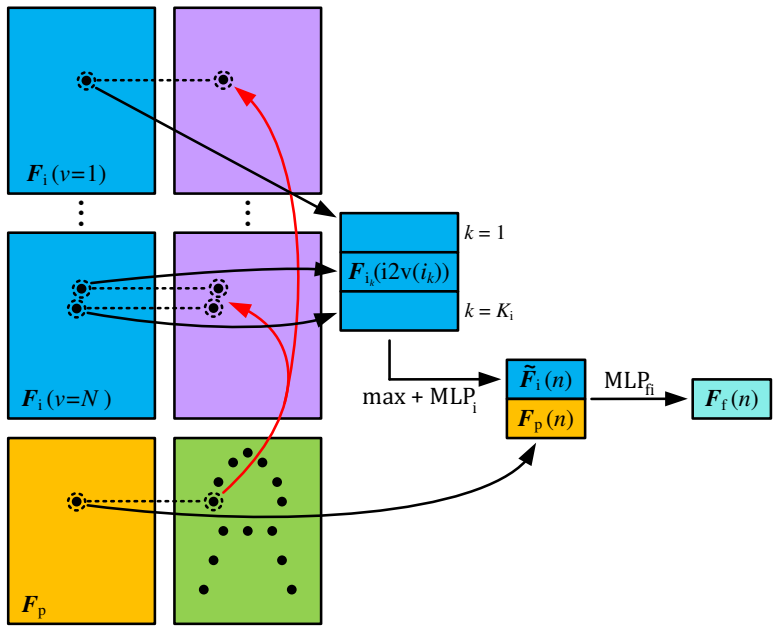
One more shared MLP fuses the resulting image features $\widetilde{\boldsymbol{F}}_{\mathrm{i}}$ with the point features $\boldsymbol{F}_{\mathrm{p}}$ as $\boldsymbol{F}_{\mathrm{f}} = \mathrm{MLP}_{\mathrm{fi}}(\widetilde{\boldsymbol{F}}_{\mathrm{i}} \oplus \boldsymbol{F}_{\mathrm{p}})$.

4.5.4 3D Keypoint Detection and Segmentation

The second stage of our SyMFM6D network contains modules for 3D keypoint detection, 3D center point detection, and instance semantic segmentation as in MV6D. However, unlike MV6D, we use the Scale Invariant Feature Transform Farthest Point Sampling



(a) Point-to-pixel fusion module.



(b) Pixel-to-point fusion module.

Figure 4.5: Overview of our proposed multi-directional multi-view fusion modules. They combine pixel-wise visual features and point-wise geometric features by exploiting the correspondence between pixels and points using the nearest neighbor algorithm. We compute the resulting features using multiple shared MLPs and max pooling. For simplification, we depict an example with $N = 2$ views and $K_i = K_p = 3$ nearest neighbors. The points of ellipsis (...) illustrate the generalization for an arbitrary number of views N.

(SIFT-FPS) algorithm [Low99] as introduced in FFB6D [He+21] to define eight target keypoints for each object class. SIFT-FPS yields keypoints with salient features which are easier to detect.

4.5.5 6D Pose Computation via Least-Squares Fitting

As in MV6D, we follow PVN3D [He+20] and compute the 6D poses of all objects in the scene based on the estimated 3D keypoints using the least-squares fitting algorithm [AHB87] in equation (4.2).

4.5.6 Symmetry-aware Keypoint Detection

Most related approaches, including PVN3D [He+20], FFB6D [He+21], and our MV6D network, do not specifically consider object symmetries. However, symmetries lead to ambiguities in the predicted keypoints as multiple 6D poses can have the same visual and geometric appearance. Therefore, we introduce a novel symmetry-aware training procedure for the 3D keypoint detection including a novel symmetry-aware objective function to make the network predicting either the original set of target keypoints for an object or a rotated version of the set corresponding to one object symmetry. Either way, we can still apply the least-squares fitting algorithm which efficiently computes an estimate of the target 6D pose or a rotated version corresponding to an object symmetry.

To enable this symmetry-aware training procedure, knowledge about the rotational symmetry axes of all objects is required. Reflectional symmetries which can be represented as rotational symmetries are handled as rotational symmetries. Other reflectional symmetries are ignored since the reflection cannot be expressed as an Euclidean transformation and thus does not result in an existing object. Objects with a continuous rotational symmetry have an infinite number of rotational symmetry transformations. To nevertheless enable efficient training, we discretize the continuous rotational symmetries into 16 discrete rotational symmetry transformations which we found to be a good compromise between computational effort and accuracy.

We propose to compute the set $\boldsymbol{S}_I$ of all rotational symmetric transformations for the given object instance I with a stochastic gradient descent algorithm [LH17]. Given the known mesh of an object and an initial estimate for the symmetry axis, we transform the object mesh along the symmetry axis estimate and optimize the symmetry axis iteratively by minimizing the ADD-S metric [Hin+12] (see section 4.7.1) as an objective function which penalizes the difference between the original object mesh and the transformed one. For objects with multiple symmetry axes, we find all symmetry axes by applying this optimization procedure multiple times with different initial values. Please note that this process of computing all symmetry transformations needs to be done only once for all required objects. This can be accomplished in advance of the training so that it does not increase training or inference time.

We extend the keypoints loss function of PVN3D [He+20] to become symmetry-aware such that it predicts the keypoints of the closest symmetric transformation, i.e.

$$L_{\text{keypoints}}(\mathcal{I}) = \frac{1}{N_I} \min_{\boldsymbol{S} \in \boldsymbol{S}_I} \sum_{i \in \mathcal{I}} \sum_{j=1}^{M} \left\| \boldsymbol{x}_{ij} - \boldsymbol{S}\hat{\boldsymbol{x}}_{ij} \right\|, \tag{4.5}$$

where N_I is the number of points in the point cloud for object instance I, M is the number of target keypoints per object, and $\mathcal{I}$ is the set of all point indices that belong to the object instance I. The vector $\hat{\boldsymbol{x}}_{ij}$ is the predicted keypoint offset for the i-th point and the j-th keypoint while $\boldsymbol{x}_{ij}$ is the corresponding ground truth.

4.5.7 Multi-Task Objective Function

We train our network by minimizing a multi-task loss function with the same basic structure as in MV6D (see equation (4.1) in section 4.4.3). However, for predicting the 3D keypoints, we instead employ the proposed symmetry-aware loss function in equation (4.5).

4.6 Experimental Setup

To evaluate the performance of our two proposed methods MV6D and SyMFM6D in comparison to related approaches, we conduct extensive experiments on five very challenging datasets.

4.6.1 Datasets

The first part of this section provides an overview of commonly used datasets for 6D object pose estimation. Afterwards, we present novel photorealistic synthetic datasets specifically designed for multi-view 6D pose estimation in very cluttered scenes.

YCB-Video

The YCB-Video dataset [Xia+18] contains a total of 133,827 RGB-D images with a resolution of 640 × 480 pixels showing static cluttered object scenes. The scenes are composed of three to nine objects from the 21 Yale-CMU-Berkeley (YCB) object set [Cal+15]. The RGB-D frames originate from 92 videos that were recorded by hand, each showing another scene from different perspectives. For training, Xiang et al. [Xia+18] have additionally created 80,000 synthetic non-sequential RGB-D frames showing a random subset of the YCB objects placed at random positions.

However, most frames from the YCB-Video dataset are very similar because they originate from videos with 30 frames per second recorded by a handheld camera that was

moved slowly. The videos also do not show the scene from all sides but just from similar perspectives. Furthermore, the scenes do not include strong occlusions, and hence, most object poses are simple to estimate from a single perspective.

T-LESS

The T-LESS dataset [Hod+17] is composed of 30 different industry-relevant objects that have no significant texture. There are 39k RGB-D images for training showing a single object from different perspectives. For evaluation, there are 10k test images depicting 20 different cluttered scenes with different complexity and multiple instances of certain object classes in some cases. The test images show each scene from all sides. However, since the training images show only single objects on a black background they are not sufficient to train a multi-view network on them.

LineMOD and Occlusion LineMOD

LineMOD [Hin+11b] is composed of 15,783 RGB-D images, showing a subset of 15 objects in cluttered scenes. There are 13 sequences and in each sequence, a single object class is annotated. Occlusion LineMOD [Bra+14] corresponds to one sequence of LineMOD showing significant occlusions among eight objects. The sequence contains 1,214 RGB-D images and annotations for all objects. It is common practice to use Occlusion LineMOD as a test set for approaches that are trained on LineMOD [Bra+14; He+20; He+21]. However, like YCB-Video [Xia+18], LineMOD [Hin+11b] does not show the scenes from all sides but only from similar perspectives.

Novel Photorealistic Datasets

Due to the drawbacks of the previously mentioned datasets, we create four novel photorealistic datasets of diverse cluttered scenes with heavy occlusions. All of these datasets contain RGB-D images from multiple very distinct perspectives which are annotated with 6D poses of all cameras and objects as well as ground truth for instance semantic segmentation.

Our datasets are composed of different numbers of objects from the YCB object set [Cal+15]. Many of these objects have a symmetric shape but are non-symmetric due to their texture. This requires the pose estimation approaches to exploit RGB information in order to correctly predict the objects' poses. Using Blender [Com18] with physics, we created cluttered scenes by spawning the YCB objects above the ground in the center with a 3D normally distributed offset. Due to the similar spawning point of all objects, it is likely that objects rest on top of each other. Instead of a flat ground plane, we use a shallow bowl for the objects to fall onto. The bowl prevents the objects from scattering in

all directions. Consequently, the objects stay close together resulting in heavily occluded scenes.

In addition to the random composition of the scenes, we apply further domain randomization techniques in order to improve the generalization of our models. For each scene, we selected a random 360×180 degree panorama photo from a set of 379 photos and a different bowl texture. Furthermore, we slightly vary the intensity and the color of the lighting to create different shadows and reflections.

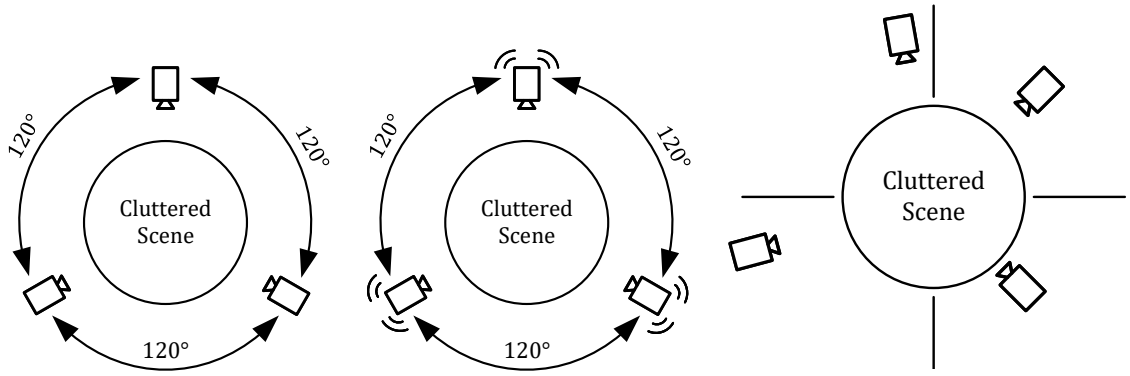
For our four datasets, we have photorealistically rendered different sets of RGB-D images with the ray-trace based rendering engine Cycles [SH14] in Blender. Since we have used YCB objects and generated data from multiple views with different camera settings, we call our dataset framework Multi-View YCB (MV-YCB). Furthermore, we have extracted the exact camera poses, and generated ground truth annotations for instance semantic segmentation and 6D pose estimation in Blender. All datasets are split into 90 % training data and 10 % test data.

For the first dataset, called **MV-YCB FixCam**, we have generated 8,333 random scenes composed of eleven non-symmetric YCB objects. We have placed three identical cameras at fixed positions equally distributed around the scene, i.e. in a circle in 120-degree intervals. This resulted in a total of 24,999 RGB-D frames with corresponding ground truth data.

For the second dataset, called **MV-YCB WiggleCam**, we have reused the same scenes and cameras as in MV-YCB FixCam but added a normally distributed 3D offset to each camera independently. This reflects a typical robotic setting where the cameras are mounted slightly inaccurately around the scene. Thus, experiments with this dataset can provide insights into how models deal with inaccurate camera calibration.

For the third dataset, called **MV-YCB MovingCam**, we have generated another 8,333 random scenes based on the same eleven non-symmetric YCB objects. Unlike in FixCam and WiggleCam, we placed four cameras at changing positions around the scene, where each camera spawns in another quadrant in a sphere. This represents a scenario, where a single camera is moved around the scene recording four frames from four very distinct perspectives. This results in a total of 33,332 RGB-D frames with corresponding ground truth data.

Since FixCam, WiggleCam, and MovingCam contain only non-symmetric objects, we have created an additional photorealistic dataset with symmetric and non-symmetric objects called **MV-YCB SymMovCam** in order to examine the effect of the proposed symmetry-aware training procedure. The SymMovCam dataset also depicts 8,333 cluttered scenes, but they are composed of 8 – 16 objects randomly chosen from the set of 21 objects which occur in the YCB-Video dataset. The increased number of objects in SymMovCam results in larger occlusions. As in MovingCam, we utilize four cameras per scene at changing positions around the scene with the restriction that in each quadrant there is only one camera so that the perspectives are very distinct.



(a) FixCam (b) WiggleCam (c) MovingCam and SymMovCam

Figure 4.6: Illustration of the camera setups employed in our MV-YCB dataset series. The FixCam dataset (a) has fixed camera positions equally distributed around the scene. The WiggleCam dataset (b) uses the same camera poses with a normally distributed offset regarding its 3D position. The datasets MovingCam and SymMovCam (c) share the same camera setup with four cameras, one in each quadrant.

Figure 4.6 provides an overview of the different setups of our generated MV-YCB datasets. Figure 4.7 present example views for each of these datasets.

4.6.2 Data Augmentation

We employ a variety of data augmentation methods to improve the generalization of our methods. During training, we follow He et al. [He+20; He+21] and apply a random combination of color jitter, sharpening filters, motion blur, Gaussian blur, and Gaussian noise on the RGB images. The depth maps and the corresponding point clouds are not augmented.

4.6.3 Training Procedure

For training our models in single-view mode on YCB-Video, we randomly use the synthetic and real images of YCB-Video with a ratio of 4:1. Since consecutive real frames are very similar, we consider only every seventh real frame. For training a multi-view model, we start from the corresponding single-view checkpoint and continue training with batches of real YCB-Video frames.

For the datasets MV-YCB FixCam and MV-YCB WiggleCam, we train with all three camera views and increase the number of training samples by using all relevant camera combinations i.e. {[cam1, cam2, cam3], [cam2, cam3, cam1], [cam3, cam1, cam2]}. We do not need every possible permutation of this list since the exact order is irrelevant. Only

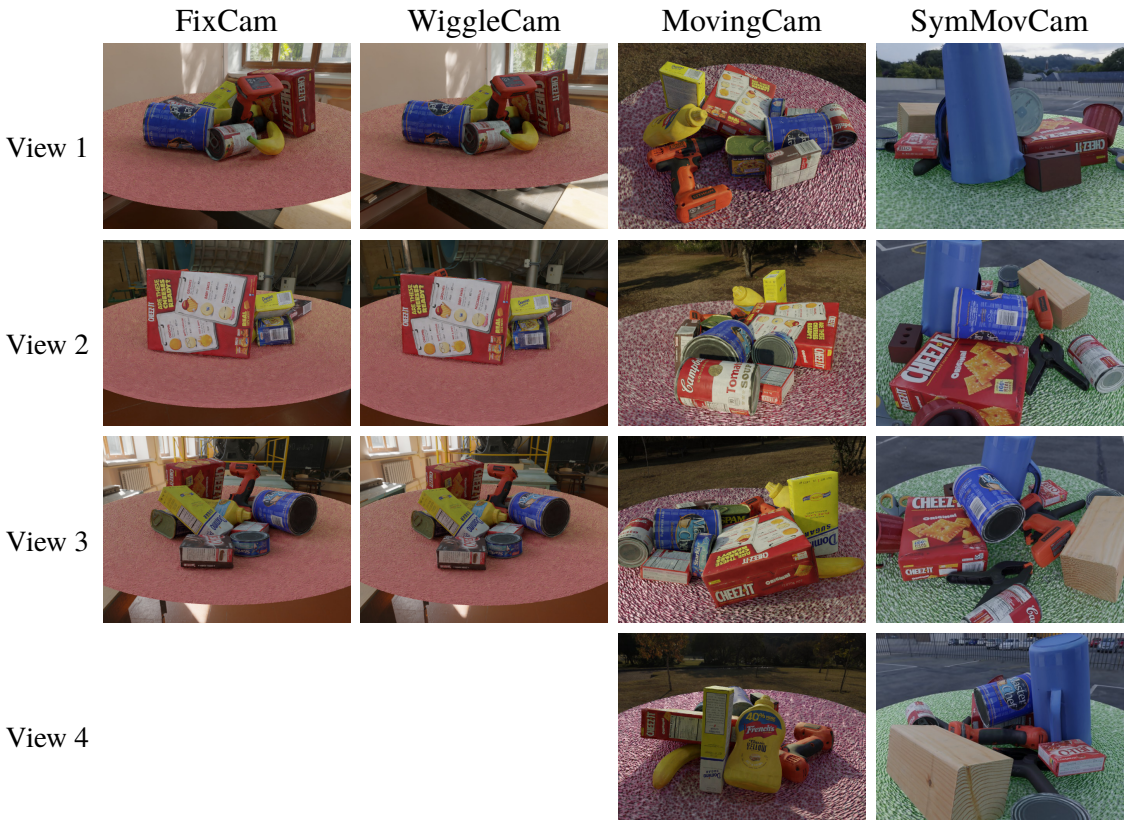


Figure 4.7: Example scenes of our four datasets MV-YCB FixCam, WiggleCam, MovingCam, and SymMovCam which have three or four views. FixCam and WiggleCam show the same scene, but the views are slightly different due to the random camera position offset. MovingCam and SymMovCam use the same camera setup, but SymMovCam has more complex scenes with more objects including symmetric and non-symmetric objects.

the first camera in the list determines the camera coordinate system in which the 6D poses are predicted.

For the datasets MV-YCB MovingCam and MV-YCB SymMovCam, we use a variable number of camera views. This allows for more flexibility when deploying the trained network as the network learns how to deal with a different number of views. We employ a set with all relevant camera combinations for each view count, e.g. {[cam1, cam2], [cam1, cam3], [cam2, cam3]} for a view count of two if we use a maximum of three cameras. The previous rotation step is then applied to each item.

Each sample in a batch must have the same view count. When training with a variable view count this is ensured by using a custom batch sampler. An equal amount of batches is sampled for each view count. A sample with a lower view count requires less memory and offers less information for network optimization. Therefore, we sample more samples

per batch for lower view counts in order to further balance the optimization. Consequently, each batch has roughly the same memory size and information amount.

4.7 Implementation Details

For processing multiple RGB images efficiently and independently with the same ResNet-34, we reshape the input tensor $[B, N, C, W, H]$ to $[B \cdot N, C, W, H]$ with batch size B, number of views N, number of color channels C, image width W, and image height H. This makes the number of CNN weights independent of N. We just have to adapt B depending on N, so that the GPU memory is exploited efficiently.

All RGB-D images that we process have a resolution of 640×480 pixels. Consequently, the corresponding point cloud has 307,200 points which is too large to process it efficiently with a deep neural network. Therefore, we follow He et al. [He+20; He+21] and randomly select 12,800 samples from the point cloud. When fusing the point clouds from N views, we will keep the $12{,}800 \cdot N$ points.

Within our proposed multi-directional point-to-pixel fusion module in SyMFM6D, we consider just the nearest geometric feature vector for each visual feature, i.e. $K_{\text{p2R}} = 1$. For our multi-directional pixel-to-point fusion module, we collect the $K_{\text{R2p}} = 16$ nearest visual features. Both values were proposed by He et al. [He+21] and also worked well for our approach.

For the 3D keypoint detection in MV6D and SyMFM6D, we use $M = 8$ target keypoints for each object which we computed in advance of the training. For computing these target keypoints, MV6D makes use of the FPS algorithm [Eld+97] as proposed in PVN3D [He+20] while SyMFM6D utilizes the SIFT-FPS algorithm [Low99] as introduced in FFB6D [He+21].

4.7.1 Evaluation Metrics

For evaluating our methods and comparing them with other approaches, we follow previous works [Xia+18; Wan+19; He+20; He+21] and use the Area Under the Curve (AUC) metrics for ADD-S and ADD(-S) as well as the percentage of predictions ADD-S < 2 cm and ADD(-S) < 2 cm. Both ADD-S and ADD(-S) are based on the idea of measuring the average distances of all model points in their original status and the same points transformed by the 6D pose.

The average distance metric ADD was introduced by Hinterstoisser et al. in 2012 [Hin+12] and is computed by

$$\text{ADD} = \frac{1}{|\mathcal{M}|} \sum_{\boldsymbol{x} \in \mathcal{M}} \left\| (\widehat{\boldsymbol{R}}\boldsymbol{x} + \widehat{\boldsymbol{t}}) - (\boldsymbol{R}\boldsymbol{x} + \boldsymbol{t}) \right\|, \tag{4.6}$$

where $\mathcal{M}$ is the set of vertices of a given object mesh. $\boldsymbol{p} = [\boldsymbol{R}|\boldsymbol{t}]$ and $\widehat{\boldsymbol{p}} = [\widehat{\boldsymbol{R}}|\widehat{\boldsymbol{t}}]$ are the ground truth pose and the predicted pose respectively. The average distance is calculated between all corresponding vertices of the ground truth pose and the predicted pose.

The average closest point distance metric ADD-S [Hin+12] is a slightly relaxed adaptation of the ADD metric which is more suitable for symmetric objects. It is computed by

$$\text{ADD-S} = \frac{1}{|\mathcal{M}|} \sum_{\boldsymbol{x}_1 \in \mathcal{M}} \min_{\boldsymbol{x}_2 \in \mathcal{M}} \left\| \left(\widehat{\boldsymbol{R}} \boldsymbol{x}_1 + \widehat{\boldsymbol{t}} \right) - (\boldsymbol{R} \boldsymbol{x}_2 + \boldsymbol{t}) \right\|. \tag{4.7}$$

Here, the average distance is computed between each vertex of the predicted pose and the closest vertex of the ground truth pose.

The average (closest point) metric ADD(-S) [He+20] is a combination of the previous two metrics. If it is applied on a symmetric object, the relaxed ADD-S metric is used. If the object is non-symmetric, the stricter ADD metric is applied.

Based on the values of ADD-S and ADD(-S), we compute the area under the accuracy-threshold curve (AUC) and the percentage that is smaller than 2 cm which is a typical threshold for successful robot manipulation.

4.7.2 Baseline Methods

We compare our methods MV6D and SyMFM6D with many established and some very recent methods namely DenseFusion [Wan+19], CosyPose [Lab+20], PVN3D [He+20], FFB6D [He+21], and ES6D [Mo+22].

DenseFusion, PVN3D, FFB6D, and ES6D are single-view 6D pose estimation methods using RGB-D input data. CosyPose is a multi-view method using RGB data. To the best of our knowledge, PVN3D was the best published 6D pose estimator when we started the development of MV6D. The same relation holds for FFB6D and SyMFM6D. CosyPose has been the best published multi-view 6D pose estimator during the development of MV6D and SyMFM6D.

Originally, CosyPose is based on the single-view approach DeepIM [Li+18], uses only RGB data, and can cope with unknown camera poses. However, CosyPose is already outperformed by PVN3D on LineMOD [Hin+11b] and YCB-Video [Xia+18] as both papers suggest [Lab+20; He+20]. In order to create an additional very challenging multi-view 6D pose estimation benchmark based on RGB-D data, we have exchanged the DeepIM network in CosyPose with the PVN3D network for a few selected experiments. Since our approach assumes the camera poses to be known, we evaluate CosyPose not only with unknown camera poses as proposed in the paper [Lab+20] but also using the ground truth camera poses to improve the hypothesis matching and to make the comparison with our approaches fairer.

4.8 Results

This section presents quantitative and qualitative results of our experiments with MV6D and SyMFM6D in comparison to the baseline methods presented in section 4.7.2.

4.8.1 Results on the YCB-Video Dataset

Table 4.1 compares the single-view performance of our SyMFM6D network with all baseline methods using the AUC of ADD-S and ADD(-S) on YCB-Video [Cal+15]. Please note that MV6D corresponds to PVN3D [He+20] in the single-view scenario. The results show that our SyMFM6D method copes very well with the dynamic camera setup of YCB-Video while outperforming all methods significantly. On the symmetry-aware ADD(-S) AUC metric, SyMFM6D outperforms the previous state-of-the-art method FFB6D [He+21] by even 1.5 %. Please note that unlike DenseFusion (iterative) [Wan+19] and CosyPose [Lab+20], our SyMFM6D method does not perform computationally expensive post-processing or iterative refinement procedures.

	ADD-S	ADD(-S)
DenseFusion (per-pixel)	91.2	82.9
DenseFusion (iterative)	93.2	86.1
CosyPose	89.8	84.5
PVN3D	95.5	91.8
FFB6D	96.6	92.7
ES6D	93.6	89.0
SyMFM6D	**96.8**	**94.1**

Table 4.1: Single-view results on the YCB-Video dataset using the AUC metrics for ADD-S and ADD(-S). The best results are printed in bold.

To examine the effect of our symmetry-aware training procedure in SyMFM6D, we provide an object-wise evaluation of the three best single-view methods on YCB-Video in table 4.2. Please note that in single-view mode, the model architecture of our SyMFM6D network is the same as in FFB6D except for our novel symmetry-aware loss function. The results show that not only most symmetric objects (highlighted in bold) are estimated more accurately but also most non-symmetric objects. This indicates that there is a synergy effect which improves the keypoint detection for non-symmetric objects due to an improvement of the keypoint detection for symmetric objects.

