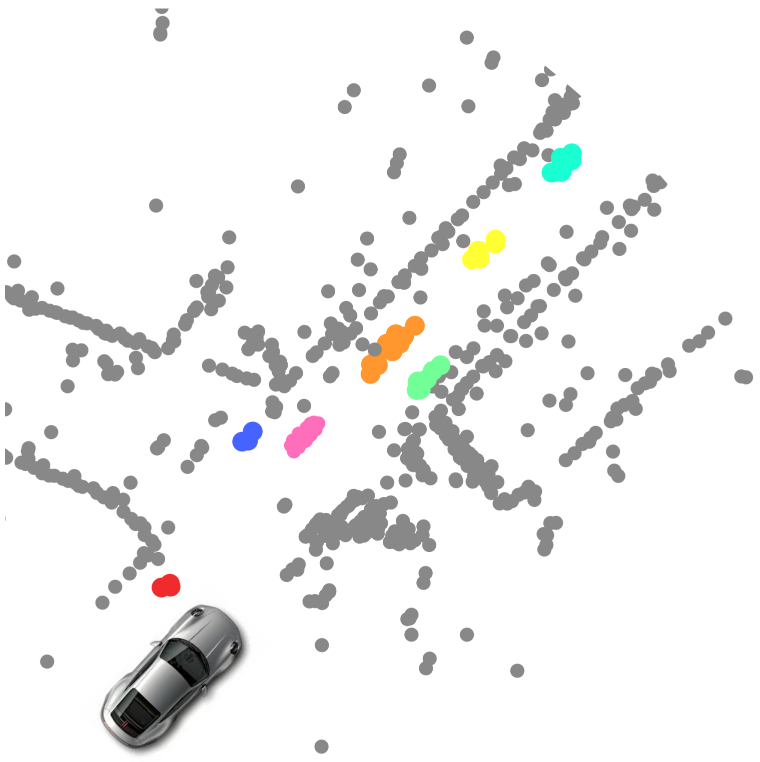
Dissertation
zur Erlangung des Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)
der Agrar-, Ernährungs- und Ingenieurwissenschaftlichen Fakultät
der Rheinischen Friedrich-Wilhelms-Universität Bonn
Institut für Geodäsie und Geoinformation

Radar-Based Scene Understanding for Autonomous Vehicles

von

Matthias Zeller

aus
Göppingen, Deutschland



Bonn 2025

Referent:
Prof. Dr. Cyrill Stachniss

1. Korreferent:
Prof. Dr. rer. nat. Markus Enzweiler

2. Korreferent:
PD Dr. rer. nat. Lasse Klingbeil

Tag der mündlichen Prüfung: 13. Mai 2025

Erscheinungsjahr: 2025

Angefertigt mit Genehmigung der Agrar-, Ernährungs- und Ingenieurwissenschaftlichen Fakultät der Universität Bonn

Zusammenfassung

Autonome Fahrzeuge besitzen das Potenzial, den Verkehr grundlegend zu verändern. Sie werden Unfälle aufgrund menschlichen Versagens reduzieren und Mobilität für alle zugänglich machen. Reale Umgebungen stellen aufgrund von wechselnden Licht- und Wetterverhältnissen sowie der komplexen Interaktion der Verkehrsteilnehmer eine große Herausforderung für autonome Fahrzeuge dar. Die Fähigkeit unterschiedlichste Szenarien handhaben zu können ist entscheidend, um Unfälle zu vermeiden und Verkehrsteilnehmer bestmöglich zu schützen. Eine zuverlässige Wahrnehmung der Umgebung ist dabei eine wesentliche Grundlage für sichere autonome Fahrfunktionen.