Object class	PVN3D	FFB6D	SyMFM6D
Master chef can	80.5	80.6	**80.7**
Cracker box	94.8	94.6	**94.9**
Sugar box	96.3	**96.6**	**96.6**
Tomato soup can	88.5	**89.6**	87.9
Mustard bottle	96.2	97.0	**97.8**
Tuna fish can	89.3	88.9	**92.3**
Pudding box	**95.7**	94.6	93.3
Gelatin box	96.1	**96.9**	96.1
Potted meat can	88.6	88.1	**90.0**
Banana	93.7	94.9	**95.2**
Pitcher base	96.5	96.9	**97.5**
Bleach cleanser	93.2	**94.8**	93.9
Bowl	90.2	96.3	**96.4**
Mug	95.4	94.2	**95.7**
Power drill	95.1	95.9	**96.4**
Wood block	90.4	92.6	**95.2**
Scissors	92.7	95.7	**95.8**
Large marker	**91.8**	89.1	90.0
Large clamp	93.6	96.8	**96.9**
Extra large clamp	88.4	**96.0**	95.3
Foam brick	96.8	97.3	**97.6**
ALL	91.8	92.7	**94.1**

Table 4.2: Single-view results on the YCB-Video dataset evaluated for each object class individually using the ADD(-S) AUC metric. Symmetric objects and the best results are printed in bold.

Figure 4.8 shows a visualization of three scenes of YCB-Video with 6D pose ground truth, predictions of FFB6D, and predictions of our SyMFM6D network using only the depicted view. It can be seen that both FFB6D and SyMFM6D estimate very accurate poses as the scenes of YCB-Video contain only a few objects and not many occlusions. However, SyMFM6D predicts even more accurate poses than FFB6D due to our proposed symmetry-aware training procedure.

Table 4.3 compares the multi-view results of our MV6D method with our SyMFM6D method and CosyPose on the YCB-Video dataset using three and five input views. We see that our SyMFM6D method with disabled symmetry training procedure already outperforms all previous multi-view methods significantly. Enabling symmetry awareness further improves the results slightly. However, using more views does not improve the accuracy as most views of YCB-Video are very similar in which case additional views do

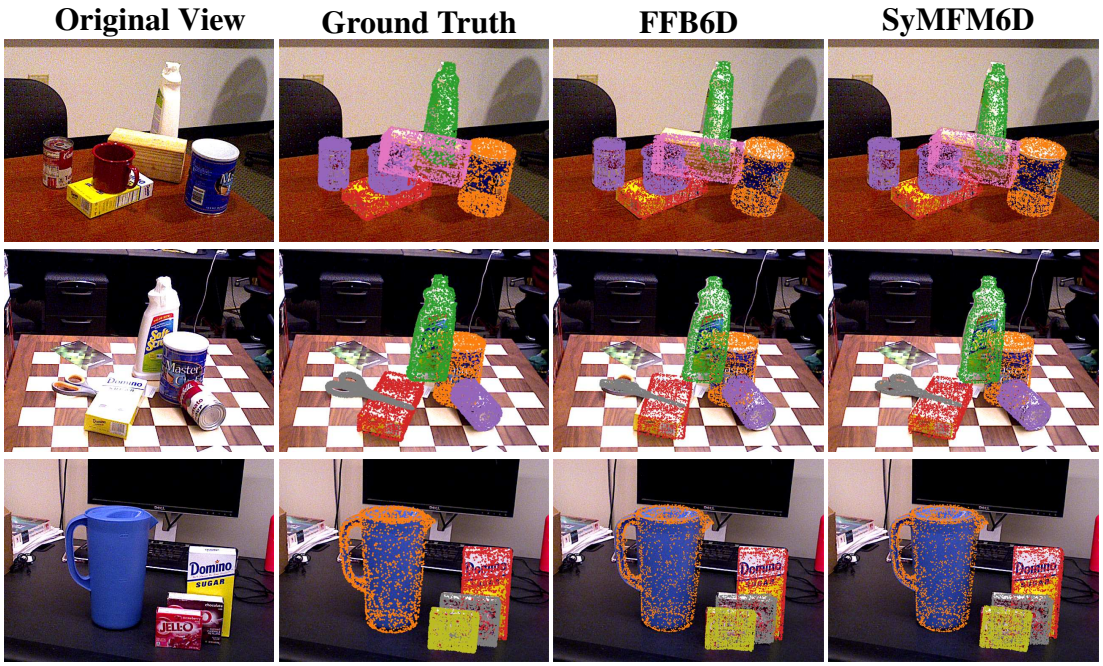


Figure 4.8: Visual comparison of 6D pose predictions on single frames of the YCB-Video dataset. The three rows show three different example scenes that represent the typical performance of the networks.

not provide beneficial information while the learning problem of fusing different views becomes slightly harder.

	ADD-S		ADD(-S)	
	3 views	5 views	3 views	5 views
CosyPose	92.3	93.4	87.7	88.8
MV6D	91.2	91.1	85.6	84.0
SyMFM6D (no sym)	95.2	95.2	91.5	91.4
SyMFM6D	**95.4**	**95.4**	**91.7**	**91.6**

Table 4.3: Quantitative multi-view results on the YCB-Video dataset. The best results are printed in bold.

4.8.2 Results on the MV-YCB FixCam Dataset

Table 4.4 shows the 6D pose estimation accuracy of our MV6D network and our SyMFM6D network using the metrics presented in section 4.7.1. It is compared with the single-view performance of PVN3D [He+20] and FFB6D [He+21] as well as the multi-view performance of CosyPose [Lab+20]. As CosyPose originally does not use depth data and does not provide results on our novel photorealistic datasets, we employ PVN3D as backbone network as described in section 4.7.2. Thus, all methods can use the same RGB-D input data.

	PVN3D	FFB6D	CosyPose	CosyPose	MV6D	SyMFM6D
Number of views	1	1	3	3	3	3
Known cam poses	✓	✓	×	✓	✓	✓
ADD-S AUC	81.3	82.3	90.8	91.9	96.9	**97.3**
ADD(-S) AUC	74.9	76.3	82.4	84.6	94.8	**95.6**
ADD-S < 2 cm	82.1	83.6	92.9	93.0	98.8	**98.9**
ADD(-S) < 2 cm	73.0	74.8	80.6	82.4	96.5	**96.8**

Table 4.4: Quantitative results on the MV-YCB FixCam dataset. The best results for each metric are printed in bold. The presented experiments on CosyPose employ PVN3D as the backbone network so that all methods can use the same RGB-D input data.

We can see that MV6D outperforms PVN3D and CosyPose on all metrics significantly. Using the known camera poses instead of estimating them leads to a small improvement on CosyPose but its accuracy stays significantly lower than ours. Even though the MV-YCB FixCam dataset has heavy occlusions, MV6D can predict more than 96 % of all objects within the 2 cm robot manipulation threshold. SyMFM6D further enhances the accuracy on all metrics in comparison to MV6D. This shows that MV6D and SyMFM6D cope very well with the strong occlusions in the datasets. However, the improvement of SyMFM6D compared to MV6D is small as the MV-YCB FixCam dataset contains only non-symmetric objects so that the symmetry-aware training procedure cannot further enhance the accuracy on this dataset.

Please refer to appendix B.3 for visualizations of some example predictions.

4.8.3 Results on the MV-YCB WiggleCam Dataset

Table 4.5 presents the results on the MV-YCB WiggleCam dataset where the known camera poses deviate slightly from the actual camera poses. It is evident from the table that the inaccurate camera positioning leads to a small accuracy decrease of all approaches. This was to be expected since imprecise camera poses cause inaccuracies in the process of merging the point clouds from the multiple input views. Nevertheless, MV6D and

SyMFM6D are still significantly better than all other baseline methods. This indicates that our methods are robust to inaccurate camera calibration.

	PVN3D	FFB6D	CosyPose	CosyPose	MV6D	SyMFM6D
Number of views	1	1	3	3	3	3
Known cam poses	✓	✓	×	✓	✓	✓
ADD-S AUC	80.8	81.9	90.0	91.3	96.2	**96.7**
ADD(-S) AUC	74.0	75.5	81.0	83.4	93.0	**94.2**
ADD-S < 2 cm	82.0	83.4	92.3	92.6	98.7	**98.8**
ADD(-S) < 2 cm	72.4	74.0	78.9	81.6	**96.0**	**96.0**

Table 4.5: Quantitative results on the MV-YCB WiggleCam dataset. The best results for each metric are printed in bold. The presented experiments on CosyPose employ PVN3D as the backbone network so that all methods can use the same RGB-D input data.

However, SyMFM6D outperforms MV6D only slightly on the WiggleCam dataset. Since the WiggleCam dataset contains the same scenes with only non-symmetric objects as the FixCam dataset, the symmetry-aware training procedure has no effect on the performance on these datasets.

Please refer to appendix B.3 for visualizations of some example predictions.

4.8.4 Results on the MV-YCB SymMovCam Dataset

Table 4.6 shows the quantitative results of our methods MV6D and SyMFM6D in comparison to the baseline methods PVN3D [He+20] and FFB6D [He+21] on our MV-YCB SymMovCam dataset. Please note that MV6D corresponds to PVN3D when using just a single input view. It becomes evident from the single-view results in the table that the symmetry-aware training procedure in SyMFM6D leads to a small improvement compared to FFB6D. When using multiple views, MV6D and SyMFM6D outperform the

	PVN3D	FFB6D	SyMFM6D	MV6D	SyMFM6D
Number of views	1	1	1	3	3
Known cam poses	✓	✓	✓	✓	✓
ADD-S AUC	75.0	79.9	80.6	92.8	**94.2**
ADD(-S) AUC	68.5	75.6	76.7	88.7	**91.6**
ADD-S < 2 cm	77.2	81.1	81.9	96.3	**96.6**
ADD(-S) < 2 cm	64.5	74.5	76.3	91.6	**93.6**

Table 4.6: Quantitative results on the MV-YCB SymMovCam dataset. The best results for each metric are printed in bold.

single-view methods significantly. This also proves that our approaches are robust to the very dynamic camera setup in the MV-YCB SymMovCam dataset where the cameras are mounted at varying positions. Moreover, SyMFM6D surpasses MV6D due to the better fusion mechanism and the symmetry-aware training procedure.

Figure 4.9 illustrates some 6D pose estimation results of SyMFM6D, MV6D, and FFB6D on the SymMovCam dataset. The four columns show four different example scenes. The first three rows provide the three input views for MV6D and SyMFM6D. The single-view network FFB6D obtains only the first view as input. It becomes evident, that FFB6D has disadvantages predicting poses of objects which are fully or partly occluded in the input view. In accordance with the quantitative results in table 4.6, MV6D provides very accurate results, even though SyMFM6D yields even better predictions, especially on symmetric objects.

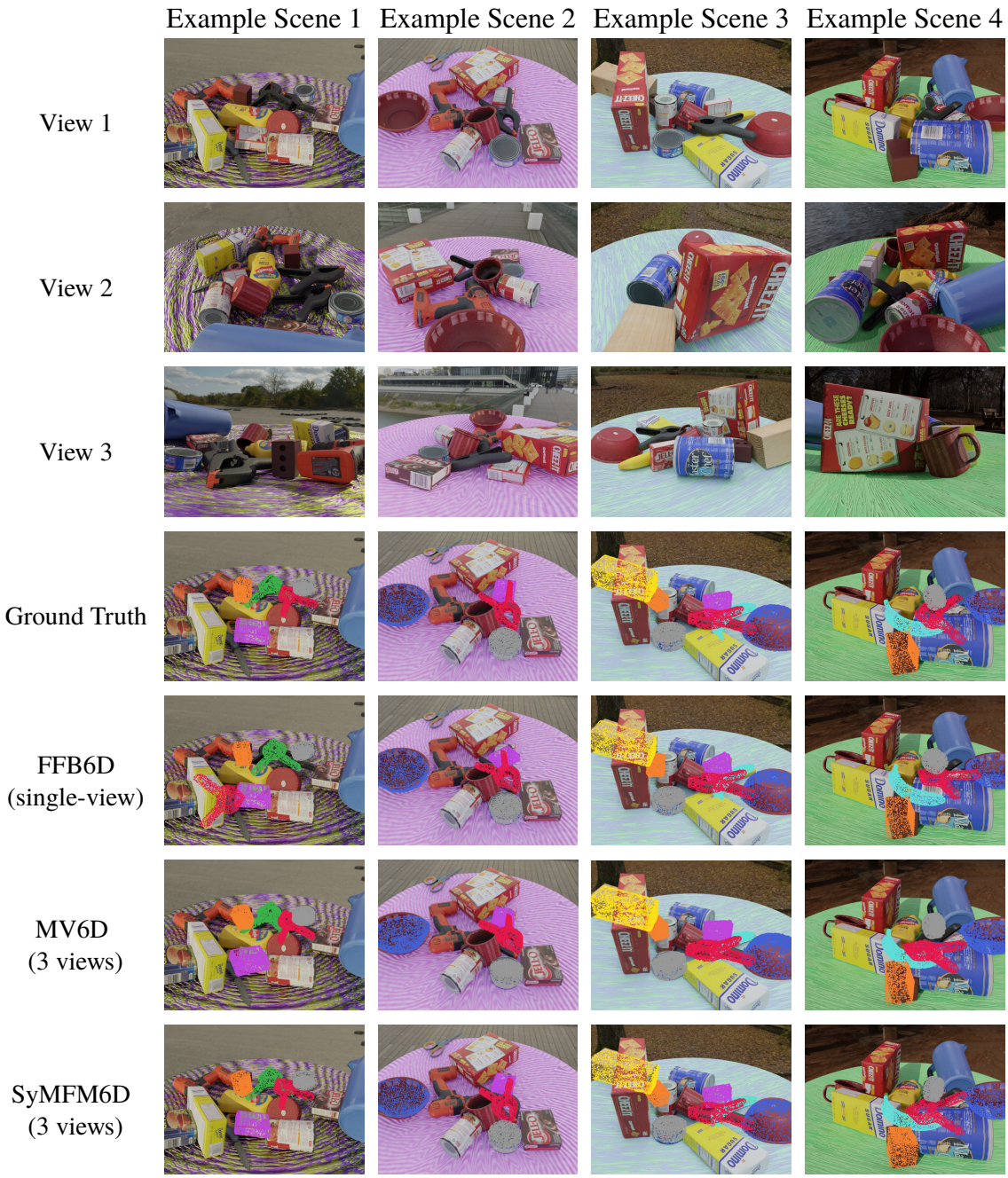


Figure 4.9: Visual comparison of predicted poses on different scenes of the MV-YCB SymMovCam dataset. The seven rows show the first three views which are used for the multi-view methods and four different pose visualizations using the first view as reference. Due to the strong occlusions, only the poses of eight of the most challenging objects are depicted, including the symmetric objects bowl (blue), wood block (yellow), large clamp (red), extra large clamp (green), foam brick (orange), and the non-symmetric objects tuna fish can (gray), banana (cyan), and gelatin box (magenta).

4.8.5 Keypoint Visualization

Figure 4.10 shows the predicted keypoints of FFB6D and our SyMFM6D network for all objects in a YCB-Video scene. We additionally visualize the keypoint proposals of each object in individual colors. The resulting predicted keypoints are white, the target keypoints are black. The figure provides visual evidence that both FFB6D and SyMFM6D predict very accurate keypoints on all non-symmetric objects. However, FFB6D fails to predict accurate keypoints on the large clamp which has one discrete rotational symmetry. This shortcoming of FFB6D is also apparent for other symmetric objects. We believe that this is caused by the ambiguities of the object poses resulting in ambiguous target keypoints which results in averaging over the multiple solutions given by the symmetry. Therefore, the training loss is minimized when predicting keypoints on the symmetric axis rather than predicting them on the desired target locations. SyMFM6D, in contrast, overcomes this problem due to our novel symmetry-aware training procedure as it can be seen in figure 4.10b.

(a) FFB6D (b) SyMFM6D

Figure 4.10: Visualization of the predicted keypoints on single frames of the YCB-Video dataset. The estimated keypoint proposals are depicted as small colored points with an individual color for each object class: bleach cleanser (gray), cracker box (green), power drill (magenta), sugar box (brown), and large clamp (cyan). The resulting keypoint predictions are highlighted as white circles with red center points. The target keypoints are black circles with yellow center points.

4.9 Runtime for MV6D and SyMFM6D

We provide the runtimes of our methods MV6D and SyMFM6D for different numbers of views from the YCB-Video dataset in table 4.7 and table 4.8 respectively. The network forward time includes the 3D keypoint offset prediction, the center point offset prediction, and the prediction of semantic labels. The pose estimation time represents the time for applying the mean shift clustering algorithm and the least-squares fitting algorithm for computing the 6D pose of a single object. Please note that the usage of the symmetry-aware loss does increase the training time slightly, but it does not affect the runtime. The runtimes are measured using a single GPU of type NVIDIA Tesla V100 with 32 GB of memory.

The results in the tables reveal a significant speedup of SyMFM6D compared to MV6D with total times being reduced by approximately 60 %. This is due to the faster network inference of SyMFM6D. Its network forward times are almost one-third compared to MV6D. The pose estimation time of both methods are identical as the same algorithms are used in the associated modules.

Number of Views	Network Forward Time	Pose Estimation Time	Total Time
1	135 ms	14 ms	149 ms
2	270 ms	19 ms	289 ms
3	400 ms	25 ms	425 ms
4	530 ms	30 ms	560 ms
5	660 ms	36 ms	696 ms

Table 4.7: Runtimes of MV6D for a different number of input views from the YCB-Video dataset.

Number of Views	Network Forward Time	Pose Estimation Time	Total Time
1	46 ms	14 ms	60 ms
2	92 ms	19 ms	111 ms
3	138 ms	25 ms	163 ms
4	184 ms	30 ms	214 ms
5	230 ms	36 ms	266 ms

Table 4.8: Runtimes of SyMFM6D for a different number of input views from the YCB-Video dataset.

4.10 Conclusion

In this chapter, we have presented two novel deep learning methods for multi-view 6D pose estimation based on RGB-D data, namely MV6D and SyMFM6D. Both methods incorporate different strategies for fusing RGB and depth data from multiple RGB-D recordings showing very cluttered object scenes. Based on the fused features, MV6D and SyMFM6D predict 3D keypoints, 3D center points, and semantic labels, which are used to compute the 6D poses of all objects in the scene using a least-squares fitting algorithm. We have additionally proposed a novel method for predicting predefined 3D keypoints of symmetric objects relying on a symmetry-aware objective function.

To examine the strengths and limitations of our methods, we have utilized challenging real-world data and created four novel photorealistic datasets showcasing cluttered scenes with multiple RGB-D cameras from very different perspectives. Our experiments demonstrate that our first method MV6D outperforms the previous state-of-the-art in multi-view 6D pose estimation significantly on these datasets. FFB6D yields an even higher accuracy while being computationally more efficient. Besides, we show that our symmetry-aware training procedure improves the 6D pose estimation accuracy of both symmetric and non-symmetric objects due to synergy effects resulting in exceeding the previous state-of-the-art in single-view 6D pose estimation. Moreover, we demonstrate the robustness of our methods towards inaccurately known camera poses and variable camera arrangements.

Chapter 5

Deep Hierarchical Variational Autoencoding for RGB Image Fusion

This chapter investigates novel data fusion approaches to combine multiple imperfect visual data inputs while considering uncertainty and prior knowledge about the data. Parts of this chapter are taken from my publication "FusionVAE: A Deep Hierarchical Variational Autoencoder for RGB Image Fusion" [Duf+22], written by Fabian Duffhauss, Ngo Anh Vien, Hanna Ziesche, and Gerhard Neumann, which has been presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2022. Please refer to the third main row of tables 1.1 and 1.2 to see the fusion modalities and most important challenges addressed in this chapter in comparison to the other main chapters.

5.1 Introduction

Sensor fusion is a popular technique in computer vision as it allows to combine the individual advantages of multiple information sources. It is especially gainful in scenarios where a single sensor is not able to capture all necessary data to perform a task satisfactorily. Over the last years, we have seen many examples, where the accuracy of computer vision tasks was significantly improved by sensor fusion, e.g. in environmental perception for autonomous driving [Che+17c; Ku+18; Yoo+20], for 6D pose estimation [Wan+19; He+20; He+21], and for robotic grasping [Zha+17a; Wan+20]. However, traditional fusion methods usually focus more on the beneficial merging of multiple modalities and less on teaching the model to obtain profound prior knowledge about the used dataset.

Our work tries to fill in this research gap by proposing a deep hierarchical variational autoencoder called FusionVAE that is able to perform both tasks: fusing information from multiple sources and supplementing it with prior knowledge about the data gained while training. As shown in figure 5.1, FusionVAE merges a varying number of input images for reconstructing the original target image using prior knowledge about the dataset. To the best of our knowledge, FusionVAE is the first approach that combines these two tasks. Therefore, we developed three challenging benchmarks based on well-known computer

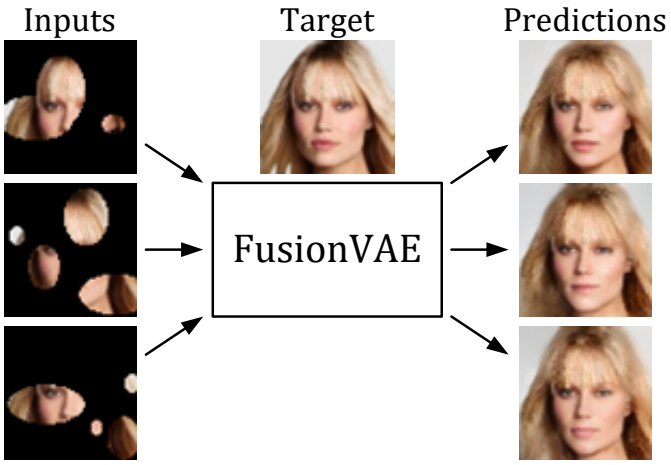


Figure 5.1: Overview of our FusionVAE approach. The network receives up to three partly occluded input images, fuses them together with prior knowledge, and predicts different hypotheses of how the target images could look like.

vision datasets to evaluate the performance of our approach. In addition, we perform comparisons to baselines by extending traditional approaches to perform the same tasks. FusionVAE outperforms all these traditional methods on all proposed benchmark tasks significantly. We show that FusionVAE can generate high-quality images given few input images with partial observability. We provide ablation studies to illustrate the impact of commonly used information aggregation operations and to prove the benefits of the employed posterior distribution.

We can summarize the four main contributions of our work in this field as follows:

- We create three challenging image fusion tasks for generative models.
- We develop a deep hierarchical VAE called FusionVAE that is able to perform image-based data fusion while employing prior knowledge of the used dataset.
- We show that FusionVAE produces high-quality fused output images and outperforms traditional methods by a large margin.
- We perform ablation studies showing the benefits of our design choices regarding both the posterior distribution and commonly used aggregation methods.

5.2 Related Work

In this section, we present related work about image generation, image fusion, and image completion. Please refer to section 2.5.1 for further details about the fundamentals of VAEs including related work.

5.2.1 VAE-based Image Generation

VAEs are powerful networks that are able to compress the essence of datasets in a small latent space while being able to exploit it for generative tasks [KW14]. However, the standard VAE is limited in capacity and expressiveness and thus, when applied to image generation leads to over-smoothed results lacking fine-grained details. Over the last years, much work has been invested into the effort of improving the generative performance of VAEs. One stream of work is based on introducing a hierarchy into the latent space of the VAE and scaling this hierarchy to greater and greater depth. First introduced in [Søn+16] many hierarchical VAEs are based on coupling the inference and generative processes by introducing a deterministic bottom-up path combined with a stochastic top-down process in the inference network and sharing the latter with the generative model. This setting has been extended by an additional deterministic top-down path and bidirectional inference in [Maa+19]. Recently, very deep hierarchical VAEs were realized in [Chi21] by introducing residual bottlenecks with dedicated scaling, update skipping, and nearest neighbor up-sampling. Closest to our work is the recently proposed NVAE architecture [VK20], which relies on depth-wise convolution, residual posterior parametrization, and spectral regularization to enhance stability.

Other approaches propose increasing the expressiveness of VAEs by combining them with auto-regressive models like RNNs or PixelCNNs [Che+17a; Gul+17; Sad+19; Gre+15], conditioning contexts (CVAE) [SLY15; Wal+16], normalizing flows [Kin+16b], GANs [Lar+16; Par+21], or variational generative adversarial networks (CVAE-GANs) [Bao+17].

5.2.2 Fusion of Multiple Images

Image fusion has been dominated by classical computer vision for a long time. Only lately deep learning methods have entered the domain starting with the CNN-based approach proposed by Liu et al. [Liu+17b]. In a subsequent publication, the authors have extended their work to a multi-scale setting [Liu+17a]. Shortly afterwards, Prabhakar et al. have developed a fusion method based on a siamese network architecture, called DeepFuse [PSB17] which was improved in subsequent work [LW18] by employing the DenseNet architecture [Hua+17]. Concurrently, Li et al. [LWK18] have proposed a fusion architecture based on VGG [SZ15] and another one [LWD19] based on ResNet-50 [He+16] in order to scale to even greater depth. The aforementioned methods use CNNs as feature extractors and as decoders, while the fusion operations themselves are restricted to classical methods like averaging or addition of feature maps or weighted source images. A fully CNN-based feature-map fusion mechanism has been proposed in [Jun+20].

While all previous publications target only a specific fusion task (e.g. multi-focus fusion, multi-resolution fusion, etc.) or were limited to specific domains (e.g. medical images), two very recent works propose novel multi-purpose fusion networks, which are applicable

to many fusion tasks and image types [Xu+20b; Zha+20a]. Very recently also GAN-based methods entered the domain of image fusion, starting with the work by Ma et al. on infrared-visible fusion [Ma+20a; Ma+19a] and with [Ma+20b; Xu+19] on multi-resolution image fusion. Most recent are two publications on GAN-based multi-focus image fusion [Guo+19; Hua+20]. While GAN-based approaches can generate high-fidelity images, it is known that they suffer from the mode collapse problem. VAE-based methods in contrast are known to generate more faithful data distribution [VK20]. Different from previous work, this paper proposes a VAE-based multi-purpose fusion framework.

5.2.3 Image Completion

Similar to image fusion, image completion (also called image inpainting) has faced significant advancements since the emergence of deep learning methods. First approaches based on simple MLPs [Köh+14] or CNNs [Gu+18] were targeted only to fill small holes in an image. However, with the introduction of GANs [Goo+14], the area quickly became dominated by GAN-based approaches, starting with the context encoders presented by Pathak et al. [Pat+16]. Many subsequent papers proposed extensions to this model in order to obtain fine-grained completions while preserving global coherence by introducing additional discriminators [ISI17], searching for closest samples to the corrupted image in a latent embedding space conditioning on semantic labels [Son+18b], or designing additional specialized loss functions [Li+17]. High-resolution results were obtained recently by multi-scale approaches [Yan+17], iterative upsampling [Zen+20b], and the application of contextual attention [Yu+18; Son+18a; Yan+18]. Another stream of current work focuses on multi-hypothesis image completion, leveraging probabilistic problem formulations [ZCC19; MMG21].