Intelligente Wahrnehmungssysteme autonomer Fahrzeuge umfassen verschiedene Sensoren wie Kameras, LiDAR-Scanner und Radarsensoren, um die Stärken der verschiedenen Modalitäten zu kombinieren. LiDAR-Scanner und Kameras stoßen bei widrigen Wetterbedingungen wie Regen, Nebel oder Schnee an ihre Grenzen. Radarsensoren hingegen behalten auch unter diesen Bedingungen ihre Funktionalität und sind daher für eine verlässliche Wahrnehmung der Umgebung und somit eine sichere Mobilität entscheidend. Im Gegensatz zu hochauflösenden Lidar-Scannern und Kameras liefern sie jedoch spärliche Punktwolken und werden aufgrund von Mehrwegeausbreitung und Interferenzen erheblich durch Rauschen beeinträchtigt. Daher werden spezielle Algorithmen benötigt, die mit spärlichen und verrauschten Radar-Punktwolken umgehen können, um die wertvollen Informationen, die Radarsensoren liefern, effektiv zu nutzen. Diese umfassen neben der Position auch die Doppler-Geschwindigkeit sowie den Radarquerschnitt, der von der Oberfläche, dem Material und der Form der Objekte abhängt. Dies ermöglicht die Unterscheidung zwischen bewegten und statischen Objekten, unterstützt die Klassifizierung und trägt zu einem verbesserten Verständnis der Umgebung bei.

Das Ziel dieser Arbeit ist die Entwicklung neuer und wirkungsvoller Ansätze, um das Szenenverständnis von autonomen Fahrzeugen auf Basis von Radar-Punktwolken in realen Umgebungen zu verbessern. Dabei untersuchen wir verschiedene, aufeinander aufbauende Aufgabenstellungen. Wir beginnen mit der semantischen Segmentierung, um Informationen über die Objektklassen in Radar-Punktwolken zu extrahieren. Die Einbindung spezifischer Radarinformationen wie der Doppler-Geschwindigkeit und des Radarquerschnitts resultiert in einem optimierten Algorithmus, der auch bei spärlichen Punktwolken zuverlässig funktioniert. Neben den semantischen Informationen ist auch die Unterscheidung zwischen der statischen Umgebung und bewegten Objekten für eine sichere Navigation unerlässlich. Daher entwickeln wir einen neuartigen Ansatz zur Segmentierung bewegter Objekte, der davon profitiert, dass eine binäre Klassifizierung die Genauigkeit im Vergleich zur allgemeinen semantischen Segmentierung erhöht. Auf der Grundlage der zuverlässigen Segmentierung von bewegten Objekten untersuchen wir die Erkennung von Instanzen, um individuelle Objekte innerhalb einer Szene zu unterscheiden. Die sich daraus ergebende Aufgabe der Segmentierung bewegter Instanzen verfeinert das Verständnis und schließt das Wissen über die Anzahl der Verkehrsteilnehmer ein, um komplexe Aufgaben wie die Kollisionsvermeidung zu lösen. Wir beziehen zeitliche Informationen aus aufeinanderfolgenden Radar-Punktwolken ein, um ein detailliertes räumliches und zeitliches Verständnis der Umgebung zu erhalten. Darüber hinaus entwickeln wir einen neuen Ansatz, der die extrahierten Informationen nutzt, um klassenunabhängige Instanzzuweisungen durchzuführen.

Die Segmentierung bewegter Instanzen führt zu herausragenden Ergebnissen und bildet damit einen idealen Ausgangspunkt, um das Szenenverständnis weiter zu verbessern. Zunächst verwenden wir die Detektionen und extrahieren zusätzliche Erscheinungsmerkmale sowie geometrische Beziehungen, um Instanzen über die Zeit zu assoziieren und zu verfolgen. Unsere Instanzzuordnung funktioniert, auch bei der Verfolgung von weit entfernten Objekten, die aus einzelnen Punkten bestehen, zuverlässig. In einem weiteren Ansatz prädizieren wir die semantische Klasse der bewegten Instanzen, da diese Information für viele Funktionen unerlässlich ist. Wir entwickeln hierzu einen neuartigen Ansatz, der die semantischen Klassen der einzelnen Agenten vorhersagt und die Informationen zur Optimierung der Instanzzuweisung nutzt.

Abschließend lässt sich festhalten, dass unsere Ansätze zu wesentlichen Fortschritten des Szenenverständnisses in verschiedenen Umgebungen beitragen. Die neuartigen Methoden sind entscheidend, um Radar-Punktwolken zuverlässig zu verarbeiten und lassen sich auf reale Daten übertragen. Zudem ist der Benchmark zur Segmentierung bewegter Instanzen öffentlich zugänglich, um die weitere Forschung zu unterstützen. Alle in dieser Arbeit vorgestellten Ansätze wurden in begutachteten Konferenzbeiträgen und Zeitschriftenartikeln veröffentlicht und tragen zur Weiterentwicklung des radarbasierten Szenenverständnisses in realen Umgebungen bei.

Abstract

Autonomous vehicles have the potential to revolutionize transportation by reducing accidents caused by human errors, improving efficiency, and enhancing mobility for everyone. The dynamic real-world environments impose several challenges, including varying lighting conditions, adverse weather, and interactions with diverse road users. Handling versatile scenarios is critical to prevent accidents and, for example, protect vulnerable road users as good as possible. Therefore, the reliable perception of the surroundings under changing conditions is a fundamental task for safe navigation in dynamic real-world environments.

Common perception stacks of modern autonomous driving systems comprise different sensors such as cameras, LiDARs, and radar sensors to leverage the advantages and mitigate the limitations of the individual modalities. LiDARs and cameras are useful sensors and provide detailed information most of the time. However, both sensor modalities face limitations under adverse weather, including rain, fog, and snow. Therefore, radar sensors, which work under these conditions, are critical to enable safe mobility. Radar sensors provide sparse point clouds to locate and identify objects within the surroundings of the autonomous vehicle. Each point in the cloud also contains additional information, such as the Doppler velocity, which is the radial velocity of the object, and the radar cross section, which quantifies the electromagnetic energy that is scattered back to radar by an object and depends on the surface, the material, and the shape of the objects. Consequently, the radar point clouds include relevant information to differentiate between moving and static instances and classify the versatile objects within the environment. However, radar scans are substantially affected by noise due to multi-path propagation, interference, and sensor limitations. Therefore, dedicated algorithms capable of handling sparse and noisy radar point clouds are fundamental to extracting high-level information. This includes achieving semantic understanding of the environment, which is a critical component for ensuring safe and robust autonomous driving functions.

The main contributions of this thesis are novel and impactful approaches that process radar point clouds to improve scene understanding of autonomous vehicles in real-world environments. We focus on several tasks that contribute to the perception and understanding of the environment. We start with semantic segmentation to extract information about the corresponding classes of objects in radar point clouds. We optimize the algorithm by including specific radar information such as the Doppler velocity and radar cross section to work reliably for sparse point clouds and enhance performance by reducing information loss. Besides the semantic information, the differentiation between static environment and moving objects is essential to navigate safely. Therefore, we propose a novel approach to address moving object segmentation, which benefits from the fact that a binary classification simplifies the overall segmentation compared to general semantic segmentation. The task is well suited for radar data because of the provided Doppler velocity. To deal with the noise, we introduced dedicated algorithms that explicitly exploit the Doppler velocity within each module to differentiate between moving and static detections. Based on the reliable segmentation of moving objects, we investigate how many agents are present. The resulting task of moving instance segmentation refines the understanding and includes the knowledge of the number of traffic participants to perform complex tasks such as collision avoidance. We incorporate temporal information from consecutive radar scans to derive a detailed spatio-temporal understanding of the scene, which is essential for online and dynamic path planning. Moreover, we propose a novel approach that utilizes extracted information to perform class-agnostic instance assignments within sparse and noisy radar data to derive the information for individual instances.

Since moving instance segmentation leverages the advantages of radar sensors and leads to exceptional results, the predictions are ideal for enhancing scene understanding further. First, we extract the appearance and geometric features of the instances to track them over time, which is essential for path planning. Our instance association works reliably, including the tracking of distant objects that only comprise one point. We utilize the moving instance predictions and predict the semantics of the individual instances. The semantic class of the instance provides additional information, which is essential for operating safely. Hence, we propose a novel approach that predicts the semantic classes of the individual agents and utilizes the information to refine the instance assignment.