5.3 Conditional Generative Models for Image Fusion

We propose a deep hierarchical conditional variational autoencoder, called FusionVAE *(Fusion Variational Auto-Encoder)*, that is able to fuse information from multiple sources and to infer the missing information in the images from a prior learned from the dataset. To the best of our knowledge, it is the first model that combines the generative ability of a hierarchical VAE to learn the underlying distribution of complex datasets with the ability to fuse multiple input images.

5.3.1 Problem Formulation

We consider image fusion problems that are concerned with generating the fused target image from multiple source images. Each source image contains partial information of the target image and the goal of the task is to recover the original target image given a

finite set of source images. In particular, we denote the target image as $\boldsymbol{y}$ and the set of K source images as context $\boldsymbol{x} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_K\}$, where each $\boldsymbol{x}_i$ is one source image. Given training sample $(\boldsymbol{x}, \boldsymbol{y})$, we aim to maximize the conditional likelihood $p(\boldsymbol{y}|\boldsymbol{x})$.

Please refer to sections 2.5.1 to 2.5.1 for fundamentals about standard VAEs, CVAEs, and hierarchical VAEs upon which we build our work.

5.3.2 Training Objective Derivation

As previously mentioned, our approach is designed to maximize the conditional likelihood $p_\theta(\boldsymbol{y}|\boldsymbol{x})$. However, optimizing this objective directly is intractable. Therefore, we derive a variational lower bound as follows.

We start with the KL divergence between the approximate posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{y})$ and the true posterior distribution $p_\theta(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})$,

$$\mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{y})||p_\theta(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{y})) \geq 0. \tag{5.1}$$

Next, we apply the Bayes's theorem to obtain

$$-\int q_\phi(\boldsymbol{z}|\boldsymbol{y}) \log \frac{p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z}) p_\theta(\boldsymbol{z}|\boldsymbol{x})}{p_\theta(\boldsymbol{y}|\boldsymbol{x}) q_\phi(\boldsymbol{z}|\boldsymbol{y})} \mathrm{d}\boldsymbol{z} \geq 0. \tag{5.2}$$

This is equivalent to

$$\begin{aligned} -\mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{y})}[\log p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})] - \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{y})||p_\theta(\boldsymbol{z}|\boldsymbol{x})) \\ + \int q_\phi(\boldsymbol{z}|\boldsymbol{y}) \log p_\theta(\boldsymbol{y}|\boldsymbol{x}) d\boldsymbol{z} \geq 0. \end{aligned} \tag{5.3}$$

The conditional log-likelihood $\log p_\theta(\boldsymbol{y}|\boldsymbol{x})$ can be moved out from the third integral component and leaves the integral becoming 1. Thus, we obtain the variational lower bound of the conditional log-likelihood

$$\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) \geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{y})}[\log p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})] + \mathrm{KL}(q_\phi(\boldsymbol{z}|\boldsymbol{y})||p_\theta(\boldsymbol{z}|\boldsymbol{x})). \tag{5.4}$$

As we propose a hierarchical VAE, we split the latent variables $\boldsymbol{z}$ into L disjoint groups $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_L$. Furthermore, we introduce annealing parameters β and α_l that control the warming-up of the KL terms as in [VK20]. This leads to the improved variational lower bound of our FusionVAE

$$\begin{aligned} \log p_\theta(\boldsymbol{y}|\boldsymbol{x}) &\geq \mathbb{E}_{q_\phi(\boldsymbol{z}|\boldsymbol{y})}[\log p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})] \\ &\quad - \beta \sum_{l=1}^{L} \alpha_l \mathbb{E}_{q_\phi(\boldsymbol{z}_{<l}|\boldsymbol{y})}[\mathrm{KL}(q_\phi(\boldsymbol{z}_l|\boldsymbol{y}, \boldsymbol{z}_{<l})||p_\theta(\boldsymbol{z}_l|\boldsymbol{x}, \boldsymbol{z}_{<l}))] \\ &\quad = \mathrm{ELBO}_{FusionVAE}(\boldsymbol{x}, \boldsymbol{y}). \end{aligned} \tag{5.5}$$

Inspired by [Søn+16], we increase β linearly from 0 to 1 during the first few training epochs to start training the reconstruction before introducing the KL term, which is increased gradually. α_l is a KL balancing coefficient as in [Vah+18] that is used during the warm-up period to encourage the equal use of all latent groups and to avoid posterior collapse.

5.3.3 Network Architecture

Figure 5.2 illustrates the network architecture of our FusionVAE for training. It is built in a hierarchical way inspired by [VK20]. In each latent hierarchy $l \in 1, \ldots, L$ we have a set of feature maps $\boldsymbol{f}_{lx}$, $\boldsymbol{f}_{ly}$ and latent distributions p_l, q_l.

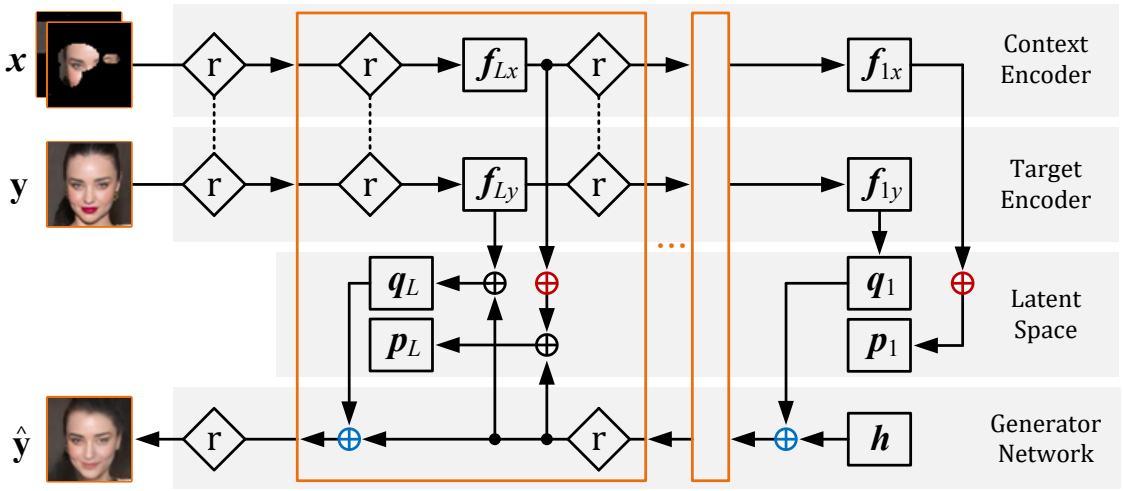


Figure 5.2: Overview of the proposed FusionVAE network architecture. $\boldsymbol{h}$ is a trainable parameter vector, $\oplus$ denotes concatenation, $\oplus$ max aggregation, and $\oplus$ pixel-wise addition. $\diamond$ is a residual network like in [VK20]. The dotted lines between the residual networks indicate shared parameters.

FusionVAE consists of four parts including three main networks and a latent space, each highlighted by a gray box. The first gray box contains the context encoder network that obtains a stack of source images $\boldsymbol{x}$ and employs residual cells [He+16] as in [VK20] to extract features $\boldsymbol{f}_{lx}$. Thus, it models the conditional prior distribution $p_\phi(\boldsymbol{z}_l|\boldsymbol{x}, \boldsymbol{z}_{<l})$.

The second gray box shows the target encoder network that encodes the target image $\boldsymbol{y}$ into the feature map $\boldsymbol{f}_{ly}$ using the same residual cells as the context encoder. Thus, it models the approximate posterior distribution $q_\phi(\boldsymbol{z}|\boldsymbol{y})$

The third gray box illustrates the latent space comprising the L latent groups which contain the prior distributions $p_\phi(\boldsymbol{z}_l|\boldsymbol{x}, \boldsymbol{z}_{<l})$ (denoted p_1, ..., p_L in figure 5.2) and the approximate posterior distributions $q_\phi(\boldsymbol{z}_l|\boldsymbol{y}, \boldsymbol{z}_{<l})$ (denoted q_1, ..., q_L in figure 5.2).

The fourth gray box contains the generator network $p_\theta(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{z})$ that aims to reconstruct the target image $\boldsymbol{y}$ by creating different output samples $\hat{\boldsymbol{y}}$. It employs a trainable parameter

vector $\boldsymbol{h}$, concatenates the information from all hierarchies, and decodes them using residual cells.

In each latent hierarchy, we aggregate the context features $\boldsymbol{f}_{lx}$ using pixel-wise max aggregation. In all but the first hierarchy, we pixel-wisely add the corresponding feature map from the generator network to the aggregated context features and to the target image features f_{ly}. Using 2D convolutional layers, we learn the prior distributions p_l and the approximate posterior distributions q_l. We propose to use the approximate posterior distributions q_l as target distributions in order to learn good prior distributions p_l. Therefore, $q_\phi(\boldsymbol{z}_l|\boldsymbol{y}, \boldsymbol{z}_{<l})$ is created from the target image features f_{ly} as well as information from the generator network.

During training the generator network aims to create a prediction $\hat{\boldsymbol{y}}$ based on samples of the posterior distributions q_l and a trainable parameter vector $\boldsymbol{h}$.

For evaluation, we can omit the target image input $\boldsymbol{y}$ and sample from the prior distributions p_l. In case no input image is given, we set p_1 to a standard normal distribution. Based on the samples and the trainable parameter vector $\boldsymbol{h}$, our FusionVAE can generate new output images.

5.4 Experimental Setup

To evaluate our approach, we have conducted a series of experiments on three different datasets using data augmentation. Furthermore, we have adapted traditional architectures for solving the same tasks in order to compare our results. Finally, we perform an ablation study to show the effects of specific design choices.

5.4.1 Datasets

For training and evaluating our approach, we have created three novel fusion datasets based on the datasets MNIST [LeC98], CelebA [Liu+15], and T-LESS [Hod+17] as described in the following.

FusionMNIST. Based on the MNIST dataset [LeC98], we have created an image fusion dataset called FusionMNIST. For each target image, it contains different noisy representations where only random parts of the target image are visible. The first three columns of figure 5.3 show different examples of FusionMNIST corresponding to the target images in the fourth column. To generate FusionMNIST, we have applied zero padding to all MNIST images to obtain a resolution of 32×32. For creating a noisy representation, we generated a mask composed of the union of a varying number of ellipses with random sizes, shapes, and positions. All parts of the given images outside the mask are blackened. Finally, we added Gaussian noise with a fixed variance and clip the pixel values afterwards to stay within the interval $[0, 1]$.

FusionCelebA. We have generated a similar fusion dataset based on the aligned and cropped version of CelebA [Liu+15] which we call FusionCelebA. Figure 5.4 depicts different example images in the first three columns which belong to the target image in the fourth column. To generate FusionCelebA, we center-cropped the CelebA images to 148×148 before scaling them down to 64×64 as proposed by [Lar+16]. As in FusionMNIST, we created different representations by using masks based on random ellipses.

FusionT-LESS. A promising application area for our FusionVAE is robot vision. Scenes in robotics settings can be very difficult to understand due to texture-less or reflective objects and occlusions. To examine the performance of our FusionVAE in this area, we have created an object dataset with challenging occlusions based on T-LESS [Hod+17] which we call FusionT-LESS. To generate FusionT-LESS, we used the real training images of T-LESS and take all images of classes 19 – 24 as the basis for the target images. This selection contains all objects with power sockets and therefore images with many similarities. Every tenth image is removed from the training set and used for evaluation. In order to create challenging occlusions, we cut all objects from images of other classes using a Canny edge detector [Can86] and overlay each target image with a random number between five and eight cropped objects. We selected all images from classes 1, 2, 5 – 7, 11 – 14, and 25 – 27 as occluding objects for training and classes 3, 4, 8 – 10, 15 – 18, and 28 – 30 for evaluation.

5.4.2 Data Augmentation

During training, we apply different augmentation methods on the datasets to avoid overfitting. For FusionMNIST, we apply the elliptical mask generation and the addition of Gaussian noise live during training so that we obtain an infinite number of different fusion tasks. For FusionT-LESS, almost the entire creation of occluded images is performed during training. We apply horizontal flips, rotations, scaling, and movement of target and occluding images with random parameters before composing the different occluded representations. Solely the object cutting with the Canny edge detector is performed offline as a pre-processing step to keep the training time low. For FusionCelebA, we apply a horizontal flip of all images randomly in 50 % of all occasions, and also the elliptical mask generation is done live during training.

5.4.3 Architectures for Comparison

To the best of our knowledge, FusionVAE is the first fusion network for multiple images with a generative ability to fill areas without input information based on prior knowledge about the dataset under consideration. Due to the absence of another suitable model from

the literature which would allow a fair comparison on our multi-image fusion benchmarks, we compare our approach with standard architectures that we adapted to support our tasks.

The first architecture for comparison is a CVAE with residual layers as employed in [VK20]. We use a shared encoder for processing the input images and applied max aggregation before the latent space as we did in our FusionVAE. The second architecture for comparison is a Fully Convolutional Network (FCN) with a shared encoder and max aggregation before the decoder.

For both baseline architectures, we created a version with skip connections (denoted as +S) and a version without. When using skip connections, we applied max aggregation at each shortcut for merging the features from the encoder with the decoder's features. To allow for a fair comparison, we designed all architectures so that they have a similar number of trainable parameters.

5.4.4 Training Procedure

We have trained all networks in a supervised manner using the augmented target images $\boldsymbol{y}$ as described in section 5.4.2. In order to teach the networks both to fuse information from a different number of input images and to learn prior knowledge about the dataset, we varied the number of input images $\boldsymbol{x}$ during the entire training. Specifically, we selected a uniformly distributed random number between zero and three for each batch.

5.4.5 Implementation Details of FusionVAE

For FusionMNIST and FusionT-LESS, we model the decoder's output by pixel-wise independent Bernoulli distributions. For FusionCelebA, we use pixel-wise independent discretized logistic mixture distributions as proposed by Salimans et al. [Sal+17].

The residual cells of the encoder are composed of batch normalization layers [IS15], Swish activation functions [RZL18], convolutional layers, and Squeeze-and-Excitation (SE) blocks [HSS18] as proposed in [VK20]. In the decoder, we also follow [VK20] and build the residual cells out of batch normalization layers, 1x1 convolutions, Swish activations, depthwise separable convolutions [Cho17], and SE blocks. However, we omitted normalizing flow because in our experiments it showed to increase the training time without improving the prediction accuracy significantly.

For each dataset, we have chosen the size of our FusionVAE architecture individually to achieve acceptable accuracy while keeping the training time reasonable. Table 5.1 provides details about the used hyperparameters and further properties of our experiments.

In general, the number of latent groups L should be chosen depending on the complexity of the task at hand. We made our decision based on the L of the NVAE [VK20] but reduced it for computational reasons. For FusionCelebA and FusionT-LESS, we use 17

Hyperparameter	FusionMNIST	FusionCelebA	FusionT-LESS
# latent groups per scale	5, 2	10, 5, 2	10, 5, 2
Spatial dimensions of z_l per scale	$4^2, 8^2$	$8^2, 16^2, 32^2$	$8^2, 16^2, 32^2$
# channels in z_l	10	20	20
# GPUs	2	4	2
# training epochs	400	90	500
Batch size	800	32	32
Training time	4 h	48 h	28 h

Table 5.1: Main hyperparameters and characteristics of our experiments with FusionVAE.

latent groups, and for FusionMNIST only seven. Using more latent groups improves the results but increases the computational effort significantly.

For all experiments, we used GPUs of type NVIDIA Tesla V100 with 32 GB of memory and trained with an AdaMax optimizer [KB15]. We applied a cosine annealing schedule for the learning rate [LH17] starting at 0.01 and ending at 0.0001.

5.4.6 Evaluation Metrics

For evaluation, we estimate the Negative Log-Likelihood (NLL) in Bits Per Dimension (BPD) using weighted importance sampling [BGS16]. We use 100 samples for all experiments with FusionCelebA as well as FusionT-LESS and 1000 samples for FusionMNIST. Since we cannot estimate the NLL of the FCN, we used the minimum over all samples of the mean squared error (MSE_{min}) as a second metric.

5.5 Results

This section presents and discusses the quantitative and qualitative results of our research in comparison to the baseline methods mentioned in section 5.4.3.

5.5.1 Quantitative Results

Tables 5.2 to 5.4 show the NLL and the MSE_{min} of all architectures on FusionMNIST, FusionCelebA, and FusionT-LESS respectively. The results are divided into the results based on zero to three input images and the average (avg) of it. We see that our FusionVAE outperforms all baseline methods on average. Regarding the NLL, our model surpasses the others additionally for 0 and 1 input images. For 2 and 3 images, CVAE+S reaches sometimes slightly better NLL values. However, our approach reaches the best MSE_{min} values for each number of input images.

	NLL in 10^{-2} BPD					MSE_{min} in 10^{-2}				
	0	1	2	3	avg	0	1	2	3	avg
FCN						10.99	5.81	5.78	5.79	7.25
FCN+S						5.80	3.74	2.54	1.78	3.56
CVAE	17.81	15.01	14.07	13.61	15.23	3.83	1.72	1.05	0.80	1.93
CVAE+S	18.43	14.57	**13.18**	**12.30**	14.77	3.62	1.75	1.19	0.97	1.95
FusionVAE	**15.93**	**14.17**	13.70	13.48	**14.39**	**3.14**	**0.99**	**0.74**	**0.65**	**1.45**

Table 5.2: Results on the FusionMNIST dataset. The columns show the metric outcomes for zero to three input images (denoted as 0, 1, 2, 3) together with an average (avg). The best results are printed in bold.

	NLL in 10^{-2} BPD					MSE_{min} in 10^{-2}				
	0	1	2	3	avg	0	1	2	3	avg
FCN						13.77	14.82	13.10	11.24	13.23
FCN+S						12.56	8.96	6.06	4.09	7.92
CVAE	446.0	280.1	273.5	266.5	316.7	9.23	3.46	2.27	1.55	4.14
CVAE+S	525.0	270.1	233.5	**203.5**	308.3	11.08	5.49	3.66	2.57	5.71
FusionVAE	**248.1**	**227.6**	**231.2**	228.7	**233.9**	**5.11**	**0.88**	**0.86**	**0.84**	**1.93**

Table 5.3: Results on the FusionCelebA dataset. The best results are printed in bold.

	NLL in 10^{-2} BPD					MSE_{min} in 10^{-2}				
	0	1	2	3	avg	0	1	2	3	avg
FCN						5.83	3.34	2.37	1.82	3.35
FCN+S						8.06	1.84	1.13	0.74	2.97
CVAE	25.24	23.73	22.70	23.13	23.71	5.57	1.54	0.77	0.37	2.08
CVAE+S	26.08	24.94	23.98	23.95	24.75	4.95	2.50	1.77	1.19	2.62
FusionVAE	**24.18**	**23.07**	**22.23**	**22.88**	**23.10**	**4.11**	**0.59**	**0.32**	**0.19**	**1.32**

Table 5.4: Results on the FusionT-LESS dataset. The best results are printed in bold.

5.5.2 Qualitative Results

The outstanding performance of our architecture in comparison to the others is also obvious when looking at the qualitative results in figures 5.3 to 5.5. For every row, these figures show the input, target, and up to three output predictions for all architectures. For the FCN, we depict just a single output prediction per row as all of them look almost identical.

In the first three rows when the network does not receive any input image, we see that our network provides very realistic images. This indicates that it is able to capture the underlying distribution of the used datasets very well and much better than the other architectures. Due to the difficulty of the FusionT-LESS dataset, none of the models is able to produce realistic images without any input. Still our model shows much better performance in generating object-like shapes. In case the models receive at least one input image (cf. rows 4 – 12), all architectures are able to extract the available information from the given input images. In addition, all VAE approaches, ours included, are able to complete the given input data based on prior knowledge. It is clearly visible, however, that the predictions of our model are much more realistic than the ones of the standard CVAE approaches especially for the more difficult datasets like FusionCelebA and FusionT-LESS.

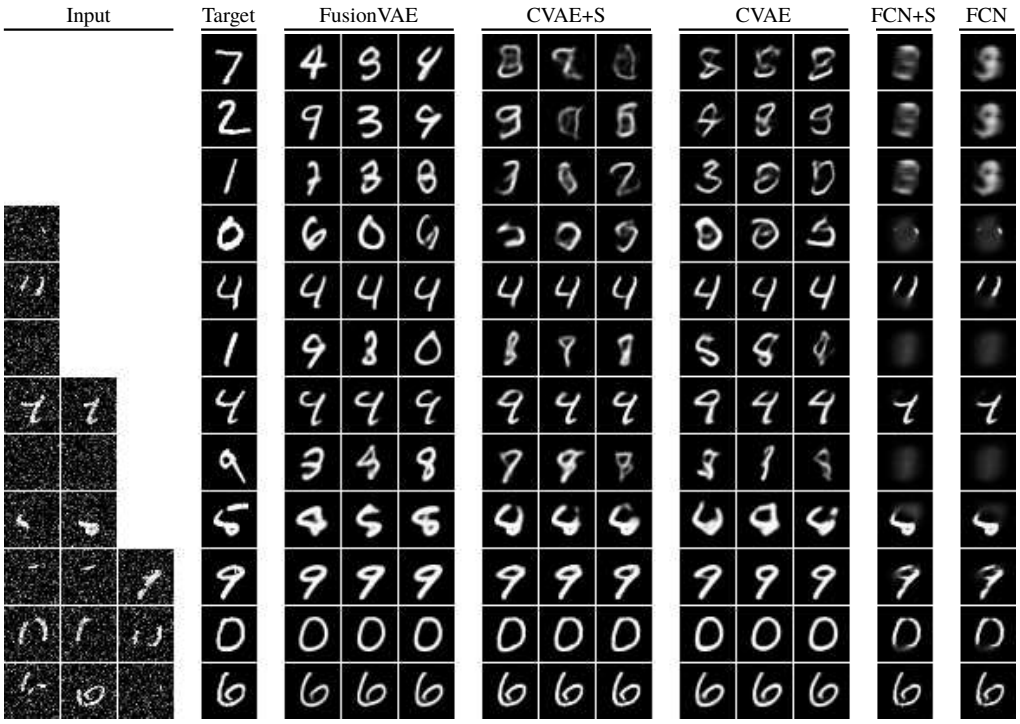


Figure 5.3: Prediction results of the different architectures on the FusionMNIST dataset for zero to three input images.

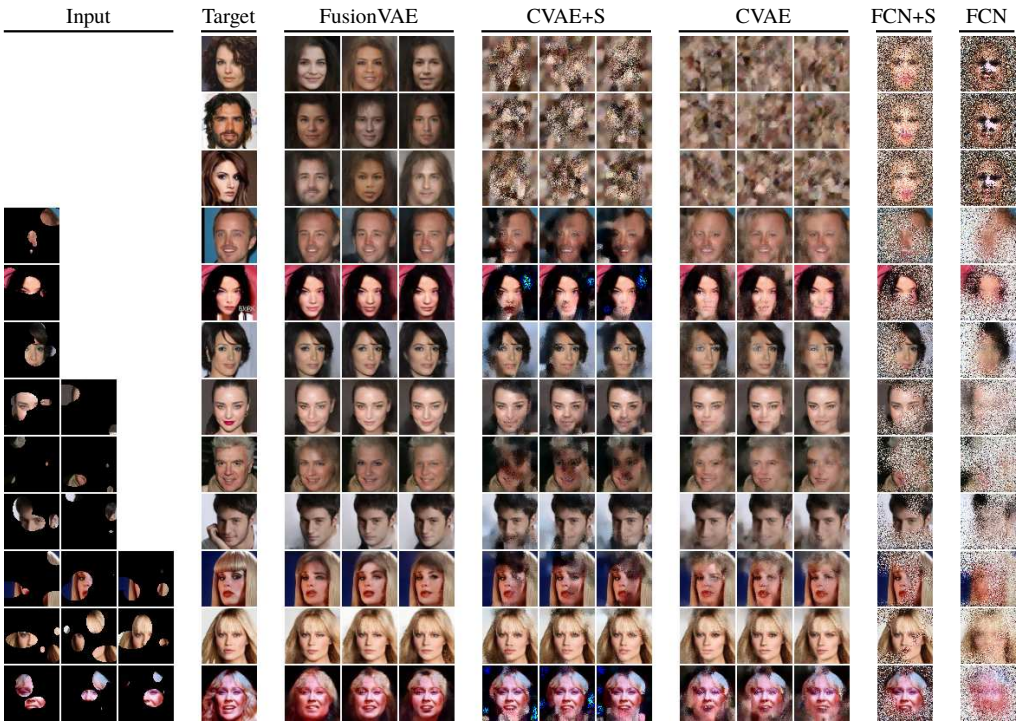


Figure 5.4: Prediction results of the different architectures on the FusionCelebA dataset for zero to three input images.

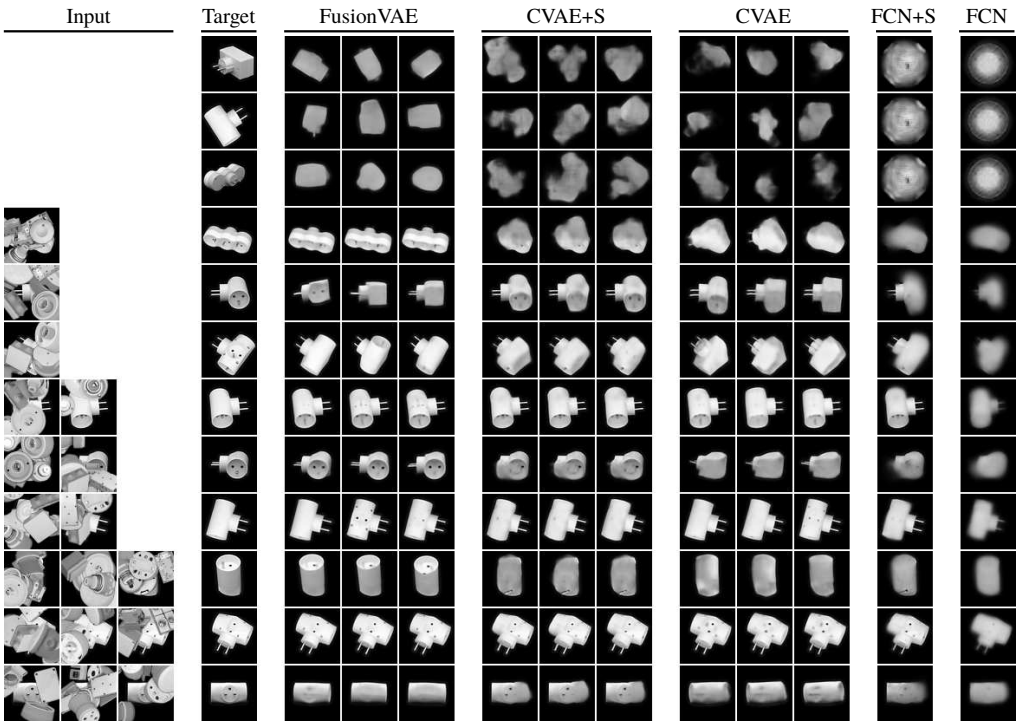


Figure 5.5: Prediction results of the different architectures on the FusionT-LESS dataset for zero to three input images.