In sum, our approaches show superior performance on diverse benchmarks, including diverse environments, and provide optimized modules to enhance scene understanding. In addition, we made the moving instance segmentation benchmark publicly available to support further research. All our proposed approaches presented in this thesis were published in peer-reviewed conference papers and journal articles, contributing to the advancements of radar-based scene understanding in real-world environments.

Acknowledgements

When starting my Ph.D. thesis, I would have never imagined how exciting this journey would be. I am deeply grateful that during that time, I was always surrounded by exceptionally talented and outstanding advisors, colleagues, and friends, who all made this possible. First and foremost, I want to thank my supervisors and mentors, Cyrill Stachniss and Michael Heidingsfeld, who both made writing an external PhD thesis an entirely fulfilling experience that combined the best of both worlds.

I want to thank Cyrill Stachniss for the endless support and invaluable advice, which helped me to develop and grow in a way far beyond my expectations. Starting from our first video call, he always supported me in developing essential academic skills. Without them, a PhD thesis within three years would never have been possible. Beyond that, Cyrill has been a source of valuable advice relevant to all aspects of life. He always enabled me to strive for excellence, and he sincerely encouraged me to drive for higher targets, which we, in the end, achieved and for which I am deeply grateful.

I want to thank Michael Heidingsfeld for his tremendous commitment and unwavering support, which helped in a way far exceeding anything I envisioned. He created an environment that enabled me to grow and learn rapidly from the first day onward, and without him, many achievements would not have been possible. He always encouraged me with a positive mindset and guided me toward achieving my goals. It was truly inspiring to learn from his advice, which helped me overcome challenges and navigate both corporate and personal life. I could not have asked for better supervisors.

I would also like to thank Markus Enzweiler and Lasse Klingbeil for their valuable time dedicated to reviewing my thesis and for their thoughtful feedback.

During my PhD, I was always surrounded by numerous extraordinary friends and colleagues who not only supported me but made the journey far more fulfilling. Sharing the same experience to pursue a PhD in the same company and on the same team was genuinely meaningful. I want to thank Daniel Casado Herraez for his thoughtful feedback, sharing of innovative concepts, and for his consistent optimism, which is truly inspiring and brightens every situation. Side by side, we advanced in our personal lives, scientific pursuits, and broader endeavors, sharing

countless memories. Thank you so much, RadarBro. I am deeply grateful to Jens Behley for his endless support, his invaluable guidance, and for helping me accomplish things beyond what I ever believed achievable. His motivation to push the boundaries shaped my work from the beginning, and without him, many achievements would not have been possible. I am grateful for his dedication to the whole project and his profound, in-depth understanding, which always helped to shape ideas. Our meetings always exceeded the planned time frame but for good reasons. Our discussions have always been inspiring and extraordinarily valuable for all my work, and I am sure that they will influence all my future endeavors.

Being part of the lab and CARIAD brought me into contact with many supportive colleagues, making it both more rewarding and enjoyable. Visiting Bonn was always a pleasure and a source of motivation and great ideas. The CARIAD PhD colleague is a true foundation of inspiration, and I deeply value our valuable discussions and shared experiences. I am profoundly grateful to Sebastian Ammann, Bengisu Ayan, Tobias Bieler, Nina Burdorf, Le Chang, Xieyuanli (Rhiney) Chen, Xavier Timoneda Comas, Yue (Linn) Chong, Tiziano Guadagnino, Suchit Gupta, Mareike Grund, Jens Heine, Antoine Hunou, Christoph Hümmer, Florian Jaumann, Liren Jin, Umair Khawaja, Franz Kaschner, Sharang Kaul, Birgit Klein, Fabian Konstantinidis, Haofei Kuang, Moritz Kuhl, Thomas Läbe, Luca Lobefaro, Federico Magistri, Rodrigo Marcuzzi, Elias Marks, Lucas Nunes, Yue Pan, Gianmarco Roggiolani, Julius Rückin, Benedikt Rosarius, Vardeep Singh Sandhu, Julian Schwab, Leon Schwarzer, Matteo Sodano, Nam Thanh, Ignacio Vizzo, Jan Weyler, Philipp Walther, Christian Witte and Xingguang (Starry) Zhong. Having an exceptional working environment eased the way, and I am thankful for all of you.