5.5.3 Ablation Studies

We conducted ablation studies to show the effect of certain design choices, such as the selection of the approximate posterior distribution and the selection of the aggregation method used within the architecture of FusionVAE. All experiments are run on the FusionCelebA dataset.

Table 5.5 compares the performance of our FusionVAE for two different approximate posterior distributions q_ϕ. The approximate posterior we selected for our FusionVAE $q_\phi(\boldsymbol{y})$, depends only on the given target image $\boldsymbol{y}$. It performs slightly better on average compared to the same method using a posterior $q_\phi(\boldsymbol{x}, \boldsymbol{y})$ that is computed based on the input images $\boldsymbol{x}$ as well as the target image $\boldsymbol{y}$. However, the latter approach is superior when fusing two or three input images.

	NLL in 10^{-2} BPD					MSE_{min} in 10^{-2}				
	0	1	2	3	avg	0	1	2	3	avg
$q_\phi(\boldsymbol{x}, \boldsymbol{y})$	309.2	246.8	**211.1**	**180.5**	237.0	**5.04**	1.50	0.95	**0.66**	2.04
$q_\phi(\boldsymbol{y})$	**248.1**	**227.6**	231.2	228.7	**233.9**	5.11	**0.88**	**0.86**	0.84	**1.93**

Table 5.5: Ablation study for the approximate posterior distribution of FusionVAE, evaluated on the FusionCelebA dataset. The best results are printed in bold.

Table 5.6 shows the performance of different aggregation methods which are applied to create the prior distributions p_l of every latent group. In our FusionVAE, the prior is created by fusing the input image features f_{lx} using max aggregation (MaxAgg) and adding them to the decoded features of the same latent group before applying a 2D convolution. We abbreviate that method with MaxAggAdd.

	NLL in 10^{-2} BPD					MSE_{min} in 10^{-2}				
	0	1	2	3	avg	0	1	2	3	avg
MaxAggAdd	**248.1**	227.6	231.2	228.7	233.9	5.11	**0.88**	0.86	0.84	**1.93**
MeanAggAdd	270.9	**223.3**	**216.4**	214.4	231.3	5.41	1.00	**0.79**	**0.70**	1.98
BayAggAdd	970.7	294.0	291.4	291.5	462.6	6.03	5.17	5.10	5.12	5.36
MaxAggAll	249.6	236.0	223.9	**212.7**	**230.6**	6.15	2.82	1.84	1.30	3.03
MeanAggAll	252.7	235.2	222.2	213.5	230.9	6.19	2.39	1.52	1.13	2.81
BayAggAll	255.6	568.3	414.7	1376.9	653.3	**5.10**	4.24	1.96	1.39	3.18

Table 5.6: Ablation study for different aggregation methods within the FusionVAE architecture, evaluated on the FusionCelebA dataset. The best results are printed in bold.

In addition to MaxAgg, we examined mean aggregation (MeanAgg) and Bayesian aggregation (BayAgg) [Vol+20] for comparison. Please refer to section 2.4.2 for more details about these aggregation methods.

For each aggregation principle, we have tried two different versions:

1. Aggregation of the input image features $\boldsymbol{f}_{lx}$ adding the corresponding information from the decoder in a pixel-wise manner (denoted by suffix Add).
2. Directly aggregating all features, i.e. both input image features $\boldsymbol{f}_{lx}$ and decoder features (denoted by suffix All).

For creating the prior p_i when using BayAgg, we moved the 2D convolutions before the aggregation in order to create the parameter vectors $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ of a latent Gaussian distribution. Unlike MaxAgg and MeanAgg, BayAgg directly outputs a new Gaussian distribution that does not need to be processed any further by a convolution.

We can see that all variations of mean aggregation and max aggregation are significantly better than Bayesian aggregation. Also, their training procedures are less often impaired due to numeric instabilities. Interestingly, the NLL values of mean aggregation and max aggregation are very similar independent of whether the aggregation is performed on all features or not. However, the MSE_{min} is much better for the aggregation with addition.

Figure 5.6: Prediction results of the different mean and max aggregation methods on the FusionCelebA dataset for zero to three input images.

Since the expressiveness of the metrics is limited, we provide additional visualizations of this ablation study for both aggregation methods with pixel-wise addition (ADD) and when aggregating all features (ALL) in figure 5.6. Even though the NLL is very similar,

the results of the aggregation of all features (MaxAggAll and MeanAggAll) are much more blurry than the results of the aggregation with addition (MaxAggAdd and MeanAggAdd). This is in conformity with the MSE_{min} results. It indicates that the NLL alone is not always the best metric to assess the visual closeness to real faces. When carefully examining the images of the addition aggregations, you could argue that the predictions with zero input images look slightly more realistic for max aggregation while for three input images, mean aggregation seems to be marginally better. This again confirms the validity of the MSE_{min} results even though the NLL results are also in accordance with this comparison.

5.6 Conclusion

In this chapter, we have presented a novel deep hierarchical variational autoencoder for generative image fusion, called FusionVAE. Our approach fuses multiple corrupted input images together with prior knowledge obtained during training. We have created three challenging image fusion benchmarks based on common computer vision datasets. Moreover, we have implemented four standard methods that we have modified to support our tasks. We show that our FusionVAE outperforms all other methods significantly while having a similar number of trainable parameters. The predicted images of our approach look very realistic and incorporate the given input information almost perfectly. During ablation studies, we have revealed the benefits of our design choices regarding the applied aggregation method and the used posterior distribution. In future work, our research could be extended by enabling the fusion of different modalities e.g. by using multiple encoders. Additionally, an explicit uncertainty estimation could be implemented that helps to weight the impact of input information according to its uncertainty. Please refer to section 6.3 for more conclusions and ideas for future work.

Chapter 6

Conclusion and Future Work

Over the last couple of years, deep learning approaches have predominantly overtaken traditional methods for environmental perception tasks of automated systems, including robots and automated vehicles. For further improving perception algorithms, sensor data fusion is an increasingly popular technique which has experienced significant advancements recently. Another trend is multi-task learning which can exploit synergy effects between multiple related tasks, reduce computational complexity, and increase inference speed. However, combining these three trends into a deep data fusion approach for multi-task learning is very challenging. Previous methods have severe limitations e.g. regarding their accuracy in complex environments with many occlusions, regarding their generalization on real-world data, or regarding their runtime. This opens up many promising research directions which are targeted in this dissertation. The following three sections conclude the main findings of this work and present promising avenues for further investigation.

6.1 Deep Temporal Point Cloud Fusion and Multi-task Learning for Automated Driving Perception

Chapter 3 addresses the problems of temporal point cloud fusion and multi-task learning using deep learning techniques for LiDAR point clouds in the automotive domain. As LiDAR point clouds are very large while involving an increased sparsity of points with increased distance, we have studied approaches compressing information from dense areas while retaining information from sparse areas.

In this context, we have presented a novel end-to-end trainable network for simultaneous LiDAR-based 3D object detection and scene flow estimation, called PillarFlowNet. We have proposed to employ two pillar-based feature encoding networks based on PointNets [Qi+17a] to efficiently learn compact feature representations of two consecutively acquired point clouds, coping with the challenging properties of large and sparse LiDAR point clouds. A subsequent fully 2D convolutional backbone network fuses the learned representations, avoiding computationally expensive 3D convolutions.

In comprehensive experiments on the KITTI Object Tracking dataset, extended for scene flow estimation, we have demonstrated the superiority of PillarFlowNet compared to previous work in terms of runtime and accuracy. Through the efficiency of the deployed pillar-based feature encoder network and the 2D convolutional backbone networks, PillarFlowNet is the first real-time capable method for simultaneous 3D object detection and scene flow estimation.

As a potential direction for future work, it would be auspicious to integrate the recent advancements in point cloud encoding, e.g. presented in PointTransformer [Zha+21c], PointMLP [Ma+22], PointNeXt [Qia+22], or PointVector [Den+23]. All these methods have led to significant improvements in diverse scene understanding tasks by introducing self-attention layers, improved training strategies, enhanced local feature aggregation, or effective model scaling strategies.

After completing the main research experiments on the KITTI Object Tracking dataset, a few novel datasets for autonomous driving have become popular, such as nuScenes [Cae+20], Argoverse [Cha+19], Argoverse 2 [Wil+21] and the Waymo Open Dataset [Sun+20]. As all of them provide LiDAR point clouds annotated for 3D object detection and ego-motion, they could be extended for scene flow estimation in the same way we computed scene flow ground truth for the KITTI Object Tracking dataset. Each of these datasets incorporates slightly different challenges in comparison to the KITTI dataset that could be addressed in future research. For example, nuScenes contains data recorded not only in clear weather scenarios with good illumination, but also at nighttime, and in rainy, cloudy, and sunny scenarios [Cae+20]. In contrast, Argoverse provides LiDAR point clouds with a range up to 200 m which is roughly twice as much as in KITTI or nuScenes, enabling to address the limited accuracy of most methods for 3D object detection beyond 50 m [Cha+19]. Also the domain transfer between different LiDAR sensors could be examined.

Our proposed approach relies on a large-scale annotated dataset with labeled scenes for 3D object detection and scene flow estimation. To reduce this dependability on expensive data, future work could employ self-supervised pre-training. Related work has recently shown the benefit of self-supervised pre-training for many applications including scene flow estimation [MOH20; Li+22b] and 3D object detection [Lia+21; Erç+22]. A common pre-training objective for point cloud sequences is a cycle consistency loss that given two consecutive point clouds incorporates two scene flow estimation steps forward and reverse in time to create a cycle ensuring the estimated scene flow is consistent in time. After thoroughly pre-training the network, a much lower amount of annotated data is required for fine-tuning the network. Furthermore, this can significantly improve the accuracy and generalization ability.

Another direction for future work would be the extension of our method to additional modalities, such as radar data or camera data. As each sensor modality has its individual advantages as presented in section 2.1, a combination has the potential to further improve the accuracy for 3D object detection and scene flow estimation. This becomes increasingly

relevant for achieving robustness in scenarios with extreme lighting or adverse weather conditions where a LiDAR sensor alone cannot provide sufficiently reliable information.

As scene flow estimation is a point-wise regression task, an extension of our method to additional point-wise regression tasks might be beneficial in terms of synergy effects. For instance, Baur et al. [Bau+21] recently showed that combining scene flow estimation with motion segmentation, i.e. classifying all points in a scene into static and dynamic points, can improve the scene flow accuracy. Another point-wise regression task interesting to combine with scene flow estimation would be 3D semantic segmentation. Recently, Unal et al. [UVD21] have demonstrated the benefits of 3D object detection for improving 3D semantic segmentation. However, the performance of their 3D object detection deteriorates slightly in this case, opening up the research question of how 3D object detection can benefit from 3D semantic segmentation.

Furthermore, the combination of scene flow estimation with ego-motion prediction could be examined in greater detail. As ego-motion corresponds to the inverse of the static scene flow, there might be synergy effects when combining these two tasks in a sophisticated way. Also, the time horizon could be increased from two LiDAR frames to more in order to fully exploit the available information in previous frames. This has great potential to improve the detection accuracy in difficult scenarios, for instance when objects are highly occluded or when the sensor data is unreliable due to noise, adverse weather, challenging lighting, or data point sparsity.

6.2 Multi-View RGB-D Fusion for 6D Pose Estimation

Chapter 4 investigates fusion strategies for combining RGB-D data from multiple perspectives showcasing the same cluttered scene. Within the scope of this research, we have devised two novel deep learning methods called MV6D and SyMFM6D for multi-view 6D object pose estimation. Both methods combine the depth data from all given perspectives into a single point cloud. We have developed and evaluated different feature extraction and fusion methodologies for combining the RGB data and the point cloud data effectively. Based on a compact feature representation of the scene, both methods employ an instance semantic segmentation and a 3D keypoint detection before computing the 6D poses of all objects in the scene with a least-squares fitting algorithm. Besides, we have explored the challenges posed by object symmetries and introduced a symmetry-aware training procedure based on a novel objective function to effectively tackle these challenges.

To comprehensively evaluate the capabilities and limitations of our methods, we have leveraged challenging real-world data and generated four photorealistic datasets with multi-view RGB-D data featuring complex scenes with large occlusions. Our experimental results demonstrate that MV6D and SyMFM6D surpass the previous state-of-the-art in multi-view 6D pose estimation by a substantial margin on these datasets. In addition, we showcase how our symmetry-aware training procedure enhances the pose estimation

accuracy for both symmetric and non-symmetric objects due to synergy effects. This leads to an outperformance of the previous state-of-the-art in single-view 6D pose estimation and further improves our multi-view pose estimation accuracy. Besides, our experiments clearly indicate that our methods cope with variable camera positioning and imprecisely measured camera poses.

Despite the success in the field of 6D pose estimation, several challenges remain, including the requirement of large amounts of labeled data. To the best of our knowledge, there is no large real-world dataset with RGB-D data and 6D pose estimation annotations depicting strongly cluttered scenes from very distinct perspectives. One way to address this shortcoming would be to record such a dataset. Using a camera mounted on a robot arm allows the acquisition of RGB-D data depicting static scenes from arbitrary perspectives. The camera poses would be easy to obtain due to the known robot arm movement. Each scene needs to be annotated only once with 6D object pose ground truth independent of the number of different views as the 6D poses in static scenes are identical for all camera views if a fixed reference coordinate system is used.

While many different software tools have been proposed to facilitate the labeling process [Xia+18; Kas+19; Jia+23], one alternative approach would be the creating of pre-defined scenes where the object poses are determined in advance. In this case, a human could place objects to pre-defined poses, which are, for example, displayed on augmented reality glasses. Alternatively, a robot manipulator could construct scenes automatically according to pre-defined scene construction plans.

Unfortunately, the creation of object scenes, the annotation, and the development of advanced automated labeling tools involve large amounts of human effort, making the generation of real-world data time-consuming and expensive. Thus, the usage of synthetic data is a common approach to reduce the required amount of real-world data. As presented in section 4.6.1, we have already exploited synthetic data in the field of multi-view 6D pose estimation. However, the domain gap between simulated and real-world data hinders the seamless transfer of models trained on exclusively synthetic data to real-world environments. Especially synthetic depth data differs significantly from data obtained by a real depth camera even when generated by photorealistic rendering engines like Cycles [SH14]. To address this issue, neural network models could be employed either to render synthetic data more realistic or to improve real-world data so that it becomes more similar to synthetic data. This could be achieved, for example, with GANs or diffusion models [HJA20] which are trained on unlabeled data with the objective of eliminating differences between real and synthetic data.

Another promising avenue for potential improvement is to reduce the dependency on annotated data by exploiting unlabeled data. Similar to the automotive domain thematized in section 6.1, self-supervised approaches could be integrated which improve the generalization without expensively annotated datasets. Recent advancements in self-supervised methodologies for 6D object pose estimation are often associated with employing a render-and-compare framework where a synthetic image, rendered from an

initial pose estimate, is compared with the original real image [YYY21; Wan+21c; Lab+22; Hai+23]. For example, Hai et al. [Hai+23] start with pre-training their network with supervised synthetic data before introducing a refinement strategy exploiting consistency of pixel-level optical flow between multiple views of the same scene. Self-supervised strategies like the previously mentioned ones could be employed in addition to other strategies including photorealistic rendering, domain randomization, and architectural improvements in order to further enhance accuracy, robustness, and generalization capability in multi-view 6D pose estimation given a limited amount of annotated data.

Apart from improvements related to data, our work could be advanced by enhancing the way of computing the 6D poses in the last stage of our approaches. So far, both MV6D and SyMFM6D employ a least-squares fitting algorithm which computes the rotation and the translation of an object given the 3D keypoint predictions. However, the keypoint predictions can be erroneous, especially for fully occluded keypoints, and least-squares fitting treats all keypoints equally. Thus, a computed 6D pose can be significantly deteriorated by just a few inaccurately predicted keypoints. To address this issue, a mechanism could be implemented which assigns weights to keypoints based on their certainty in the detection process. This can be achieved, for instance, by utilizing a neural network that predicts keypoints together with their associated uncertainties, which are determined by the amount and quality of the input data relevant to each individual keypoint.

Furthermore, concerning the feature extraction and fusion stage of SyMFM6D, there is room for potential improvements. For example, the point cloud encoding could be enhanced by considering the latest advancements in this field as discussed in section 6.1. However, regardless of the selected approach for point cloud encoding, it is essential to customize this approach to enable the multi-directional fusion of image features at multiple intermediate stages. Furthermore, there are significant differences between LiDAR point clouds employed in chapter 3 and point clouds generated from RGB-D cameras as in chapter 4 that need to be considered. For example, LiDAR data is much more accurate whereas the depth values of RGB-D cameras are very noisy, especially for far distance measurements. Unlike RGB-D camera-based point clouds, LiDAR point clouds are more sparse and they have a very low vertical resolution. Besides, LiDAR measurement points contain reflectivity values whereas each point in a RGB-D point cloud has an RGB value attached to it. Therefore, it is imperative to take into account these different properties of point clouds when designing feature extraction and fusion methods.

Despite the evident speed advancement of SyMFM6D over its predecessor MV6D, as indicated in section 4.9, opportunities for further optimization in terms of efficiency remain. Currently, the memory requirements and the network forward time of SyMFM6D exhibit an almost linear increase with the number of input views. While being acceptable for a low number of views, this relation leads to a waste of resources for a high number of views. In order to address this limitation, the feature extraction and fusion could become more efficient by earlier aggregating and down-sampling the image and point cloud data.

For optimal data compression, while preserving pertinent information, attention-based modules could be employed which prioritize essential features and facilitate the discarding of less relevant data.

6.3 Deep Hierarchical Variational Autoencoding for RGB Image Fusion

Chapter 5 explores novel data fusion techniques which effectively merge multiple imperfect visual data sources, taking into account uncertainties, and leveraging prior knowledge pertaining to the data. This research has resulted in a novel deep hierarchical variational autoencoder called FusionVAE that can serve as a fundamental framework for diverse fusion tasks. FusionVAE is the first generative approach able to generate a wide range of new high-quality image samples conditioned on an arbitrary number of input images, effectively handling uncertainty due to large occlusions, noise, or partial visibility. Our experiments on three challenging datasets specially developed for this purpose demonstrate a significant outperformance of our approach compared to conventional fusion methods. Moreover, we substantiate the benefits of the employed posterior distribution and present the impact of commonly used data aggregation operations.

To further advance this research field, future work could target current drawbacks of our FusionVAE network, such as the requirement for high computational resources which limits the possible resolution of input and output images. So far, there is a trade-off with unfavorable characteristics between computational efficiency and expressiveness of our method: To further improve the quality of the generated images and to enable higher resolutions, a significantly higher number of latent groups is required which vastly increases the number of trainable parameters, the memory requirements, and the computational complexity. Thus, future work has to conceive approaches achieving the previously mentioned advantages without significantly increasing the network size. In this context, Child et al. [Chi21] have recently achieved greater performance of hierarchical VAEs by increasing the depth of their architecture while keeping the number of trainable parameters comparably small. Besides, they introduced gradient skipping and nearest-neighbor upsampling to better exploit convolutional layers while mitigating the problem of posterior collapse. Luhman et al. [LL22; LL23] have further improved hierarchical VAEs with a KL-reweighting strategy, a Gaussian output layer, a classifier-free guidance strategy, and a two-stage setup based on a hierarchical VAE trained on the latent space of a deterministic autoencoder. However, further investigation is required to determine which of the recently proposed strategies are suitable for our hierarchical image fusion approach.

Another promising research direction would be the deeper exploration of mechanisms explicitly considering uncertainty in the input data. So far, we have implemented Bayesian aggregation [Vol+20] as one very general mechanism to incorporate a latent uncertainty estimation process. However, a majority of our experiments have indicated conventional

aggregation methods without uncertainty estimation, such as maximum aggregation, to be superior for our addressed applications. For this reason, future work could focus on incorporating explicit uncertainty estimation procedures which are better customized to our application. In this context, we could consider the various factors contributing to uncertainty in sensor data, including challenging lighting conditions, reflections, fog, dirt on the lens, and noise. One approach to further advance might be the introduction of a detection mechanism particularly designed to identify these types of interferences, for instance, by utilizing neural network modules which are pre-trained in a self-supervised manner for this auxiliary task. Based on such a framework, a weighting of input data according to their uncertainty could be established. Thus, input data components with low uncertainty could obtain a higher impact on the final outcome in comparison to components with high uncertainty.

Moreover, the concept of generative fusion incorporating prior knowledge could be extended to other generative models apart from VAEs. For example, GANs [Goo+14] could be utilized, which have experienced remarkable advancements over the last few years, enabling the generation of high-resolution images with astonishingly authentic and realistic appearance [Gui+23]. Very recently, also diffusion models [Soh+15] achieved this ability and even outperformed GANs in selected tasks [DN21]. However, there are even more types of generative model technologies that might be worth to consider, such as normalizing flows [RM15b], autoregressive models [BDV00], and energy-based models [LeC+06].

In chapter 5, we have focused on visual input data from RGB cameras only. However, it would be interesting to extend this research field to enable the fusion of multiple data modalities, for example, RGB data, depth data, LiDAR data, and radar data. This could be achieved by leveraging individual feature encoding networks, specifically adapted to each data modality. Furthermore, uncertainty estimation modules for each modality could be added along with a selection of other previously mentioned strategies for improving the performance of our method. All in all, enabling the fusion of multiple data modalities would extend the range of possible applications significantly, for example to depth estimation with perturbated input data, or to multi-modal object detection in very challenging scenarios.

Appendix A

Supplementary for Chapter 3

A.1 Analysis of the KITTI Object Tracking Dataset

We have examined the KITTI Object Tracking dataset [GLU12] in terms of various key indicators in order to find an appropriate split between training and test data. Figure A.1 shows the number of LiDAR points for each of the 21 sequences of the KITTI Object Tracking dataset including the number of dynamic points, which are defined as points within ground truth bounding boxes. It becomes evident that only a very small fraction of the total number of LiDAR points belongs to dynamic objects and there is a significant variance in the quantity of total and dynamic points across sequences. For example,

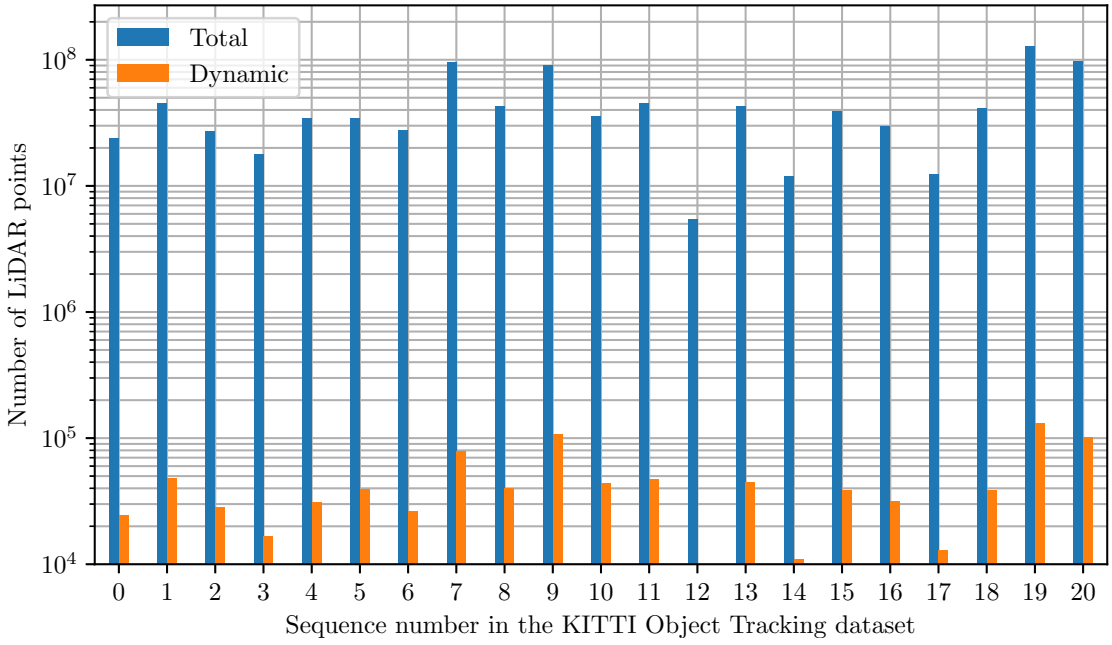


Figure A.1: Number of LiDAR points per sequence of the KITTI Object Tracking dataset. Please note the logarithmic scale required to depict static and dynamic points in the same diagram as on average only 1.9 % of the points are dynamic.

sequence number 19 is the largest sequence with around 127M points in total including around 132k dynamic points. It is characterized by the ego vehicle driving slowly through a commercial street with many pedestrians, cycles, and parking vehicles. In contrast, sequence number 12 is the shortest sequence with around 5M points including around 4k dynamic points. Here, the ego vehicle stands at a red traffic light on an urban road for the entire sequence with just two road users passing by.

As mentioned in section 3.5.1, we have selected the sequences for training and testing in a way to ensure a similar ratio of dynamic points and objects within each set. For the finally chosen split, presented in table 3.1, we provide relevant properties in table A.1.

	Training data	Test data
Total number of sequences	16	5
Total number of frame pairs	5,633	2,349
Average number of points	115,620 ± 6,516	115,848 ± 7,199
Average number of dynamic points	2,226 ± 3, 135	2,147 ± 2,381
Average number of objects per frame	8.2 ± 4.5	8.1 ± 4.8
Average number of cars and vans per frame	4.4 ± 3.5	2.4 ± 3.1

Table A.1: Key numbers for the training and test split of the KITTI Object Tracking dataset. We provide the average numbers μ together with their standard deviation σ with the notation $\mu \pm \sigma$.

Appendix B

Supplementary for Chapter 4

B.1 Network Parameters of MV6D

The network architecture of our MV6D method presented in figure 4.2 consists of 39.2 million trainable parameters. The resulting features of the PSPNets and the PointNet++ have 128 channels each. The DenseFusion module in figure 4.3 concatenates the image features with the point features resulting in a 256-channel feature vector. The shared MLP with average pooling creates a global feature vector with 1024 channels. The image and point features are also processed by shared MLPs which create 256 channel vectors each. For creating the final point-wise features, all vectors are concatenated resulting in 1792 channels ($2 \cdot 128 + 2 \cdot 256 + 1024$) as in PVN3D [He+20].

Table B.1 shows the channel sizes of the MLPs responsible for predicting keypoint offsets, center point offsets, and semantic labels in the second stage of our network architecture in figure 4.4.

MLP	Channel Sizes
Keypoint Detection	1792, 1024, 512, 128, n_{cls}
Center Point Detection	1792, 1024, 512, 128, $3n_{\text{kps}}$
Semantic Segmentation	1792, 1024, 512, 128, 3

Table B.1: MLP channel sizes for the output heads of MV6D. n_{cls} is the number of classes in the used dataset, i.e. $n_{\text{cls}} = 22$ for YCB-Video and SymMovCam and $n_{\text{cls}} = 12$ for FixCam, WiggleCam, and MovingCam. n_{kps} is the number of keypoints per object, which we set to eight as in PVN3D [He+20].

B.2 Network Parameters of SyMFM6D

The network architecture of our SyMFM6D method presented in figure 4.4 consists of 33.9 million trainable parameters. Table B.2 provides an overview of the tensor shapes in

this network architecture. Table B.3 shows the channel sizes of the MLPs responsible for predicting keypoint offsets, center point offsets, and semantic labels in the second stage of our network architecture in figure 4.4. Table B.4 shows the channel sizes of the MLPs employed in our multi-directional fusion modules presented in figure 4.5.

Layer	CNN Tensor Shape	PCN Tensor Shape
1	$H/4$, $W/4$, 64	N_p, 8
2	$H/4$, $W/4$, 64	$N_\mathrm{p}/4$, 64
3	$H/8$, $W/8$, 128	$N_\mathrm{p}/16$, 128
4	$H/8$, $W/8$, 512	$N_\mathrm{p}/64$, 256
5	$H/8$, $W/8$, 1024	$N_\mathrm{p}/256$, 512
6	$H/4$, $W/4$, 256	$N_\mathrm{p}/64$, 256
7	$H/2$, $W/2$, 64	$N_\mathrm{p}/16$, 128
8	$H/2$, $W/2$, 64	$N_\mathrm{p}/4$, 64

Table B.2: Parameters of the network architecture of SyMFM6D. H and W are height and width of the input images, i.e. $H = 480$ and $W = 640$ for all pose estimation datasets we use. N_p is the number of points in the point cloud, i.e. $N_\mathrm{p} = 12{,}800$.

MLP	Channel Sizes
Keypoint Detection	$c_\mathrm{i} + c_\mathrm{p}$, 128, 128, 128, n_cls
Center Point Detection	$c_\mathrm{i} + c_\mathrm{p}$, 128, 128, 128, $3n_\mathrm{kps}$
Semantic Segmentation	$c_\mathrm{i} + c_\mathrm{p}$, 128, 128, 128, 3

Table B.3: MLP channel sizes for the output heads of SyMFM6D. c_i and c_p denote the channel sizes of the image and point features of the feature encoding network output. We set $c_\mathrm{i} = c_\mathrm{p} = 64$. n_cls is the number of classes in the used dataset, i.e. $n_\mathrm{cls} = 22$ for YCB-Video and SymMovCam and $n_\mathrm{cls} = 12$ for FixCam, WiggleCam, and MovingCam. n_kps is the number of keypoints per object, which we set to eight as in FFB6D [He+21].

MLP	Channel Sizes
$\mathrm{MLP_i}$	c_i, c_p
$\mathrm{MLP_{fi}}$	$2c_\mathrm{p}$, c_p
$\mathrm{MLP_p}$	c_p, c_i
$\mathrm{MLP_{fp}}$	$2c_\mathrm{i}$, c_i

Table B.4: MLP channel sizes for the multi-directional fusion modules of SyMFM6D. c_i and c_p denote the channel sizes of the image and point features of the respective layer.

B.3 Qualitative Results on the FixCam and WiggleCam Datasets

Figure B.1 and figure B.2 visualize some 6D pose predictions of FFB6D [He+21], our MV6D method, and our SyMFM6D method in comparison with the ground truth poses on the MV-YCB FixCam dataset and the MV-YCB WiggleCam dataset respectively. While FFB6D uses only the first depicted view, MV6D and SyMFM6D use all three views.

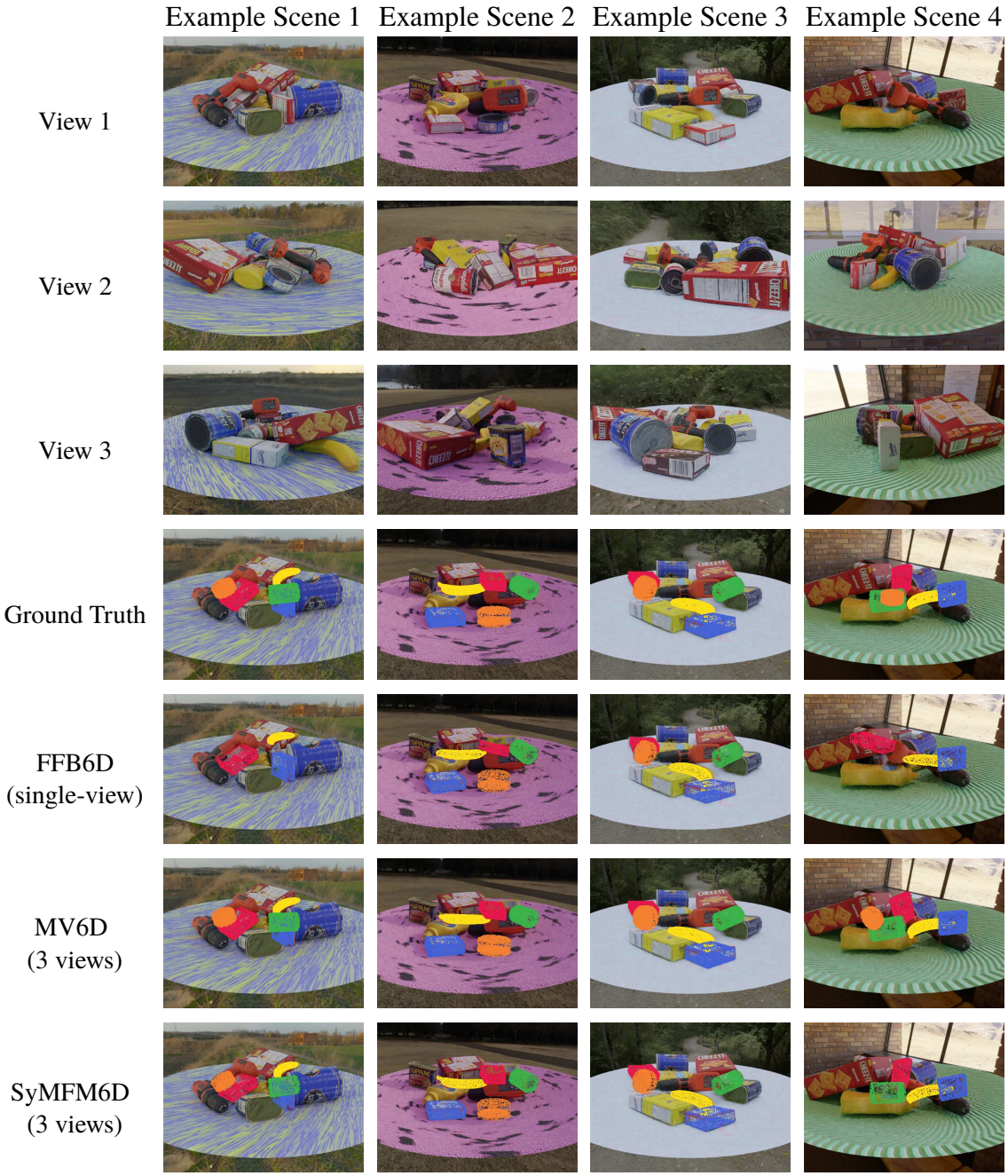


Figure B.1: Visual comparison of predicted poses on different scenes of the MV-YCB FixCam dataset. The seven rows show the first three views which are used for the multi-view methods and four different pose visualizations using the first view as reference. Due to the strong occlusions, only the poses of the five most challenging objects are depicted: banana (yellow), gelatin box (blue), tuna fish can (orange), tomato soup can (green), and pudding box (red).

Figure B.2: Visual comparison of predicted poses on different scenes of the MV-YCB WiggleCam dataset. See caption of figure B.1 for details.

B.4 Quantitative Results on the MV-YCB MovingCam Dataset

Tables B.5 and B.6 show the ADD-S and ADD(-S) AUC results of PVN3D [He+20] and our MV6D network on the MV-YCB MovingCam dataset for each object class individually. In this scenario, we trained MV6D with a varying number of views and evaluated the same model on one to four views.

The high accuracy of MV6D indicates that our architecture copes very well with the dynamic camera setup. MV6D outperforms PVN3D vastly even when using just two views and the accuracy further increases with an increasing number of input images. Furthermore, MV6D almost matches PVN3D when inputting just a single view which is naturally not the designated use case of our method.

Object classes	PVN3D	MV6D (variable number of views)			
	1 view	1 view	2 views	3 views	4 views
Banana	80.9	80.5	94.3	96.7	**97.5**
Cracker box	97.2	96.9	97.9	**98.2**	**98.2**
Gelatin box	78.1	76.3	93.4	96.5	**97.7**
Master chef can	94.5	94.6	98.1	**98.2**	**98.2**
Mustard bottle	91.2	90.2	97.2	97.7	**97.9**
Potted meat can	88.0	86.9	97.2	98.3	**98.4**
Power drill	94.4	93.6	97.1	97.2	**97.8**
Pudding box	86.4	85.4	95.8	97.7	**98.3**
Sugar box	93.1	91.9	97.5	98.0	**98.1**
Tomato soup can	87.7	87.3	97.3	98.2	**98.4**
Tuna fish can	78.6	77.3	93.1	97.5	**97.6**
ALL	88.2	87.4	96.3	97.7	**98.0**

Table B.5: ADD-S AUC results on the MV-YCB MovingCam dataset. MV6D is evaluated for one to four input views on each object class individually. The last row provides the AUC results averaged over all object classes. The best results are printed in bold.

Object classes	PVN3D	MV6D (variable number of views)			
	1 view	1 view	2 views	3 views	4 views
Banana	73.0	73.0	90.1	94.2	**95.9**
Cracker box	96.8	96.3	97.7	98.0	**98.1**
Gelatin box	73.4	72.2	90.4	94.6	**96.4**
Master chef can	91.5	91.4	97.1	**97.7**	**97.7**
Mustard bottle	87.2	86.8	95.5	96.9	**97.2**
Potted meat can	84.9	83.7	96.1	97.6	**97.9**
Power drill	92.6	91.5	96.3	96.8	**97.4**
Pudding box	82.6	82.0	94.3	96.8	**97.6**
Sugar box	90.5	89.5	96.7	97.6	**97.8**
Tomato soup can	83.6	83.3	95.4	97.2	**97.7**
Tuna fish can	71.6	70.8	88.5	94.5	**95.3**
ALL	84.4	83.7	94.4	96.5	**97.2**

Table B.6: ADD(-S) AUC results on the MV-YCB MovingCam dataset. See caption of table B.5 for details.

B.5 Qualitative Results on the MV-YCB MovingCam Dataset

Figure B.3 shows a visualization of predicted 6D poses from PVN3D [He+20] and our MV6D network in comparison to the ground truth on the MV-YCB MovingCam dataset. Figures B.3a to B.3d depict the four possible input views of two example scenes of the dataset. Figure B.3e visualizes the ground truth poses of the five most challenging objects. Figures B.3f and B.3g show the prediction results of PVN3D and MV6D using just the first input view. Figures B.3h to B.3j illustrate the pose predictions using two to four input views. It becomes evident, that the single-view accuracy of MV6D and PVN3D is almost identical. With an increasing number of input images, the accuracy of MV6D increases gradually.

(a) View 1 (b) View 2

(c) View 3 (d) View 4

(e) Ground truth (f) PVN3D (single-view)

(g) MV6D (single-view) (h) MV6D (2 views)

(i) MV6D (3 views) (j) MV6D (4 views)

Figure B.3: Qualitative results of PVN3D and our MV6D method in comparison to the ground truth on the MV-YCB MovingCam dataset. The first two rows depict four input views of two scenes. The single-view method PVN3D is evaluated on the first view, while MV6D is evaluated on one to four views.

Appendix C

Supplementary for Chapter 5

C.1 Statistic Significance of the Results

All experiments in chapter 5 and this supplementary are carefully designed and optimized so that the training procedures are stable and lead to reproducible results. However, the data processing pipelines introduce randomness which lead to non-deterministic training outcomes due to multi-GPU training. Therefore, we have run every experiment three times and reported the results of the best training in section 5.5. In tables C.1 to C.6 we provide the means and variances of the three training runs.

	0	1	2	3	avg
CVAE	17.67 ± 0.10	15.11 ± 0.07	14.19 ± 0.08	13.71 ± 0.07	15.27 ± 0.02
CVAE+S	18.45 ± 0.02	14.64 ± 0.06	**13.22** ± 0.03	**12.32** ± 0.02	14.81 ± 0.03
FusionVAE	**15.91** ± 0.03	**14.13** ± 0.07	13.64 ± 0.09	13.41 ± 0.10	**14.34** ± 0.07

Table C.1: Mean and standard deviation of the FusionMNIST NLL results in 10^{-2} BPD. The best results are printed in bold.

	0	1	2	3	avg
FCN	10.84 ± 0.37	5.96 ± 0.11	6.02 ± 0.18	6.13 ± 0.25	7.38 ± 0.11
FCN+S	6.21 ± 0.65	3.79 ± 0.04	2.64 ± 0.07	1.88 ± 0.08	3.73 ± 0.22
CVAE	3.87 ± 0.03	1.76 ± 0.03	1.09 ± 0.03	0.83 ± 0.02	1.97 ± 0.03
CVAE+S	3.53 ± 0.06	1.77 ± 0.01	1.23 ± 0.04	1.02 ± 0.04	1.96 ± 0.01
FusionVAE	**3.14** ± 0.01	**1.04** ± 0.06	**0.77** ± 0.04	**0.67** ± 0.03	**1.47** ± 0.03

Table C.2: Mean and standard deviation of the FusionMNIST MSE_{min} results in 10^{-2}. The best results are printed in bold.

	0	1	2	3	avg
CVAE	456.9 ± 9.42	289.1 ± 6.53	278.9 ± 4.51	270.3 ± 3.66	324.0 ± 5.17
CVAE+S	487.4 ± 27.45	355.4 ± 60.39	280.7 ± 35.12	230.9 ± 22.59	338.8 ± 22.99
FusionVAE	**251.0** ± 2.06	**222.3** ± 3.71	**226.8** ± 3.16	**224.0** ± 3.36	**231.0** ± 2.05

Table C.3: Mean and standard deviation of the FusionCelebA NLL results in 10^{-2} BPD. The best results are printed in bold.

	0	1	2	3	avg
FCN	13.07 ± 0.87	20.00 ± 3.77	17.87 ± 3.42	15.56 ± 3.08	16.62 ± 2.48
FCN+S	11.94 ± 0.52	18.58 ± 7.02	14.90 ± 6.59	11.19 ± 5.39	14.15 ± 4.65
CVAE	8.70 ± 0.42	6.25 ± 2.16	4.33 ± 1.77	3.02 ± 1.37	5.58 ± 1.21
CVAE+S	9.87 ± 1.10	9.60 ± 3.34	7.30 ± 3.07	5.57 ± 2.64	8.09 ± 2.19
FusionVAE	**5.82** ± 0.52	**1.10** ± 0.16	**0.93** ± 0.06	**0.84** ± 0.03	**2.18** ± 0.17

Table C.4: Mean and standard deviation of the FusionCelebA MSE_{min} results in 10^{-2}. The best results are printed in bold.

	0	1	2	3	avg
CVAE	25.27 ± 0.04	24.00 ± 0.25	22.98 ± 0.24	23.34 ± 0.19	23.90 ± 0.17
CVAE+S	26.12 ± 0.23	25.29 ± 0.27	24.27 ± 0.25	24.15 ± 0.20	24.97 ± 0.16
FusionVAE	**24.32** ± 0.10	**23.09** ± 0.02	**22.25** ± 0.02	**22.90** ± 0.02	**23.15** ± 0.04

Table C.5: Mean and standard deviation of the FusionT-LESS NLL results in 10^{-2} BPD. The best results are printed in bold.

	0	1	2	3	avg
FCN	5.88 ± 0.04	3.32 ± 0.10	2.50 ± 0.09	1.96 ± 0.10	3.43 ± 0.06
FCN+S	8.83 ± 1.05	1.95 ± 0.14	1.28 ± 0.15	0.86 ± 0.14	3.26 ± 0.37
CVAE	5.49 ± 0.11	1.73 ± 0.22	0.95 ± 0.18	0.44 ± 0.07	2.18 ± 0.13
CVAE+S	4.87 ± 0.06	2.98 ± 0.29	2.06 ± 0.19	1.27 ± 0.09	2.81 ± 0.12
FusionVAE	**4.15** ± 0.03	**0.62** ± 0.03	**0.33** ± 0.03	**0.20** ± 0.02	**1.34** ± 0.02

Table C.6: Mean and standard deviation of the FusionT-LESS MSE_{min} results in 10^{-2}. The best results are printed in bold.

C.2 Image Reconstruction Capability

Figures C.1 to C.3 visualize the reconstruction outputs for all our datasets and architectures. For these results, the target image is always given as input. The first three rows of each figure show the reconstruction, when additionally three noisy or partly occluded input images are fed into the network.

The images show that our FusionVAE reconstructs the target images almost perfectly for all three datasets. On FusionMNIST, only the FCN does not manage to reconstruct the target images but shows blurry versions of them. We also see the same behavior for FusionCelebA and FusionT-LESS which underlines the importance of skip connections for this type of network. On FusionCelebA, we see that CVAE+S suffers from numeric instabilities causing colorful artifacts in some images. Omitting the skip connections here avoids that issue. On FusionT-LESS, all baseline methods create more or less blurry versions of the target image when just the target image is given. When inputting the occluded images in addition to the target image, the reconstruction is much better which shows that these networks have over-fitted to the task of removing occluded objects so that they cannot deal well with non-occluded images. In contrast, FusionVAE has the ability to reconstruct non-occluded input images very well.

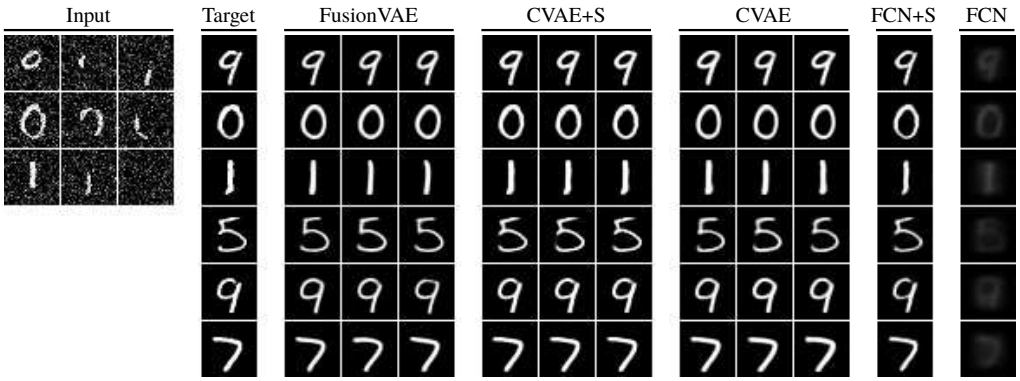


Figure C.1: Reconstruction results of the different architectures on the FusionMNIST dataset. The first three rows show experimental results when inputting the target image together with the three depicted input images. In last three rows, only the corresponding target image is given as input.

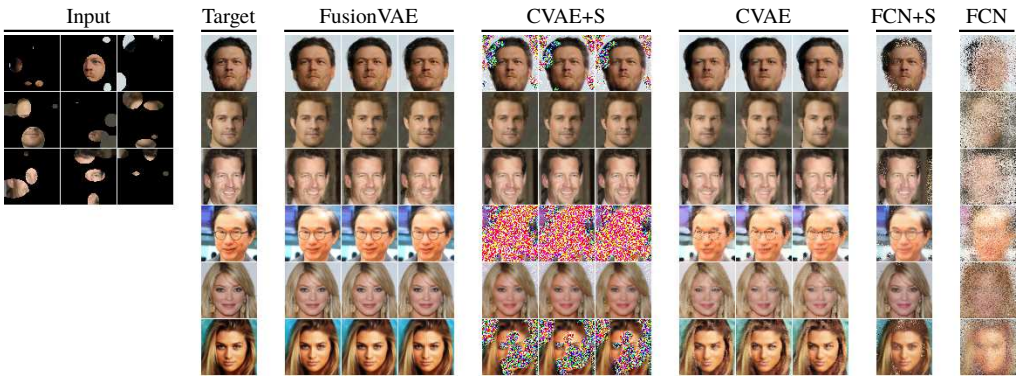


Figure C.2: Reconstruction results on the FusionCelebA dataset.

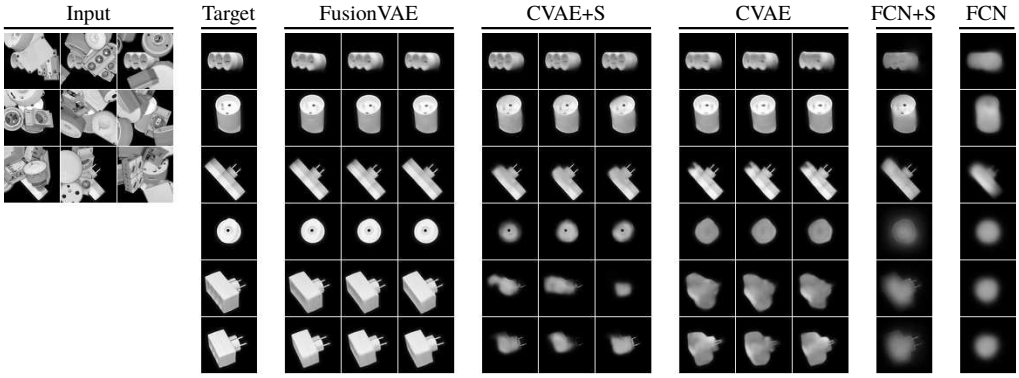


Figure C.3: Reconstruction results on the FusionT-LESS dataset.

Abbreviations

ACD Average Cosine Distance
AEE Average End-point Error
ALR Chair for Autonomous Learning Robots
AP Average Precision
AUC Area Under the Curve
BCAI Bosch Center for Artificial Intelligence
BPD Bits Per Dimension
CNN Convolutional Neural Network
CPU Central Processing Unit
CVAE Conditional Variational Auto-Encoder
DGM Deep Generative Model
DDPM Denoising Diffusion Probabilistic Model
ELBO Evidence Lower Bound
FCN Fully Convolutional Network
FPS Farthest Point Sampling
GAN Generative Adversarial Network
GPT Generative Pre-trained Transformer
GPU Graphics Processing Unit
HOG Histogram of Oriented Gradients
HSV Hue Saturation Value (color space)
KIT Karlsruhe Institute of Technology
KL Kullback-Leibler divergence
LBP Local Binary Patterns
LiDAR Light Detection And Ranging
MLP Multi-Layer Perceptron
MSE Mean Squared Error
NLL Negative Log-Likelihood
PCN Point Cloud Network
PnP Perspective-n-Point (algorithm)

ReLU Rectified Linear Unit
ResNet Residual Network
RGB Red Green Blue (i.e. color image)
RGB-D Red Green Blue Depth (i.e. combined color and depth image)
RNN Recurrent Neural Network
SIFT Scale-Invariant Feature Transform
SL Structured Light
ToF Time of Flight
VAE Variational Auto-Encoder
YCB Yale-CMU-Berkeley (dataset)

List of Tables

List of Figures

Publications

This dissertation contains text, tables, and figures from the following peer-reviewed publications:

- Fabian Duffhauss and Stefan A. Baur. "PillarFlowNet: A Real-time Deep Multitask Network for LiDAR-based 3D Object Detection and Scene Flow Estimation". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 10734–10741.
- Fabian Duffhauss, Ngo Anh Vien, Hanna Ziesche, and Gerhard Neumann. "FusionVAE: A Deep Hierarchical Variational Autoencoder for RGB Image Fusion". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 674–691.
- Fabian Duffhauss, Tobias Demmler, and Gerhard Neumann. "MV6D: Multi-view 6D Pose Estimation on RGB-D Frames Using a Deep Point-wise Voting Network". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 3568–3575.
- Fabian Duffhauss, Sebastian Koch, Hanna Ziesche, Ngo Anh Vien, and Gerhard Neumann. "SyMFM6D: Symmetry-aware Multi-directional Fusion for Multi-View 6D Object Pose Estimation". In: *IEEE Robotics and Automation Letters (RA-L)* 8.9 (2023), pp. 5315–5322.