You have been by my side long before the PhD adventure started, and your encouragement and belief have always been invaluable for the success of my journey. I want to express my heartfelt gratitude to my wonderful friends for their enduring trust and unwavering support. Each of you holds a special place in my heart, and the moments we share are not only full of joy but also a constant source of motivation. I feel incredibly fortunate to have you all by my side.

The foundation of all my achievements lies in the endless support of my family. My parents, Christel and Rudolf, whose consistent love and guidance have been truly invaluable to me. Your confidence in my strengths and unconditional support in my decisions are everything for me. Without you, this thesis and everything else would not have been possible. I am forever thankful for everything you have done. My brother, Christian, always believed in me, giving me strength in times of doubt. Your optimism and effortless handling of life have always been truly inspiring. I am deeply thankful for all our unforgettable moments of pure joy and happiness.

Finally, to my greatest supporter, Steffi, your belief in me throughout every challenge and your unwavering patience and encouragement throughout this journey have been my constant source of strength. My gratitude is beyond what words can ever convey, and I am endlessly thankful to have you as my companion. You inspire me to be the best version of myself, and I know I could never have achieved all of that without you.

Contents

Acronyms

***k*NN**	k nearest neighbor
FFT	fast Fourier transform
IoU	intersection over union
LiDAR	light detection and ranging
MLP	multi-layer perceptron
PQ	panoptic quality
radar	radio detection and ranging

Chapter 1

Introduction

The advent of autonomous robots and vehicles marks a fundamental transformation of the industry and changes our everyday lives. The applications of autonomous vehicles for transportation are versatile, from retail warehouse and delivery robots, which automate the distribution of goods, enhancing the efficiency of production and facilitating last-mile logistics, to autonomous vehicles, which operate independently in real-world environments, increasing efficiency, reducing accidents, and enabling mobility for everyone. The improvement of safety in real-world environments especially makes autonomous vehicles valuable to society.

Globally, traffic accidents claim 1.35 million lives annually, making them the leading cause of death among children and young adults [228]. Aiming to reduce these fatalities by minimizing critical errors caused by human failures positions autonomous vehicles as a promising technology to make mobility safer. To operate independently or with minimal human intervention, automated vehicles and robots rely on the precise perception of the environment and a safe driving strategy, including path planning and the accurate control and actuation of the vehicle to perform tasks reliably. The tasks are strongly interconnected, and thus, the precision and reliability of the perception algorithms are crucial to reducing failures in planning and execution. The perception system is complex and must work in diverse and dynamic environments, including rural roads, crowded streets, and highways. Furthermore, the systems must work reliably under changing lighting conditions, diverse traffic scenarios, and weather conditions. Perception systems of autonomous vehicles aim to ensure robustness against these external factors, permanently preserving high accuracy. To achieve this, the sensor setup of highly automated vehicles is versatile, incorporating LiDARs, cameras, and radars. The redundant sensor setup of autonomous cars with different modalities aims to reduce the risk of critical malfunctions by combining the advantages of the individual sensors. While camera and LiDAR processing have made tremendous