I declare that I wrote this dissertation by myself based on my research and the aforementioned publications. My advisor Prof. Dr. Gerhard Neuann contributed to all projects with ideas and proofreading of the publications. The work [DB20] is built upon my master thesis [Duf19]. During my doctoral study, I continued the research and further optimized the network architecture, implemented novel baseline methods, and conducted all experiments for the paper [DB20]. Stefan Baur contributed with ideas, code snippets, writing, and illustrations. Dr. Ngo Ahn Vien and Dr. Hanna Ziesche contributed to the mathematical derivations, writing, and ideas of [Duf+22]. They also contributed to ideas and writing of [Duf+23]. Tobias Demmler contributed to the data generation, experiments, and ideas published in [DDN22]. Sebastian Koch contributed to the experiments, ideas, and proofreading of [Duf+23]. [DDN22] and [Duf+23] are also built upon results and insights obtained in the context of the master theses [Bod20; Dem21; Koc22; Kra22] and the bachelor thesis [Bei21] which I supervised during my doctoral study.

Bibliography

[AB22] Simegnew Yihunie Alaba and John E. Ball. "A Survey on Deep-Learning-Based LiDAR 3D Object Detection for Autonomous Driving". In: *Sensors* 22.24 (2022), p. 9577.

[Aba+16] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. "TensorFlow: A System for Large-Scale Machine Learning". In: *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. Nov. 2016, pp. 265–283.

[ABD10] Andrew Adams, Jongmin Baek, and Myers Abraham Davis. "Fast High-Dimensional Filtering Using the Permutohedral Lattice". In: *Computer Graphics Forum*. Vol. 29. 2. Wiley Online Library. 2010, pp. 753–762.

[ACB17] Martin Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34rd International Conference on Machine Learning (ICML)*. Vol. 70. PMLR, 2017, pp. 214–223.

[AEL21] Andrea Asperti, Davide Evangelista, and Elena Loli Piccolomini. "A Survey on Variational Autoencoders from a Green AI Perspective". In: *SN Computer Science* 2.4 (2021), p. 301.

[AG22] Janis Arents and Modris Greitans. "Smart Industrial Robot Control Trends, Challenges and Opportunities within Manufacturing". In: *Applied Sciences* 12.2 (2022), p. 937.

[AHB87] K. Somani Arun, Thomas S. Huang, and Steven D. Blostein. "Least-Squares Fitting of Two 3-D Point Sets". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 5 (1987), pp. 698–700.

[Ala+18] Youssef Alami Mejjati, Christian Richardt, James Tompkin, Darren Cosker, and Kwang In Kim. "Unsupervised Attention-guided Image-to-Image Translation". In: *Advances in Neural Information Processing Systems (NeurIPS)* 31 (2018).

[AMS20] Alaa Eldin Abdelaal, Prateek Mathur, and Septimiu E. Salcudean. "Robotics in Vivo: A Perspective on Human-Robot Interaction in Surgical Robotics". In: *Annual Review of Control, Robotics, and Autonomous Systems* 3 (2020), pp. 221–242.

[AT20] Andrea Asperti and Matteo Trentin. “Balancing Reconstruction Error and Kullback-Leibler Divergence in Variational Autoencoders”. In: *IEEE Access* 8 (2020), pp. 199440–199448.

[Bao+17] Jianmin Bao, Dong Chen, Fang Wen, Houqiang Li, and Gang Hua. “CVAE-GAN: Fine-Grained Image Generation Through Asymmetric Training”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2017, pp. 2745–2754.

[Bau+19] Stefan A. Baur, Frank Moosmann, Sascha Wirges, and Christoph B. Rist. “Real-time 3D LiDAR Flow for Autonomous Vehicles”. In: *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*. June 2019, pp. 1288–1295.

[Bau+21] Stefan Andreas Baur, David Josef Emmerichs, Frank Moosmann, Peter Pinggera, Björn Ommer, and Andreas Geiger. “SLIM: Self-Supervised LiDAR Scene Flow and Motion Segmentation”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 13126–13136.

[BDV00] Yoshua Bengio, Réjean Ducharme, and Pascal Vincent. “A Neural Probabilistic Language Model”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 13 (2000).

[Bec+19] Philipp Becker, Harit Pandya, Gregor Gebhardt, Cheng Zhao, C. James Taylor, and Gerhard Neumann. “Recurrent Kalman Networks: Factorized Inference in High-Dimensional Deep Feature Spaces”. In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. PMLR. 2019, pp. 544–552.

[Beh+19a] Aseem Behl, Despoina Paschalidou, Simon Donne, and Andreas Geiger. “PointFlowNet: Learning Representations for Rigid Motion Estimation From Point Clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 7954–7963.

[Beh+19b] Aseem Behl, Despoina Paschalidou, Simon Donné, and Andreas Geiger. “PointFlowNet: Learning Representations for Rigid Motion Estimation from Point Clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7962–7971.

[Bei21] Jens Beißwenger. “Improving the Sim2Real Performance for 6D Pose Estimation Using Photorealistic RGB-D Images”. Bachelor thesis. Karlsruhe Institute of Technology (KIT), 2021.

[Bel+18] Jorge Beltrán, Carlos Guindel, Francisco Miguel Moreno, Daniel Cruzado, Fernando Garcia, and Arturo De La Escalera. “BirdNet: a 3D Object Detection Framework from LiDAR Information”. In: *Proceedings of the 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE. 2018, pp. 3517–3523.

[BGS16] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. "Importance Weighted Autoencoders". In: *Proceedings of the 4th International Conference on Learning Representations (ICLR)*. 2016.

[BI15] Tolga Birdal and Slobodan Ilic. "Point Pair Features Based Object Detection and Pose Estimation Revisited". In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE. 2015, pp. 527–535.

[Bi21] Xin Bi. *Environmental Perception Technology for Unmanned Systems*. Springer Nature Singapore, 2021.

[BKH16] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. "Layer Normalization". In: *arXiv:1607.06450 [stat.ML]* (2016).

[BL19] Garrick Brazil and Xiaoming Liu. "M3D-RPN: Monocular 3D Region Proposal Network for Object Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 9287–9296.

[BM92] Paul J. Besl and Neil D. McKay. "A Method for Registration of 3-D Shapes". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 14.2 (1992), pp. 239–256.

[Bod20] Lennard Bodden. "Point Cloud Based Object Recognition in RGB-D Video Streams for Robot Manipulation". Master thesis. University of Tübingen, 2020.

[Bon+21a] Andrea Bonci, Pangcheng David Cen Cheng, Marina Indri, Giacomo Nabissi, and Fiorella Sibona. "Human-Robot Perception in Industrial Environments: A Survey". In: *Sensors* 21.5 (2021), p. 1571.

[Bon+21b] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G. Willcocks. "Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2021).

[Bow+16] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. "Generating Sentences from a Continuous Space". In: *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning (CoNLL)*. ACL, 2016, pp. 10–21.

[Bra+14] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. "Learning 6D Object Pose Estimation Using 3D Object Coordinates". In: *Proceedings of the 13th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2014, pp. 536–551.

[Bro+20] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D. Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. "Language Models are Few-Shot Learners". In: *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), pp. 1877–1901.

[BTV06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". In: *Proceedings of the 9th European Conference on Computer Vision (ECCV)*. Springer Berlin Heidelberg. 2006, pp. 404–417.

[Cae+20] 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 the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11621–11631.

[Cai+19] Tian Cai, Mengjun Shen, Huailiang Peng, Lei Jiang, and Qiong Dai. "Improving Transformer with Sequential Context Representations for Abstractive Text Summarization". In: *Proceedings of the 8th CCF International Conference on Natural Language Processing and Chinese Computing (NLPCC)*. Springer. 2019, pp. 512–524.

[Cai+23] Hongxiang Cai, Zeyuan Zhang, Zhenyu Zhou, Ziyin Li, Wenbo Ding, and Jiuhua Zhao. "BEVFusion4D: Learning LiDAR-Camera Fusion Under Bird's-Eye-View via Cross-Modality Guidance and Temporal Aggregation". In: *arXiv:2303.17099 [cs.CV]* (2023).

[Cal+15] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. "The YCB Object and Model Set: Towards Common Benchmarks for Manipulation Research". In: *Proceedings of the 17th International Conference on Advanced Robotics (ICAR)*. IEEE. 2015, pp. 510–517.

[Can86] John Canny. "A Computational Approach to Edge Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 8.6 (1986), pp. 679–698.

[Car+20] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. "End-to-End Object Detection with Transformers". In: *Proceedings of the 16th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2020, pp. 213–229.

[Cha+19] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. "Argoverse: 3D Tracking and Forecasting with Rich Maps". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 8748–8757.

[Cha+22] Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T. Freeman. "MaskGIT: Masked Generative Image Transformer". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 11315–11325.

[Che+16] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. "Monocular 3D Object Detection for Autonomous Driving". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2147–2156.

[Che+17a] Xi Chen, Diederik P. Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. "Variational Lossy Autoencoder". In: *Proceedings of the 5th International Conference on Learning Representations (ICLR)*. 2017.

[Che+17b] 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 (PAMI)* 40.5 (2017), pp. 1259–1272.

[Che+17c] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. "Multi-View 3D Object Detection Network for Autonomous Driving". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1907–1915.

[Che+20a] Jiale Chen, Lijun Zhang, Yi Liu, and Chi Xu. "Survey on 6D Pose Estimation of Rigid Object". In: *Proceedings of the 39th Chinese Control Conference (CCC)*. IEEE. 2020, pp. 7440–7445.

[Che+20b] Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. "Generative Pretraining from Pixels". In: *Proceedings of the 37th International Conference on Machine Learning (ICML)*. PMLR. 2020, pp. 1691–1703.

[Che+21] Qiping Chen, Yinfei Xie, Shifeng Guo, Jie Bai, and Qiang Shu. "Sensing System of Environmental Perception Technologies for Driverless Vehicle: A Review of State of the Art and Challenges". In: *Sensors and Actuators A: Physical* 319 (2021), p. 112566.

[Che95] Yizong Cheng. "Mean Shift, Mode Seeking, and Clustering". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 17.8 (1995), pp. 790–799.

[Chi21] Rewon Child. "Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images". In: *Proceedings of the 9th International Conference on Learning Representations (ICLR)*. 2021.

[Cho17] François Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 1251–1258.

[Cin+21] Lucas Pinheiro Cinelli, Matheus Araújo Marins, Eduardo Antônio Barros da Silva, and Sérgio Lima Netto. *Variational Methods for Machine Learning with Applications to Deep Networks*. Springer, 2021.

[CMS11] Alvaro Collet, Manuel Martinez, and Siddhartha S. Srinivasa. "The MOPED framework: Object recognition and pose estimation for manipulation". In: *The International Journal of Robotics Research (IJRR)* 30.10 (2011), pp. 1284–1306.

[Col+09] Alvaro Collet, Dmitry Berenson, Siddhartha S. Srinivasa, and Dave Ferguson. "Object Recognition and Full Pose Registration from a Single Image for Robotic Manipulation". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2009, pp. 48–55.

[Com18] Blender Online Community. *Blender - A 3D modelling and rendering package*. Blender Foundation. Stichting Blender Foundation, Amsterdam, 2018. URL: `http://www.blender.org`.

[Cor+22] Artur Cordeiro, Luís F. Rocha, Carlos Costa, Pedro Costa, and Manuel F. Silva. "Bin Picking Approaches Based on Deep Learning Techniques: A State-of-the-Art Survey". In: *Proceedings of the IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2022, pp. 110–117.

[Cro+23] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. "Diffusion Models in Vision: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2023).

[CSB16] Zhe Cao, Yaser Sheikh, and Natasha Kholgade Banerjee. "Real-time Scalable 6DOF Pose Estimation for Textureless Objects". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 2441–2448.

[Cui+21] 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* 23.2 (2021), pp. 722–739.

[Cyb89] George Cybenko. "Approximation by Superpositions of a Sigmoidal Function". In: *Mathematics of Control, Signals and Systems* 2.4 (1989), pp. 303–314.

[Dai+17] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. "ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5828–5839.

[DB20] Fabian Duffhauss and Stefan A. Baur. "PillarFlowNet: A Real-time Deep Multitask Network for LiDAR-based 3D Object Detection and Scene Flow Estimation". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 10734–10741.

[DDN22] Fabian Duffhauss, Tobias Demmler, and Gerhard Neumann. "MV6D: Multi-view 6D Pose Estimation on RGB-D Frames Using a Deep Point-wise Voting Network". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pp. 3568–3575.

[De +13] Mark De Deuge, Alastair Quadros, Calvin Hung, and Bertrand Douillard. "Unsupervised Feature Learning for Classification of Outdoor 3D Scans". In: *Proceedings of Australasian Conference on Robotics and Automation (ACRA)*. Vol. 2. 1. University of New South Wales Kensington, Australia. 2013.

[Dem21] Tobias Demmler. "Multi-view 6D Pose Estimation on RGB-D Frames Using a Deep Point-wise Voting Network". Master thesis. University of Freiburg, 2021.

[Den+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009, pp. 248–255.

[Den+23] Xin Deng, WenYu Zhang, Qing Ding, and XinMing Zhang. "PointVector: A Vector Representation In Point Cloud Analysis". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 9455–9465.

[Dev+19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Association for Computational Linguistics, 2019, pp. 4171–4186.

[Dew+16] Ayush Dewan, Tim Caselitz, Gian Diego Tipaldi, and Wolfram Burgard. "Rigid Scene Flow for 3D LiDAR Scans". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2016, pp. 1765–1770.

[Di+21] Yan Di, Fabian Manhardt, Gu Wang, Xiangyang Ji, Nassir Navab, and Federico Tombari. "SO-Pose: Exploiting Self-Occlusion for Direct 6D Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 12396–12405.

[DM19] Yilun Du and Igor Mordatch. "Implicit Generation and Modeling with Energy-Based Models". In: *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[DN21] Prafulla Dhariwal and Alexander Nichol. "Diffusion Models Beat GANs on Image Synthesis". In: vol. 34. 2021, pp. 8780–8794.

[Dos+21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *Proceedings of the 9th International Conference on Learning Representations (ICLR)*. 2021.

[DSB17] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. "Density Estimation Using Real NVP". In: *Proceedings of the 5th International Conference on Learning Representations (ICLR)*. 2017.

[Dua+19] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. "CenterNet: Keypoint Triplets for Object Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6569–6578.

[Duf+22] Fabian Duffhauss, Ngo Anh Vien, Hanna Ziesche, and Gerhard Neumann. "FusionVAE: A Deep Hierarchical Variational Autoencoder for RGB Image Fusion". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 674–691.

[Duf+23] Fabian Duffhauss, Sebastian Koch, Hanna Ziesche, Ngo Anh Vien, and Gerhard Neumann. "SyMFM6D: Symmetry-aware Multi-directional Fusion for Multi-View 6D Object Pose Estimation". In: *IEEE Robotics and Automation Letters (RA-L)* 8.9 (2023), pp. 5315–5322.

[Duf19] Fabian Duffhauss. "Deep Multitask Learning for LiDAR-based 3D Object Detection and Scene Flow Estimation". Master thesis. RWTH Aachen University, 2019.

[DW19] Bin Dai and David P. Wipf. "Diagnosing and Enhancing VAE Models". In: *Proceedings of the 7th International Conference on Learning Representations (ICLR)*. 2019.

[EBD21] Nico Engel, Vasileios Belagiannis, and Klaus Dietmayer. "Point Transformer". In: *IEEE Access* 9 (2021), pp. 134826–134840.

[Eld+97] Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Y. Zeevi. "The Farthest Point Strategy For Progressive Image Sampling". In: *IEEE Transactions on Image Processing (TIP)* 6.9 (1997), pp. 1305–1315.

[Elh+20] Omar Elharrouss, Noor Almaadeed, Somaya Al-Maadeed, and Younes Akbari. "Image Inpainting: A Review". In: *Neural Processing Letters* 51 (2020), pp. 2007–2028.

[Erç+22] Emeç Erçelik, Ekim Yurtsever, Mingyu Liu, Zhijie Yang, Hanzhen Zhang, Pınar Topçam, Maximilian Listl, Yılmaz Kaan Caylı, and Alois Knoll. "3D Object Detection with a Self-supervised Lidar Scene Flow Backbone". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 247–265.

[ERS14] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. "Learning Factored Representations in a Deep Mixture of Experts". In: *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*. 2014.

[Eve+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 (June 2010), pp. 303–338.

[Fan+23] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. "EVA: Exploring the Limits of Masked Visual Representation Learning at Scale". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 19358–19369.

[Fay+20a] Jamil Fayyad, Mohammad A. Jaradat, Dominique Gruyer, and Homayoun Najjaran. "Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review". In: *Sensors* 20.15 (2020), p. 4220.

[Fay+20b] Jamil Fayyad, Mohammad A. Jaradat, Dominique Gruyer, and Homayoun Najjaran. "Deep Learning Sensor Fusion for Autonomous Vehicle Perception and Localization: A Review". In: *Sensors* 20.15 (2020), p. 4220.

[FB81] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Apphcatlons to Image Analysis and Automated Cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[Fel+09] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. "Object Detection with Discriminatively Trained Part-Based Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 32.9 (2009), pp. 1627–1645.

[Fen+21] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Gläser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. "Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges". In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2021), pp. 1341–1360.

[Fer+21] Duarte Fernandes, António Silva, Rafael Névoa, Cláudia Simões, Dibet Gonzalez, Miguel Guevara, Paulo Novais, João Monteiro, and Pedro Melo-Pinto. "Point-Cloud based 3D Object Detection and Classification Methods for Self-Driving Applications: A Survey and Taxonomy". In: *Information Fusion* 68 (2021), pp. 161–191.

[FGH22] Tim Fingscheidt, Hanno Gottschalk, and Sebastian Houben. *Deep Neural Networks and Data for Automated Driving. Robustness, Uncertainty Quantification, and Insights Towards Safety*. Springer Nature Switzerland, 2022.

[Gäh+20] Nils Gählert, Nicolas Jourdan, Marius Cordts, Uwe Franke, and Joachim Denzler. "Cityscapes 3D: Dataset and Benchmark for 9 DoF Vehicle Detection". In: *arXiv:2006.07864 [cs.CV]* (2020).

[Gao+20] Ge Gao, Mikko Lauri, Yulong Wang, Xiaolin Hu, Jianwei Zhang, and Simone Frintrop. "6D Object Pose Regression via Supervised Learning on Point Clouds". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 3643–3649.

[Gao+21] Ge Gao, Mikko Lauri, Xiaolin Hu, Jianwei Zhang, and Simone Frintrop. "CloudAAE: Learning 6D Object Pose Regression with On-line Data Synthesis on Point Clouds". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 11081–11087.

[Gei+13] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. "Vision Meets Robotics: The KITTI Dataset". In: *The International Journal of Robotics Research (IJRR)* 32.11 (2013), pp. 1231–1237.

[Gho+23] Benyamin Ghojogh, Mark Crowley, Fakhri Karray, and Ali Ghodsi. *Elements of Dimensionality Reduction and Manifold Learning*. Springer, 2023.

[Gir+14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2014, pp. 580–587.

[Gir15] Ross Girshick. "Fast R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015, pp. 1440–1448.

[GLU12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2012, pp. 3354–3361.

[Goo+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 27. 2014, pp. 2672–2680.

[GPH19] Kartik Gupta, Lars Petersson, and Richard Hartley. "CullNet: Calibrated and Pose Aware Confidence Scores for Object Pose Estimation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. 2019, pp. 2758–2766.

[GR10] Chunhui Gu and Xiaofeng Ren. "Discriminative Mixture-of-Templates for Viewpoint Classification". In: *Proceedings of the 11th European Conference on Computer Vision (ECCV)*. Springer Berlin Heidelberg. 2010, pp. 408–421.

[Gre+15] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. "DRAW: A Recurrent Neural Network For Image Generation". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. Vol. 37. PMLR, July 2015, pp. 1462–1471.

[Gu+18] Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, Jianfei Cai, and Tsuhan Chen. "Recent Advances in Convolutional Neural Networks". In: *Pattern Recognition* 77 (2018), pp. 354–377.

[Gu+19] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang. "HPLFlowNet: Hierarchical Permutohedral Lattice FlowNet for Scene Flow Estimation on Large-Scale Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 3249–3258.

[Gu+21] Bo Gu, Jianxun Liu, Huiyuan Xiong, Tongtong Li, and Yuelong Pan. "ECPC-ICP: A 6D Vehicle Pose Estimation Method by Fusing the Roadside Lidar Point Cloud and Road Feature". In: *Sensors* 21.10 (2021), p. 3489.

[Gu+22] Jiaqi Gu, Hyoukjun Kwon, Dilin Wang, Wei Ye, Meng Li, Yu-Hsin Chen, Liangzhen Lai, Vikas Chandra, and David Z. Pan. "Multi-Scale High-Resolution Vision Transformer for Semantic Segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 12094–12103.

[Gui+23] Jie Gui, Zhenan Sun, Yonggang Wen, Dacheng Tao, and Jieping Ye. "A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications". In: *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 35.4 (2023), pp. 3313–3332.

[Gul+17] Ishaan Gulrajani, Kundan Kumar, Faruk Ahmed, Adrien Ali Taiga, Francesco Visin, David Vazquez, and Aaron Courville. "PixelVAE: A Latent Variable Model for Natural Images". In: *Proceedings of the 5th International Conference on Learning Representations (ICLR)*. 2017.

[Guo+19] Xiaopeng Guo, Rencan Nie, Jinde Cao, Dongming Zhou, Liye Mei, and Kangjian He. "FuseGAN: Learning to Fuse Multi-Focus Image via Conditional Generative Adversarial Network". In: *IEEE Transactions on Multimedia (TMM)* 21.8 (2019), pp. 1982–1996.

[Guo+20] Yulan Guo, Hanyun Wang, Qingyong Hu, Hao Liu, Li Liu, and Mohammed Bennamoun. "Deep Learning for 3D Point Clouds: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 43.12 (2020), pp. 4338–4364.

[Guo+21a] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. "PCT: Point Cloud Transformer". In: *Computational Visual Media* 7 (2021), pp. 187–199.

[Guo+21b] Xiaoyang Guo, Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. "LIGA-Stereo: Learning LiDAR Geometry Aware Representations for Stereo-based 3D Detector". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 3153–3163.

[Guo+21c] Zhiyang Guo, Yingping Huang, Xing Hu, Hongjian Wei, and Baigan Zhao. "A Survey on Deep Learning Based Approaches for Scene Understanding in Autonomous Driving". In: *Electronics* 10.4 (2021), p. 471.

[Guo+22] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R Martin, Ming-Ming Cheng, and Shi-Min Hu. "Attention mechanisms in computer vision: A survey". In: *Computational Visual Media* 8.3 (2022), pp. 331–368.

[Gup+21] Anunay Gupta, Tanzina Afrin, Evan Scully, and Nita Yodo. "Advances of UAVs toward Future Transportation: The State-of-the-Art, Challenges, and Opportunities". In: *Future Transportation* 1.2 (2021), pp. 326–350.

[Hac+17] 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 Science* IV-1/W1 (2017).

[Hai+23] Yang Hai, Rui Song, Jiaojiao Li, David Ferstl, and Yinlin Hu. "Pseudo Flow Consistency for Self-Supervised 6D Object Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 14075–14085.

[Han+22] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. "A Survey on Vision Transformer". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 45.1 (2022), pp. 87–110.

[HB20] Frederik Hagelskjær and Anders Glent Buch. "PointVoteNet: Accurate Object Detection and 6 DoF Pose Estimation in Point Clouds". In: *Proceedings of the IEEE International Conference on Image Processing (ICIP)*. 2020, pp. 2641–2645.

[HBM20] Tomas Hodan, Daniel Barath, and Jiri Matas. "EPOS: Estimating 6D Pose of Objects with Symmetries". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11703–11712.

[He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[He+20] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. "PVN3D: A Deep Point-wise 3D Keypoints Voting Network for 6DoF Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11632–11641.

[He+21] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. "FFB6D: A Full Flow Bidirectional Fusion Network for 6D Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 3003–3013.

[HFS22] Yinlin Hu, Pascal Fua, and Mathieu Salzmann. "Perspective Flow Aggregation for Data-Limited 6D Object Pose Estimation". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 89–106.

[Hin+11a] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. "Gradient Response Maps for Real-Time Detection of Textureless Objects". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 34.5 (2011), pp. 876–888.

[Hin+11b] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. "Multimodal Templates for Real-Time Detection of Texture-less Objects in Heavily Cluttered Scenes". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2011, pp. 858–865.

[Hin+12] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. "Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes". In: *Proceedings of the 11th Asian Conference on Computer Vision (ACCV)*. Springer Berlin. 2012, pp. 548–562.

[Hin+16] Stefan Hinterstoisser, Vincent Lepetit, Naresh Rajkumar, and Kurt Konolige. "Going Further with Point Pair Features". In: *Proceedings of the 14th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2016, pp. 834–848.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *Advances in Neural Information Processing Systems (NeurIPS)* 33 (2020), pp. 6840–6851.

[Hod+17] Tomáš Hodaň, Pavel Haluza, Štěpán Obdržálek, Jiří Matas, Manolis Lourakis, and Xenophon Zabulis. "T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects". In: *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)* (2017).

[Hod+18] Tomas Hodan, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent-Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, et al. "BOP: Benchmark for 6D Object Pose Estimation". In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 19–34.

[Hod+20] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas. "BOP Challenge 2020 on 6D Object Localization". In: *Proceedings of the 16th European Conference on Computer Vision Workshops (ECCVW)*. Springer Nature Switzerland. 2020, pp. 577–594.

[HSS18] Jie Hu, Li Shen, and Gang Sun. "Squeeze-and-Excitation Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 7132–7141.

[HSS22] Dinh-Cuong Hoang, Johannes A. Stork, and Todor Stoyanov. "Voting and Attention-Based Pose Relation Learning for Object Pose Estimation From 3D Point Clouds". In: *IEEE Robotics and Automation Letters (RA-L)* 7.4 (2022), pp. 8980–8987.

[Hu+20] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. "RandLA-Net: Efficient Semantic Segmentation of Large-Scale Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11108–11117.

[Hu+23] Haotian Hu, Fanyi Wang, Jingwen Su, Laifeng Hu, Tianpeng Feng, Zhaokai Zhang, and Wangzhi Zhang. "EA-BEV: Edge-aware Bird's-Eye-View Projector for 3D Object Detection". In: *arXiv:2303.17895 [cs.CV]* (2023).

[Hua+17] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. "Densely Connected Convolutional Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 4700–4708.

[Hua+20] Jun Huang, Zhuliang Le, Yong Ma, Xiaoguang Mei, and Fan Fan. "A Generative Adversarial Network With Adaptive Constraints for Multi-Focus Image Fusion". In: *Neural Computing and Applications* 32.18 (2020), pp. 15119–15129.

[IS15] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. PMLR. 2015, pp. 448–456.

[ISI17] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. "Globally and Locally Consistent Image Completion". In: *ACM Transactions on Graphics (ToG)* 36.4 (2017), pp. 1–14.

[Jac+91] Robert A. Jacobs, Michael I. Jordan, Steven J. Nowlan, and Geoffrey E. Hinton. "Adaptive Mixtures of Local Experts". In: *Neural Computation* 3.1 (1991), pp. 79–87.

[Jad+15] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. "Spatial Transformer Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)* 28 (2015).

[Jan+23] Thomas Georg Jantos, Mohamed Amin Hamdad, Wolfgang Granig, Stephan Weiss, and Jan Steinbrener. "PoET: Pose Estimation Transformer for Single-View, Multi-Object 6D Pose Estimation". In: *Proceedings of the 7th Conference on Robot Learning (CoRL)*. PMLR. 2023, pp. 1060–1070.

[JCW21] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. "TransGAN: Two Pure Transformers Can Make One Strong GAN, and That Can Scale Up". In: *Advances in Neural Information Processing Systems (NeurIPS)* 34 (2021), pp. 14745–14758.

[Jia+19] Xiaoyue Jiang, Abdenour Hadid, Yanwei Pang, Eric Granger, and Xiaoyi Feng. *Deep Learning in Object Detection and Recognition*. Springer Nature Singapore, 2019.

[Jia+23] ZhiHong Jiang, JinHong Chen, YaMan Jing, Xiao Huang, and Hui Li. "6D Pose Annotation and Pose Estimation Method for Weak-Corner Objects Under Low-Light Conditions". In: *SCIENCE China Technological Sciences* 66.3 (2023), pp. 630–640.

[JKG16] Varun Jampani, Martin Kiefel, and Peter V. Gehler. "Learning Sparse High Dimensional Filters: Image Filtering, Dense CRFs and Bilateral Neural Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4452–4461.

[Jun+20] Hyungjoo Jung, Youngjung Kim, Hyunsung Jang, Namkoo Ha, and Kwanghoon Sohn. "Unsupervised Deep Image Fusion With Structure Tensor Representations". In: *IEEE Transactions on Image Processing (TIP)* 29 (2020), pp. 3845–3858.

[Kal13] Kourosh Kalantar-zadeh. *Sensors – An Introductory Course*. Berlin Heidelberg: Springer Science & Business Media, 2013.

[Kas+19] Roman Kaskman, Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. "HomebrewedDB: RGB-D Dataset for 6D Pose Estimation of 3D Objects". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*. 2019.

[KB15] Diederik P. Kingma and Jimmy Lei Ba. "Adam: A Method for Stochastic Optimization". In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. May 2015.

[Keh+17] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. "SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 1521–1529.

[KGC18] Alex Kendall, Yarin Gal, and Roberto Cipolla. "Multi-task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 7482–7491.

[Kha+22] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. "Transformers in Vision: A Survey". In: *ACM computing surveys (CSUR)* 54.10s (2022), pp. 1–41.

[Kin+16a] Durk P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improved Variational Inference with Inverse Autoregressive Flow". In: *Advances in Neural Information Processing Systems (NeurIPS)* 29 (2016).

[Kin+16b] Durk P. Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. "Improving Variational Inference with Inverse Autoregressive Flow". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. 2016, pp. 4743–4751.

[KJG15] Martin Kiefel, Varun Jampani, and Peter V. Gehler. "Permutohedral Lattice CNNs". In: *Workshop Track Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. 2015.

[KL51] Solomon Kullback and Richard A. Leibler. “On Information and Sufficiency”. In: *The Annals of Mathematical Statistics* 22.1 (1951), pp. 79–86.

[KLA19] Tero Karras, Samuli Laine, and Timo Aila. “A Style-Based Generator Architecture for Generative Adversarial Networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4401–4410.

[KNK21] Oliver Kroemer, Scott Niekum, and George Konidaris. “A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms”. In: *The Journal of Machine Learning Research* 22.1 (2021), pp. 1395–1476.

[Koc22] Sebastian Koch. “Multi-View RGB-D Fusion for 6D Pose Estimation”. Master thesis. University of Tübingen, 2022.

[Köh+14] Rolf Köhler, Christian Schuler, Bernhard Schölkopf, and Stefan Harmeling. “Mask-Specific Inpainting with Deep Neural Networks”. In: *Proceedings of the 36th German Conference on Pattern Recognition (GCPR)*. Springer International Publishing Switzerland. 2014, pp. 523–534.

[Kra22] Lukas Krauch. “Photorealistic Rendering and Dynamics Randomization for Sim2real Transfer for Pushing and Grasping”. Master thesis. University of Tübingen, 2022.

[Kri09] Alex Krizhevsky. *Learning Multiple Layers of Features from Tiny Images*. Technical Report. University of Toronto. 2009.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)* 25 (2012).

[Ku+18] 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 the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–8.

[KW14] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes”. In: *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*. 2014.

[Kyr+21] Maria Kyrarini, Fotios Lygerakis, Akilesh Rajavenkatanarayanan, Christos Sevastopoulos, Harish Ram Nambiappan, Kodur Krishna Chaitanya, Ashwin Ramesh Babu, Joanne Mathew, and Fillia Makedon. “A Survey of Robots in Healthcare”. In: *Technologies* 9.1 (2021), p. 8.

[Lab+20] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. “CosyPose: Consistent Multi-view Multi-object 6D Pose Estimation”. In: *Proceedings of the 16th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2020, pp. 574–591.

[Lab+22] Yann Labbé, Lucas Manuelli, Arsalan Mousavian, Stephen Tyree, Stan Birchfield, Jonathan Tremblay, Justin Carpentier, Mathieu Aubry, Dieter Fox, and Josef Sivic. "MegaPose: 6D Pose Estimation of Novel Objects via Render & Compare". In: *Proceedings of the 6th Conference on Robot Learning (CoRL)*. Vol. 205. PMLR, 2022, pp. 715–725.

[Lai+22] Xin Lai, Jianhui Liu, Li Jiang, Liwei Wang, Hengshuang Zhao, Shu Liu, Xiaojuan Qi, and Jiaya Jia. "Stratified Transformer for 3D Point Cloud Segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 8500–8509.

[Lan+19] Alex H Lang, Sourabh Vora, Holger Caesar, Lubing Zhou, Jiong Yang, and Oscar Beijbom. "PointPillars: Fast Encoders for Object Detection from Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 12697–12705.

[Lar+16] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. "Autoencoding Beyond Pixels Using a Learned Similarity Metric". In: *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. PMLR. 2016, pp. 1558–1566.

[LBH18] Chi Li, Jin Bai, and Gregory D. Hager. "A Unified Framework for Multi-view Multi-class Object Pose Estimation". In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 254–269.

[LD18] Hei Law and Jia Deng. "CornerNet: Detecting Objects as Paired Keypoints". In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 734–750.

[LeC+06] Yann LeCun, Sumit Chopra, Raia Hadsell, M. Ranzato, and Fujie Huang. "A Tutorial on Energy-Based Learning". In: *Predicting Structured Data* 1.0 (2006).

[LeC+98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[LeC98] Yann LeCun. *The MNIST database of handwritten digits*. 1998. URL: `http://yann.lecun.com/exdb/mnist`.

[Lee+22] Kwonjoon Lee, Huiwen Chang, Lu Jiang, Han Zhang, Zhuowen Tu, and Ce Liu. "ViTGAN: Training GANs with Vision Transformers". In: *Proceedings of the 10th International Conference on Learning Representations (ICLR)*. 2022.

[LG17] Jean Lahoud and Bernard Ghanem. "2D-Driven 3D Object Detection in RGB-D Images". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2017, pp. 4622–4630.

[LH17] Ilya Loshchilov and Frank Hutter. “SGDR: Stochastic Gradient Descent with Warm Restarts”. In: *Proceedings of the 5th International Conference on Learning Representations (ICLR)*. 2017.

[Li+17] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. “Generative Face Completion”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 3911–3919.

[Li+18] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. “DeepIM: Deep Iterative Matching for 6D Pose Estimation”. In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 683–698.

[Li+22a] Bing Li, Cheng Zheng, Silvio Giancola, and Bernard Ghanem. “SCTN: Sparse Convolution-Transformer Network for Scene Flow Estimation”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36. 2. 2022, pp. 1254–1262.

[Li+22b] Ruibo Li, Chi Zhang, Guosheng Lin, Zhe Wang, and Chunhua Shen. “RigidFlow: Self-Supervised Scene Flow Learning on Point Clouds by Local Rigidity Prior”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 16959–16968.

[Li+22c] Wenbo Li, Zhe Lin, Kun Zhou, Lu Qi, Yi Wang, and Jiaya Jia. “MAT: Mask-Aware Transformer for Large Hole Image Inpainting”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 10758–10768.

[Li+22d] Yanghao Li, Hanzi Mao, Ross Girshick, and Kaiming He. “Exploring Plain Vision Transformer Backbones for Object Detection”. In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 280–296.

[Li17] Bo Li. “3D Fully Convolutional Network for Vehicle Detection in Point Cloud”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 1513–1518.

[LI20] You Li and Javier Ibanez-Guzman. “LiDAR for Autonomous Driving: The Principles, Challenges, and Trends for Automotive LiDAR and Perception Systems”. In: *IEEE Signal Processing Magazine* 37.4 (2020), pp. 50–61.

[Lia+19] Ming Liang, Bin Yang, Yun Chen, Rui Hu, and Raquel Urtasun. “Multi-Task Multi-Sensor Fusion for 3D Object Detection”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 7345–7353.

[Lia+21] Hanxue Liang, Chenhan Jiang, Dapeng Feng, Xin Chen, Hang Xu, Xiaodan Liang, Wei Zhang, Zhenguo Li, and Luc Van Gool. "Exploring Geometry-aware Contrast and Clustering Harmonization for Self-supervised 3D Object Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 3293–3302.

[Lin+17a] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. "Feature Pyramid Networks for Object Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2117–2125.

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

[Liu+15] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep Learning Face Attributes in the Wild". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Dec. 2015.

[Liu+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 14th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2016, pp. 21–37.

[Liu+17a] Yu Liu, Xun Chen, Juan Cheng, and Hu Peng. "A Medical Image Fusion Method Based on Convolutional Neural Networks". In: *Proceedings of the 20th International Conference on Information Fusion*. IEEE. 2017, pp. 1–7.

[Liu+17b] Yu Liu, Xun Chen, Hu Peng, and Zengfu Wang. "Multi-Focus Image Fusion With a Deep Convolutional Neural Network". In: *Information Fusion* 36 (2017), pp. 191–207.

[Liu+21] Shan Liu, Min Zhang, Pranav Kadam, and C.-C. Jay Kuo. *3D Point Cloud Analysis: Traditional, Deep Learning, and Explainable Machine Learning Methods*. Springer Nature Switzerland, 2021.

[Liu+23] Yang Liu, Yao Zhang, Yixin Wang, Feng Hou, Jin Yuan, Jiang Tian, Yang Zhang, Zhongchao Shi, Jianping Fan, and Zhiqiang He. "A survey of visual transformers". In: *IEEE Transactions on Neural Networks and Learning Systems* (2023).

[LJ21] Jie Liu and Cheolkon Jung. "Facial Image Inpainting Using Attention-based Multi-Level Generative Network". In: *Neurocomputing* 437 (2021), pp. 95–106.

[LL19] Peng Li and Xiangpeng Liu. "Common Sensors in Industrial Robots: A Review". In: *Journal of Physics: Conference Series* 1267.1 (July 2019), p. 012036.

[LL22] Eric Luhman and Troy Luhman. “Optimizing Hierarchical Image VAEs for Sample Quality”. In: *arXiv:2210.10205 [cs.LG]* (2022).

[LL23] Troy Luhman and Eric Luhman. “High Fidelity Image Synthesis With Deep VAEs In Latent Space”. In: *arXiv:2303.13714 [cs.CV]* (2023).

[Low04] David G. Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision (IJCV)* 60.2 (2004), pp. 91–110.

[Low99] David G. Lowe. “Object Recognition from Local Scale-Invariant Features”. In: *Proceedings of the Seventh IEEE International Conference on Computer Vision (ICCV)*. Vol. 2. 1999, pp. 1150–1157.

[LQG19] Xingyu Liu, Charles R. Qi, and Leonidas J. Guibas. “FlowNet3D: Learning Scene Flow in 3D Point Clouds”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019, pp. 529–537.

[LS22] Yan Li and Hualiang Shi. *Advanced Driver Assistance Systems and Autonomous Vehicles: From Fundamentals to Applications*. Springer Nature Singapore, 2022.

[Lu+22] Dening Lu, Qian Xie, Kyle Gao, Linlin Xu, and Jonathan Li. “3DCTN: 3D Convolution-Transformer Network for Point Cloud Classification”. In: *IEEE Transactions on Intelligent Transportation Systems* 23.12 (2022), pp. 24854–24865.

[Lug+22] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. “RePaint: Inpainting Using Denoising Diffusion Probabilistic Models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 11461–11471.

[LW18] Hui Li and Xiao-Jun Wu. “DenseFuse: A Fusion Approach to Infrared and Visible Images”. In: *IEEE Transactions on Image Processing (TIP)* 28.5 (2018), pp. 2614–2623.

[LWD19] Hui Li, Xiao-Jun Wu, and Tariq S. Durrani. “Infrared and Visible Image Fusion with ResNet and Zero-Phase Component Analysis”. In: *Infrared Physics & Technology* 102 (2019), p. 103039.

[LWK18] Hui Li, Xiao-Jun Wu, and Josef Kittler. “Infrared and Visible Image Fusion using a Deep Learning Framework”. In: *Proceedings of the 23rd International Conference on Pattern Recognition (ICPR)*. IEEE. 2018, pp. 2705–2710.

[LWT20] Zechen Liu, Zizhang Wu, and Roland Tóth. "SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2020, pp. 996–997.

[LYB19] Xingyu Liu, Mengyuan Yan, and Jeannette Bohg. "MeteorNet: Deep Learning on Dynamic 3D Point Cloud Sequences". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019, pp. 9246–9255.

[LYU18] Wenjie Luo, Bin Yang, and Raquel Urtasun. "Fast and Furious: Real Time End-to-End 3D Detection, Tracking and Motion Forecasting with a Single Convolutional Net". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 3569–3577.

[LZX16] Bo Li, Tianlei Zhang, and Tian Xia. "Vehicle Detection from 3D Lidar Using Fully Convolutional Network". In: *Robotics: Science and Systems (RSS)*. 2016.

[Ma+19a] Jiayi Ma, Wei Yu, Pengwei Liang, Chang Li, and Junjun Jiang. "FusionGAN: A Generative Adversarial Network for Infrared and Visible Image Fusion". In: *Information Fusion* 48 (2019), pp. 11–26.

[Ma+19b] 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 the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 6851–6860.

[Ma+20a] Jiayi Ma, Pengwei Liang, Wei Yu, Chen Chen, Xiaojie Guo, Jia Wu, and Junjun Jiang. "Infrared and Visible Image Fusion via Detail Preserving Adversarial Learning". In: *Information Fusion* 54 (2020), pp. 85–98.

[Ma+20b] Jiayi Ma, Han Xu, Junjun Jiang, Xiaoguang Mei, and Xiao-Ping Zhang. "DDcGAN: A Dual-Discriminator Conditional Generative Adversarial Network for Multi-Resolution Image Fusion". In: *IEEE Transactions on Image Processing (TIP)* 29 (2020), pp. 4980–4995.

[Ma+22] Xu Ma, Can Qin, Haoxuan You, Haoxi Ran, and Yun Fu. "Rethinking Network Design and Local Geometry in Point Cloud: A Simple Residual MLP Framework". In: *Proceedings of the 10th International Conference on Learning Representations (ICLR)*. 2022.

[Maa+19] Lars Maaløe, Marco Fraccaro, Valentin Liévin, and Ole Winther. "BIVA: A Very Deep Hierarchy of Latent Variables for Generative Modeling". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 32. 2019, pp. 6551–6562.

[Mao+17] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. "Least Squares Generative Adversarial Networks". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2017, pp. 2794–2802.

[Mao+21] Jiageng Mao, Yujing Xue, Minzhe Niu, Haoyue Bai, Jiashi Feng, Xiaodan Liang, Hang Xu, and Chunjing Xu. "Voxel Transformer for 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 3164–3173.

[May+16] Nikolaus Mayer, Eddy Ilg, Philip Hausser, 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 the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 4040–4048.

[MEB16] Oier Mees, Andreas Eitel, and Wolfram Burgard. "Choosing Smartly: Adaptive Multimodal Fusion for Object Detection in Changing Environments". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 151–156.

[MHG15] Moritz Menze, Christian Heipke, and Andreas Geiger. "Joint 3D Estimation of Vehicles and Scene Flow". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2 (2015), pp. 427–434.

[MHG18] Moritz Menze, Christian Heipke, and Andreas Geiger. "Object Scene Flow". In: *ISPRS Journal of Photogrammetry and Remote Sensing (P&RS)* 140 (2018), pp. 60–76.

[Mit12] Harvey B. Mitchell. *Data Fusion: Concepts and Ideas*. Springer Science & Business Media, 2012.

[ML20] Francisco Madrigal and Frédéric Lerasle. "Robust Head Pose Estimation Based on Key Frames for Human-Machine Interaction". In: *EURASIP Journal on Image and Video Processing* 2020.1 (2020), pp. 1–19.

[MMG21] Razvan V. Marinescu, Daniel Moyer, and Polina Golland. "Bayesian Image Reconstruction using Deep Generative Models". In: *Proceedings of the NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. 2021.

[Mni+14] Volodymyr Mnih, Nicolas Heess, Alex Graves, and Koray Kavukcuoglu. "Recurrent Models of Visual Attention". In: *Advances in Neural Information Processing Systems (NeurIPS)* 27 (2014).

[Mo+22] Ningkai Mo, Wanshui Gan, Naoto Yokoya, and Shifeng Chen. "ES6D: A Computation Efficient and Symmetry-Aware 6D Pose Regression Framework". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 6718–6727.

[MO14] Mehdi Mirza and Simon Osindero. “Conditional Generative Adversarial Nets”. In: *arXiv:1411.1784 [cs.LG]* (2014).

[MOH20] Himangi Mittal, Brian Okorn, and David Held. “Just Go With the Flow: Self-Supervised Scene Flow Estimation”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 11177–11185.

[Mor+20] Walter Morales-Alvarez, Oscar Sipele, Régis Léberon, Hadj Hamma Tadjine, and Cristina Olaverri-Monreal. “Automated Driving: A Literature Review of the Take over Request in Conditional Automation”. In: *Electronics* 9.12 (2020), p. 2087.

[Mou+17] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. “3D Bounding Box Estimation Using Deep Learning and Geometry”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 7074–7082.

[MS15] Daniel Maturana and Sebastian Scherer. “VoxNet: A 3D Convolutional Neural Network for Real-Time Object Recognition”. In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 922–928.

[MUS15] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. “Pose Estimation for Augmented Reality: A Hands-On Survey”. In: *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 22.12 (2015), pp. 2633–2651.

[ND21] Alexander Quinn Nichol and Prafulla Dhariwal. “Improved Denoising Diffusion Probabilistic Models”. In: *Proceedings of the 38th International Conference on Machine Learning (ICML)*. PMLR. 2021, pp. 8162–8171.

[OB23] Chahinez Ounoughi and Sadok Ben Yahia. “Data Fusion for ITS: A Systematic Literature Review”. In: *Information Fusion* 89 (2023), pp. 267–291.

[OMS21] Luiz F. P. Oliveira, António P. Moreira, and Manuel F. Silva. “Advances in Agriculture Robotics: A State-of-the-Art Review and Challenges Ahead”. In: *Robotics* 10.2 (2021), p. 52.

[Ott+18] Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. “Scaling Neural Machine Translation”. In: *Proceedings of the Third Conference on Machine Translation: Research Papers*. Association for Computational Linguistics, 2018, pp. 1–9.

[Pai+21] Anshul Paigwar, David Sierra-Gonzalez, Ozgur Erkent, and Christian Laugier. “Frustum-PointPillars: A Multi-Stage Approach for 3D Object Detection using RGB Camera and LiDAR”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 2926–2933.

[Par+21] Gaurav Parmar, Dacheng Li, Kwonjoon Lee, and Zhuowen Tu. "Dual Contradistinctive Generative Autoencoder". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 823–832.

[Pat+16] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. "Context Encoders: Feature Learning by Inpainting". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 2536–2544.

[Pav+17] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G. Derpanis, and Kostas Daniilidis. "6-DoF Object Pose from Semantic Keypoints". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 2011–2018.

[Pen+19] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. "PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 4561–4570.

[Pit+19] Giorgia Pitteri, Michaël Ramamonjisoa, Slobodan Ilic, and Vincent Lepetit. "On Object Symmetries and 6D Pose Estimation from Images". In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE. 2019, pp. 614–622.

[PMR20] Su Pang, Daniel Morris, and Hayder Radha. "CLOCs: Camera-LiDAR Object Candidates Fusion for 3D Object Detection". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pp. 10386–10393.

[Pon+20] Alex D. Pon, Jason Ku, Chengyao Li, and Steven L. Waslander. "Object-Centric Stereo Matching for 3D Object Detection". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 8383–8389.

[Pou+18] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S. Iyengar. "A Survey on Deep Learning: Algorithms, Techniques, and Applications". In: *ACM Computing Surveys (CSUR)* 51.5 (2018), pp. 1–36.

[PP00] Constantine Papageorgiou and Tomaso Poggio. "A Trainable System for Object Detection". In: *International Journal of Computer Vision (IJCV)* 38 (2000), pp. 15–33.

[PPV19] Kiru Park, Timothy Patten, and Markus Vincze. "Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 7668–7677.

[PSB17] K. Ram Prabhakar, V. Sai Srikar, and R. Venkatesh Babu. "DeepFuse: A Deep Unsupervised Approach for Exposure Fusion with Extreme Exposure Image Pairs". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2017, pp. 4714–4722.

[QBZ21] Jia Qin, Huihui Bai, and Yao Zhao. "Multi-Scale Attention Network for Image Inpainting". In: *Computer Vision and Image Understanding* 204 (2021), p. 103155.

[Qi+17a] 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 the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017, pp. 652–660.

[Qi+17b] 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 (NeurIPS)*. Vol. 30. 2017, pp. 5099–5108.

[Qi+18] 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 the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 918–927.

[Qi+19] Charles R. Qi, Or Litany, Kaiming He, and Leonidas J. Guibas. "Deep Hough Voting for 3D Object Detection in Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 9277–9286.

[Qia+22] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. "PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 35. 2022, pp. 23192–23204.

[Qin+21] Zhen Qin, Qingliang Zeng, Yixin Zong, and Fan Xu. "Image Inpainting Based on Deep Learning: A Review". In: *Displays* 69 (2021), p. 102028.

[QWL19] Zengyi Qin, Jinglu Wang, and Yan Lu. "Triangulation Learning Network: From Monocular to Stereo 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7615–7623.

[Rad+19] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. *Language Models are Unsupervised Multitask Learners*. Technical Report. OpenAI. 2019.

[RD06] Edward Rosten and Tom Drummond. “Machine Learning for High-Speed Corner Detection”. In: *Proceedings of the 9th European Conference on Computer Vision (ECCV)*. Springer Berlin Heidelberg. 2006, pp. 430–443.

[Red+16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788.

[Ren+17] 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 (PAMI)* 39.6 (June 2017), pp. 1137–1149.

[Ren+18] Mengye Ren, Andrei Pokrovsky, Bin Yang, and Raquel Urtasun. “SBNet: Sparse Blocks Network for Fast Inference”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 8711–8720.

[RF21] Jesse Richter-Klug and Udo Frese. “Handling Object Symmetries in CNN-based Pose Estimation”. In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2021, pp. 13850–13856.

[Rib+21] Jorge Ribeiro, Rui Lima, Tiago Eckhardt, and Sara Paiva. “Robotic Process Automation and Artificial Intelligence in Industry 4.0 – A Literature Review”. In: *Procedia Computer Science* 181 (2021), pp. 51–58.

[RL17] Mahdi Rad and Vincent Lepetit. “BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects Without Using Depth”. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 3828–3836.

[RM15a] Danilo Rezende and Shakir Mohamed. “Variational Inference with Normalizing Flows”. In: *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. PMLR. 2015, pp. 1530–1538.

[RM15b] Danilo Rezende and Shakir Mohamed. “Variational Inference with Normalizing Flows”. In: *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. PMLR. 2015, pp. 1530–1538.

[RMC16] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. In: *Proceedings of the 4th International Conference on Learning Representations (ICLR)*. 2016.

[RN18] Alec Radford and Karthik Narasimhan. *Improving Language Understanding by Generative Pre-Training*. Technical Report. OpenAI. 2018.

[Rom+22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-Resolution Image Synthesis With Latent Diffusion Models". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 10684–10695.

[Ros+19] Paul L. Rosin, Yu-Kun Lai, Ling Shao, and Yonghuai Liu. *RGB-D Image Analysis and Processing*. Springer Nature Switzerland, 2019.

[Rot+06] Fred Rothganger, Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. "3D Object Modeling and Recognition Using Local Affine-Invariant Image Descriptors and Multi-View Spatial Constraints". In: *International Journal of Computer Vision (IJCV)* 66.3 (2006), pp. 231–259.

[RT17] Dhanesh Ramachandram and Graham W. Taylor. "Deep Multimodal Learning: A Survey on Recent Advances and Trends". In: *IEEE Signal Processing Magazine* 34.6 (2017), pp. 96–108.

[RTG00] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. "The Earth Mover's Distance as a Metric for Image Retrieval". In: *International Journal of Computer Vision (IJCV)* 40 (2000), pp. 99–121.

[Rus+15] 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.

[RZL18] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. "Searching for Activation Functions". In: *Proceedings of the 6th International Conference on Learning Representations (ICLR)*. 2018.

[Sad+19] Hossein Sadeghi, Evgeny Andriyash, Walter Vinci, Lorenzo Buffoni, and Mohammad H. Amin. "PixelVAE++: Improved PixelVAE With Discrete Prior". In: *arXiv:1908.09948 [cs.CV]* (2019).

[Sah+20] Caner Sahin, Guillermo Garcia-Hernando, Juil Sock, and Tae-Kyun Kim. "A Review on Object Pose Recovery: from 3D Bounding Box Detectors to Full 6D Pose Estimators". In: *Image and Vision Computing* 96 (2020), p. 103898.

[Sal+17] Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P. Kingma. "PixelCNN++: Improving the PixelCNN With Discretized Logistic Mixture Likelihood and Other Modifications". In: *Proceedings of the 5th International Conference on Learning Representations (ICLR)*. 2017.

[Sch+18] René Schuster, Christian Bailer, Oliver Wasenmüller, and Didier Stricker. "Combining Stereo Disparity and Optical Flow for Basic Scene Flow". In: *Proceedings of the 5th Commercial Vehicle Technology Symposium (CVT)*. Springer. 2018, pp. 90–101.

[Ser+14] Pierre Sermanet, David Eigen, Xiang Zhang, Michaël Mathieu, Rob Fergus, and Yann LeCun. "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks". In: *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*. 2014.

[SG22] Muhammad Shafiq and Zhaoquan Gu. "Deep Residual Learning for Image Recognition: A Survey". In: *Applied Sciences* 12.18 (2022), p. 8972.

[SH14] Frederik Steinmetz and Gottfried Hofmann. *The Cycles Encyclopedia*. 2014. URL: `http://www.blender.org`.

[Sha+19] Taihua Shao, Yupu Guo, Honghui Chen, and Zepeng Hao. "Transformer-Based Neural Net work for Answer Selection in Question Answering". In: *IEEE Access* 7 (2019), pp. 26146–26156.

[Shi+15] Baoguang Shi, Song Bai, Zhichao Zhou, and Xiang Bai. "DeepPano: Deep Panoramic Representation for 3-D Shape Recognition". In: *IEEE Signal Processing Letters* 22.12 (2015), pp. 2339–2343.

[SK04] Henry Schneiderman and Takeo Kanade. "Object Detection Using the Statistics of Parts". In: *International Journal of Computer Vision (IJCV)* 56.3 (2004), pp. 151–177.

[SK16] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*. 2nd ed. Springer International Publishing Switzerland, 2016.

[SK18] Ozan Sener and Vladlen Koltun. "Multi-Task Learning as Multi-Objective Optimization". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Dec. 2018, pp. 527–538.

[SLX15] Shuran Song, Samuel P. Lichtenberg, and Jianxiong Xiao. "SUN RGB-D: A RGB-D Scene Understanding Benchmark Suite". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 567–576.

[SLY15] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. "Learning Structured Output Representation using Deep Conditional Generative Models". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 28. 2015, pp. 3483–3491.

[Soc+17] Juil Sock, S. Hamidreza Kasaei, Luis Seabra Lopes, and Tae-Kyun Kim. "Multi-view 6D Object Pose Estimation and Camera Motion Planning Using RGBD Images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (ICCVW)*. 2017, pp. 2228–2235.

[Soh+15] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". In: *Proceedings of the 32nd International Conference on Machine Learning (ICML)*. PMLR. 2015, pp. 2256–2265.

[Søn+16] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. "Ladder Variational Autoencoders". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 29. 2016, pp. 3738–3746.

[Son+18a] Yuhang Song, Chao Yang, Zhe Lin, Xiaofeng Liu, Qin Huang, Hao Li, and C.-C. Jay Kuo. "Contextual-based Image Inpainting: Infer, Match, and Translate". In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 3–19.

[Son+18b] Yuhang Song, Chao Yang, Yeji Shen, Peng Wang, Qin Huang, and C.-C. Jay Kuo. "SPG-Net: Segmentation Prediction and Guidance Network for Image Inpainting". In: *Proceedings of the 29th British Machine Vision Conference (BMVC)*. 2018.

[Son22] Dongwon Son. "Grasping as Inference: Reactive Grasping in Heavily Cluttered Environment". In: *IEEE Robotics and Automation Letters (RA-L)* 7.3 (2022), pp. 7193–7200.

[SST23] Rajhans Singh, Ankita Shukla, and Pavan Turaga. "Polynomial Implicit Neural Representations For Large Diverse Datasets". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 2041–2051.

[Str+21] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. "Segmenter: Transformer for Semantic Segmentation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 7262–7272.

[Su+15a] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. "Multi-view Convolutional Neural Networks for 3D Shape Recognition". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2015, pp. 945–953.

[Su+15b] Hao Su, Charles R. Qi, Yangyan Li, and Leonidas J. Guibas. "Render for CNN: Viewpoint Estimation in Images Using CNNs Trained with Rendered 3D Model Views". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015, pp. 2686–2694.

[Su+19] Yongzhi Su, Jason Rambach, Nareg Minaskan, Paul Lesur, Alain Pagani, and Didier Stricker. "Deep Multi-state Object Pose Estimation for Augmented Reality Assembly". In: *Proceedings of the IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE. 2019, pp. 222–227.

[Su+22] Yongzhi Su, Mahdi Saleh, Torben Fetzer, Jason Rambach, Nassir Navab, Benjamin Busam, Didier Stricker, and Federico Tombari. "ZebraPose: Coarse to Fine Surface Encoding for 6DoF Object Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 6738–6748.

[Sun+18] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. "Implicit 3D Orientation Learning for 6D Object Detection from RGB Images". In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 699–715.

[Sun+20] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. "Scalability in Perception for Autonomous Driving: Waymo Open Dataset". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 2446–2454.

[Sun+21] Zhiqing Sun, Shengcao Cao, Yiming Yang, and Kris M. Kitani. "Rethinking Transformer-based Set Prediction for Object Detection". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 3611–3620.

[Sun+23] Martin Sundermeyer, Tomáš Hodaň, Yann Labbe, Gu Wang, Eric Brachmann, Bertram Drost, Carsten Rother, and Jiří Matas. "BOP Challenge 2022 on Detection, Segmentation and Pose Estimation of Specific Rigid Objects". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 2784–2793.

[SX14] Shuran Song and Jianxiong Xiao. "Sliding Shapes for 3D Object Detection in Depth Images". In: *Proceedings of the 13th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2014, pp. 634–651.

[SX16] Shuran Song and Jianxiong Xiao. "Deep Sliding Shapes for Amodal 3D Object Detection in RGB-D Images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 808–816.

[SZ15] Karen Simonyan and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition". In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. 2015.

[Sze+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. "Going Deeper with Convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1–9.

[SZG23] Pourya Shamsolmoali, Masoumeh Zareapoor, and Eric Granger. "TransInpaint: Transformer-based Image Inpainting with Context Adaptation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 849–858.

[Tei+18] Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. "MultiNet: Real-time Joint Semantic Reasoning for Autonomous Driving". In: *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*. June 2018, pp. 1013–1020.

[TM15] Shubham Tulsiani and Jitendra Malik. "Viewpoints and Keypoints". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1510–1519.

[Tom22] Jakub M. Tomczak. *Deep Generative Modeling*. Springer Nature Switzerland, 2022.

[Tri+00] Bill Triggs, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon. "Bundle Adjustment – A Modern Synthesis". In: *Vision Algorithms: Theory and Practice*. Springer. 2000, pp. 298–372.

[TSD10] Federico Tombari, Samuele Salti, and Luigi Di Stefano. "Unique Signatures of Histograms for Local Surface Description". In: *Proceedings of the 11th European Conference on Computer Vision (ECCV)*. Springer Berlin Heidelberg. 2010, pp. 356–369.

[TSF18] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. "Real-Time Seamless Single Shot 6D Object Pose Prediction". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 292–301.

[Tu+19] Zhigang Tu, Wei Xie, Dejun Zhang, Ronald Poppe, Remco C. Veltkamp, Baoxin Li, and Junsong Yuan. "A Survey of Variational and CNN-based Optical Flow Techniques". In: *Signal Processing: Image Communication* 72 (2019), pp. 9–24.

[Tyr+22] Stephen Tyree, Jonathan Tremblay, Thang To, Jia Cheng, Terry Mosier, Jeffrey Smith, and Stan Birchfield. "6-DoF Pose Estimation of Household Objects for Robotic Manipulation: An Accessible Dataset and Benchmark". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022.

[UE18] Arash K. Ushani and Ryan M. Eustice. "Feature Learning for Scene Flow Estimation from LIDAR". In: *Proceedings of the 2nd Annual Conference on Robot Learning (CoRL)*. Oct. 2018, pp. 283–292.

[Ush+17] Arash K. Ushani, Ryan W. Wolcott, Jeffrey M. Walls, and Ryan M. Eustice. "A learning approach for real-time temporal scene flow estimation from LIDAR data". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. June 2017, pp. 5666–5673.

[UVD21] Ozan Unal, Luc Van Gool, and Dengxin Dai. "Improving Point Cloud Semantic Segmentation by Learning 3D Object Detection". In: *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2021, pp. 2950–2959.

[Uy+19] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. "Revisiting Point Cloud Classification: A New Benchmark Dataset and Classification Model on Real-World Data". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 1588–1597.

[Vah+18] Arash Vahdat, William Macready, Zhengbing Bian, Amir Khoshaman, and Evgeny Andriyash. "DVAE++: Discrete Variational Autoencoders with Overlapping Transformations". In: *Proceedings of the 35th International Conference on Machine Learning (ICML)*. PMLR. 2018, pp. 5035–5044.

[Val+17] Abhinav Valada, Johan Vertens, Ankit Dhall, and Wolfram Burgard. "AdapNet: Adaptive Semantic Segmentation in Adverse Environmental Conditions". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 4644–4651.

[Van+21] Simon Vandenhende, Stamatios Georgoulis, Wouter Van Gansbeke, Marc Proesmans, Dengxin Dai, and Luc Van Gool. "Multi-Task Learning for Dense Prediction Tasks: A Survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 44.7 (2021), pp. 3614–3633.

[Var+21] Jorge Vargas, Suleiman Alsweiss, Onur Toker, Rahul Razdan, and Joshua Santos. "An Overview of Autonomous Vehicles Sensors and Their Vulnerability to Weather Conditions". In: *Sensors* 21.16 (2021), p. 5397.

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. "Attention is all you need". In: *Advances in Neural Information Processing Systems (NeurIPS)* 30 (2017).

[Ved+99] Sundar Vedula, Simon Baker, Peter Rander, Robert Collins, and Takeo Kanade. "Three-Dimensional Scene Flow". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision (ICCV)*. Vol. 2. IEEE. 1999, pp. 722–729.

[Vid+18] Joel Vidal, Chyi-Yeu Lin, Xavier Lladó, and Robert Martí. "A Method for 6D Pose Estimation of Free-Form Rigid Objects Using Point Pair Features on Range Data". In: *Sensors* 18.8 (2018), p. 2678.

[VJ01] Paul Viola and Michael Jones. "Rapid Object Detection using a Boosted Cascade of Simple Features". In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. IEEE. 2001, pp. 511–518.

[VK20] Arash Vahdat and Jan Kautz. "NVAE: A Deep Hierarchical Variational Autoencoder". In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 33. 2020, pp. 19667–19679.

[VKK16] Aäron Van Den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. "Pixel Recurrent Neural Networks". In: *Proceedings of the 33rd International Conference on Machine Learning (ICML)*. PMLR. 2016, pp. 1747–1756.

[Vol+20] Michael Volpp, Fabian Flürenbrock, Lukas Grossberger, Christian Daniel, and Gerhard Neumann. "Bayesian Context Aggregation for Neural Processes". In: *Proceedings of the 8th International Conference on Learning Representations (ICLR)*. 2020.

[VSM18] Victor Vaquero, Alberto Sanfeliu, and Francesc Moreno-Noguer. "Deep Lidar CNN to Understand the Dynamics of Moving Vehicles". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. May 2018, pp. 1–6.

[Wad+20] Kentaro Wada, Edgar Sucar, Stephen James, Daniel Lenton, and Andrew J. Davison. "MoreFusion: Multi-object Reasoning for 6D Pose Estimation from Volumetric Fusion". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020, pp. 14540–14549.

[Wal+16] Jacob Walker, Carl Doersch, Abhinav Gupta, and Martial Hebert. "An Uncertain Future: Forecasting from Static Images using Variational Autoencoders". In: *Proceedings of the 14th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2016, pp. 835–851.

[Wan+17] Kunfeng Wang, Chao Gou, Yanjie Duan, Yilun Lin, Xinhu Zheng, and Fei-Yue Wang. "Generative Adversarial Networks: Introduction and Outlook". In: *IEEE/CAA Journal of Automatica Sinica* 4.4 (2017), pp. 588–598.

[Wan+18a] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. "Deep Parametric Continuous Convolutional Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 2589–2597.

[Wan+18b] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. "Non-local Neural Networks". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 7794–7803.

[Wan+19] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. "DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 3343–3352.

[Wan+20] Weiming Wang, Wenhai Liu, Jie Hu, Yi Fang, Quanquan Shao, and Jin Qi. "GraspFusionNet: A Two-Stage Multi-Parameter Grasp Detection Network Based on RGB–XYZ Fusion in Dense Clutter". In: *Machine Vision and Applications* 31.7 (2020), pp. 1–19.

[Wan+21a] Biyao Wang, Yi Han, Di Tian, and Tian Guan. "Sensor-Based Environmental Perception Technology for Intelligent Vehicles". In: *Journal of Sensors* 2021 (2021), pp. 1–14.

[Wan+21b] Changshuo Wang, Chen Wang, Weijun Li, and Haining Wang. "A Brief Survey on RGB-D Semantic Segmentation Using Deep Learning". In: *Displays* 70 (2021), p. 102080.

[Wan+21c] Gu Wang, Fabian Manhardt, Xingyu Liu, Xiangyang Ji, and Federico Tombari. "Occlusion-Aware Self-Supervised Monocular 6D Object Pose Estimation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2021).

[Wan+21d] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. "GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 16611–16621.

[Wan+23] Yingjie Wang, Qiuyu Mao, Hanqi Zhu, Jiajun Deng, Yu Zhang, Jianmin Ji, Houqiang Li, and Yanyong Zhang. "Multi-Modal 3D Object Detection in Autonomous Driving: A Survey". In: *International Journal of Computer Vision (IJCV)* (2023), pp. 1–31.

[WFU15] Shenlong Wang, Sanja Fidler, and Raquel Urtasun. "Holistic 3D Scene Understanding from a Single Geo-tagged Image". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015, pp. 3964–3972.

[Wil+21] Benjamin Wilson et al. "Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting". In: *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks)*. 2021.

[WJ19] Zhixin Wang and Kui Jia. "Frustum ConvNet: Sliding Frustums to Aggregate Local Point-Wise Features for Amodal 3D Object Detection". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 1742–1749.

[WSH20] Xiongwei Wu, Doyen Sahoo, and Steven C. H. Hoi. "Recent Advances in Deep Learning for Object Detection". In: *Neurocomputing* 396 (2020), pp. 39–64.

[Wu+15] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. "3D ShapeNets: A Deep Representation for Volumetric Shapes". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 1912–1920.

[Wu+20] Wenxuan Wu, Zhi Yuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin. "PointPWC-Net: Cost Volume on Point Clouds for (Self-)Supervised Scene Flow Estimation". In: *Proceedings of the 16th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2020, pp. 88–107.

[Wu+22] Yangzheng Wu, Mohsen Zand, Ali Etemad, and Michael Greenspan. "Vote from the Center: 6 DoF Pose Estimation in RGB-D Images by Radial Keypoint Voting". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 335–352.

[Wu+23] Hai Wu, Chenglu Wen, Shaoshuai Shi, Xin Li, and Cheng Wang. "Virtual Sparse Convolution for Multimodal 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 21653–21662.

[XAJ18] Danfei Xu, Dragomir Anguelov, and Ashesh Jain. "PointFusion: Deep Sensor Fusion for 3D Bounding Box Estimation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 244–253.

[Xia+10] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. "SUN Database: Large-scale Scene Recognition from Abbey to Zoo". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2010, pp. 3485–3492.

[Xia+18] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. "PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes". In: *Robotics: Science and Systems (RSS)* (2018).

[Xia+22] Ziwei Xia, Zhen Deng, Bin Fang, Yiyong Yang, and Fuchun Sun. "A Review on Sensory Perception for Dexterous Robotic Manipulation". In: *International Journal of Advanced Robotic Systems (IJARS)* 19.2 (2022), p. 17298806221095974.

[Xie+19] Chaohao Xie, Shaohui Liu, Chao Li, Ming-Ming Cheng, Wangmeng Zuo, Xiao Liu, Shilei Wen, and Errui Ding. "Image Inpainting with Learnable Bidirectional Attention Maps". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 8858–8867.

[Xie+21] Qian Xie, Yu-Kun Lai, Jing Wu, Zhoutao Wang, Dening Lu, Mingqiang Wei, and Jun Wang. "VENet: Voting Enhancement Network for 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2021, pp. 3712–3721.

[Xie+23] Shaoan Xie, Zhifei Zhang, Zhe Lin, Tobias Hinz, and Kun Zhang. "SmartBrush: Text and Shape Guided Object Inpainting with Diffusion Model". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2023, pp. 22428–22437.

[Xu+18] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. "PAD-Net: Multi-tasks Guided Prediction-and-Distillation Network for Simultaneous Depth Estimation and Scene Parsing". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 675–684.

[Xu+19] Han Xu, Pengwei Liang, Wei Yu, Junjun Jiang, and Jiayi Ma. "Learning a Generative Model for Fusing Infrared and Visible Images via Conditional Generative Adversarial Network with Dual Discriminators". In: *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*. 2019.

[Xu+20a] Chi Xu, Jiale Chen, Mengyang Yao, Jun Zhou, Lijun Zhang, and Yi Liu. "6DoF Pose Estimation of Transparent Object from a Single RGB-D Image". In: *Sensors* 20.23 (2020), p. 6790.

[Xu+20b] Han Xu, Jiayi Ma, Zhuliang Le, Junjun Jiang, and Xiaojie Guo. "FusionDN: A Unified Densely Connected Network for Image Fusion". In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. Vol. 34. Apr. 2020, pp. 12484–12491.

[Yan+17] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. "High-Resolution Image Inpainting Using Multi-Scale Neural Patch Synthesis". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 6721–6729.

[Yan+18] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. "Shift-Net: Image Inpainting via Deep Feature Rearrangement". In: *Proceedings of the 15th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2018, pp. 1–17.

[Yan+21] Jun Yang, Yizhou Gao, Dong Li, and Steven L. Waslander. "ROBI: A Multi-View Dataset for Reflective Objects in Robotic Bin-Picking". In: *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pp. 9788–9795.

[Yan+22] Ling Yang, Zhilong Zhang, Yang Song, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Bin Cui, and Ming-Hsuan Yang. "Diffusion Models: A Comprehensive Survey of Methods and Applications". In: *arXiv:2209.00796 [cs.LG]* (2022).

[Yan20] Xiao Yang. "An Overview of the Attention Mechanisms in Computer Vision". In: *Journal of Physics: Conference Series*. Vol. 1693. 1. IOP Publishing. 2020, p. 012173.

[YCL23] Shiyuan Yang, Xiaodong Chen, and Jing Liao. "Uni-paint: A Unified Framework for Multimodal Image Inpainting with Pretrained Diffusion Model". In: *Proceedings of the 31th ACM International Conference on Multimedia (ACMMM)*. New York, NY, USA: Association for Computing Machinery, 2023.

[Yeo+21] 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), p. 2140.

[Yeu+17] Serena Yeung, Anitha Kannan, Yann Dauphin, and Li Fei-Fei. "Tackling Over-pruning in Variational Autoencoders". In: *ICML 2017 Workshop on Principled Approaches to Deep Learning* (2017).

[YLU18] Bin Yang, Wenjie Luo, and Raquel Urtasun. "PIXOR: Real-time 3D Object Detection from Point Clouds". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 7652–7660.

[YM17] Yuichi Yoshida and Takeru Miyato. "Spectral Norm Regularization for Improving the Generalizability of Deep Learning". In: *arXiv:1705.10941 [stat.ML]* (2017).

[YML18] Yan Yan, Yuxing Mao, and Bo Li. "SECOND: Sparsely Embedded Convolutional Detection". In: *Sensors* 18.10 (Oct. 2018), p. 3337.

[Yoo+20] 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 16th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2020, pp. 720–736.

[Yu+18] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. "Generative Image Inpainting With Contextual Attention". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 5505–5514.

[Yur+20] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. "A Survey of Autonomous Driving: Common Practices and Emerging Technologies". In: *IEEE Access* 8 (2020), pp. 58443–58469.

[YYY21] Zongxin Yang, Xin Yu, and Yi Yang. "DSC-PoseNet: Learning 6DoF Object Pose Estimation via Dual-scale Consistency". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 3907–3916.

[Zam+18] Amir R. Zamir, Alexander Sax, William Shen, Leonidas J. Guibas, Jitendra Malik, and Silvio Savarese. "Taskonomy: Disentangling Task Transfer Learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2018, pp. 3712–3722.

[Zan+16] Pietro Zanuttigh, Giulio Marin, Carlo Dal Mutto, Fabio Dominio, Ludovico Minto, and Guido Maria Cortelazzo. *Time-of-Flight and Structured Light Depth Cameras: Technology and Applications*. Springer International Publishing Switzerland, 2016.

[ZCC19] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. "Pluralistic Image Completion". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 1438–1447.

[Zen+17] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker, Alberto Rodriguez, and Jianxiong Xiao. "Multi-view Self-supervised Deep Learning for 6D Pose Estimation in the Amazon Picking Challenge". In: *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2017, pp. 1386–1383.

[Zen+20a] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, and Huchuan Lu. "High-Resolution Image Inpainting with Iterative Confidence Feedback and Guided Upsampling". In: *Proceedings of the 16th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2020, pp. 1–17.

[Zen+20b] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, and Huchuan Lu. "High-Resolution Image Inpainting with Iterative Confidence Feedback and Guided Upsampling". In: *Proceedings of the 16th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2020, pp. 1–17.

[Zha+17a] Qiang Zhang, Daokui Qu, Fang Xu, and Fengshan Zou. "Robust Robot Grasp Detection in Multimodal Fusion". In: *Proceedings of the MATEC Web of Conferences*. Vol. 139. EDP Sciences. 2017, p. 00060.

[Zha+17b] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. "Pyramid Scene Parsing Network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 2881–2890.

[Zha+19] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. "Self-Attention Generative Adversarial Networks". In: *Proceedings of the 36th International Conference on Machine Learning (ICML)*. PMLR. 2019, pp. 7354–7363.

[Zha+20a] Hao Zhang, Han Xu, Yang Xiao, Xiaojie Guo, and Jiayi Ma. "Rethinking the Image Fusion: A Fast Unified Image Fusion Network based on Proportional Maintenance of Gradient and Intensity". In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence*. Vol. 34. Apr. 2020, pp. 12797–12804.

[Zha+20b] Hongjia Zhang, Junwen Huang, Xin Xu, Qiang Fang, and Yifei Shi. "Symmetry-Aware 6D Object Pose Estimation via Multitask Learning". In: *Complexity* 2020 (Oct. 2020).

[Zha+20c] Xinyan Zhao, Feng Xiao, Haoming Zhong, Jun Yao, and Huanhuan Chen. "Condition Aware and Revise Transformer for Question Answering". In: *Proceedings of the World Wide Web Conference (WWW)*. 2020, pp. 2377–2387.

[Zha+21a] Yifei Zhang, Désiré Sidibé, Olivier Morel, and Fabrice Mériaudeau. "Deep Multimodal Fusion for Semantic Image Segmentation: A Survey". In: *Image and Vision Computing* 105 (2021), p. 104042.

[Zha+21b] Zehan Zhang, Zhidong Liang, Ming Zhang, Xian Zhao, Hao Li, Ming Yang, Wenming Tan, and Shiliang Pu. "RangeLVDet: Boosting 3D Object Detection in LIDAR With Range Image and RGB Image". In: *IEEE Sensors Journal* 22.2 (2021), pp. 1391–1403.

[Zha+21c] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. "Point Transformer". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 16259–16268.

[Zha+22a] Bowen Zhang, Shuyang Gu, Bo Zhang, Jianmin Bao, Dong Chen, Fang Wen, Yong Wang, and Baining Guo. "StyleSwin: Transformer-based GAN for High-resolution Image Generation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 11304–11314.

[Zha+22b] Zhongqun Zhang, Wei Chen, Linfang Zheng, Aleš Leonardis, and Hyung Jin Chang. "Trans6D: Transformer-Based 6D Object Pose Estimation and Refinement". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 112–128.

[Zha+23] Xiaobo Zhang, Donghai Zhai, Tianrui Li, Yuxin Zhou, and Yang Lin. "Image Inpainting Based on Deep Learning: A Review". In: *Information Fusion* 90 (2023), pp. 74–94.

[Zho+19] Peilin Zhong, Yuchen Mo, Chang Xiao, Pengyu Chen, and Changxi Zheng. "Rethinking Generative Mode Coverage: A Pointwise Guaranteed Approach". In: *Advances in Neural Information Processing Systems (NeurIPS)* 32 (2019).

[Zho+22] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. "CenterFormer: Center-based Transformer for 3D Object Detection". In: *Proceedings of the 17th European Conference on Computer Vision (ECCV)*. Springer Nature Switzerland. 2022, pp. 496–513.

[Zhu+22] Yingzhao Zhu, Man Li, Wensheng Yao, and Chunhua Chen. "A Review of 6D Object Pose Estimation". In: *Proceedings of the 10th Joint International Information Technology and Artificial Intelligence Conference (ITAIC)*. Vol. 10. IEEE. 2022, pp. 1647–1655.

[Zie+17] Adam Ziebinski, Rafal Cupek, Damian Grzechca, and Lukas Chruszczyk. "Review of Advanced Driver Assistance Systems (ADAS)". In: *AIP Conference Proceedings*. Vol. 1906. 1. AIP Publishing. 2017.

[Zou+23] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. "Object Detection in 20 Years: A Survey". In: *Proceedings of the IEEE* (2023).

[ZSI19] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. "DPOD: 6D Pose Object Detector and Refiner". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2019, pp. 1941–1950.

[ZT18] Yin Zhou and Oncel Tuzel. "VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4490–4499.

[ZWC22] Jiahao Zeng, Decheng Wang, and Peng Chen. "A Survey on Transformers for Point Cloud Processing: An Updated Overview". In: *IEEE Access* 10 (2022), pp. 86510–86527.

[ZY21] Yu Zhang and Qiang Yang. "A Survey on Multi-Task Learning". In: *IEEE Transactions on Knowledge and Data Engineering (TKDE)* 34.12 (2021), pp. 5586–5609.