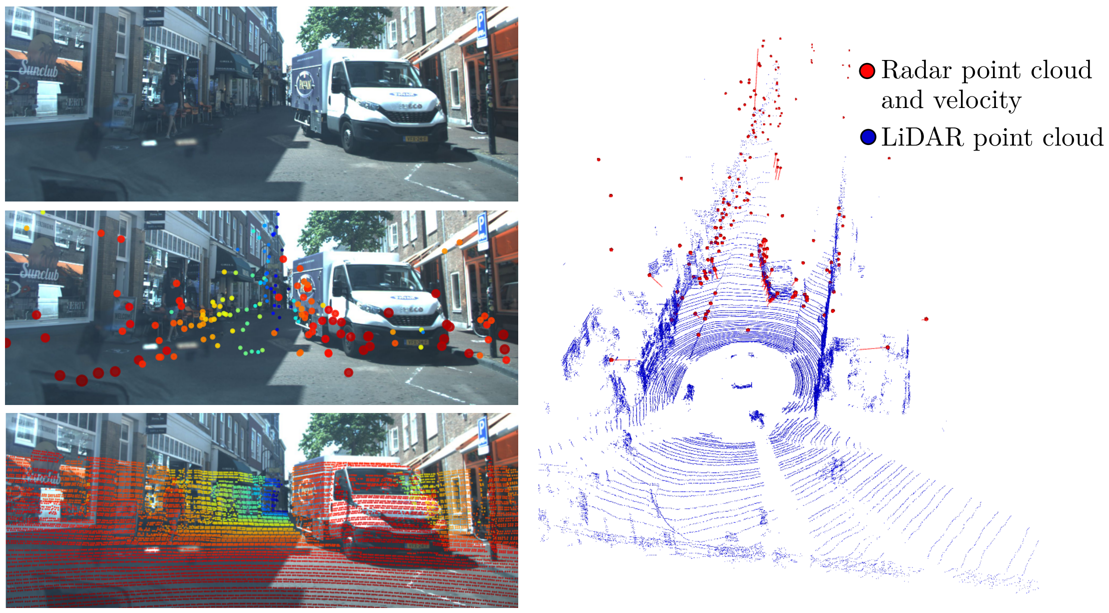


Figure 1.1: Overview of camera, LiDAR, and radar data for an urban street scene. The images on the left include the projected radar (middle) and LiDAR point clouds (bottom), which are colored according to the distance to the actual sensor. The 3D point cloud of the scene on the right combines LiDAR and radar data.

progress in recent years, strengthening the overall scene understanding, these sensors face inherent limitations in adverse weather. Radar sensors work reliably under adverse weather conditions, including rain, fog, and snow, making them indispensable by overcoming the limitations of cameras and LiDARs. Furthermore, changing lighting conditions do not affect radar sensors, making them robust for various applications. Therefore, processing radar data separately from LiDAR data is key to enhancing the overall scene understanding.

Compared to LiDAR point clouds, the data of automotive radar sensors are sparse point clouds, as illustrated in Figure 1.1. Furthermore, radar data is affected by noise due to multi-path propagation, sensor noise, signal processing artifacts, and interference. However, besides the position information of the individual points, the radar data includes specific information about the objects within the environment, such as the radar cross section and the Doppler velocity. The radar cross section value depends on the surface, the material, and the shape of the object and measures how detectable an object is by the radar. Hence, it helps to differentiate between various objects. The Doppler velocity, the radial velocity between the sensor and the detected object, is essential to identify moving agents. Knowing which parts of the environment are moving and which are static is crucial for many tasks, including safe path planning, localization, and mapping. The significant advantage of measuring the Doppler velocity is that the moving parts are directly identifiable within a single point cloud. In contrast, other sensors, such as cameras and LiDARs, usually require the processing of sequences of data to identify moving objects, resulting in increased latency.

The scene understanding of autonomous vehicles incorporates multiple tasks such as segmentation, instance assignment, and tracking of objects to operate safely in real-world environments. The primary goal is to extract all relevant information, such as the semantic class of the traffic participants, to leverage the full potential of consecutive downstream tasks. Whereas geometric information about objects was sufficient for the early implementations of adaptive cruise control [226], achieving higher levels of autonomy today requires more complex scene understanding to handle versatile driving scenarios.

The tremendous advancements in artificial intelligence and the evolution of deep learning techniques for processing sensor data have driven significant improvements in semantic scene understanding. While the primary focus has been on dense data provided by cameras and LiDARs, enabling successful applications, extending these techniques to radar presents unique challenges due to the sparsity of radar point clouds. To address this, the aggregation of multiple radar point clouds is often used to achieve competitive performance, resulting in densification that simplifies the task. However, for real-world applications, instantaneous processing remains essential due to the strict latency requirements of driving tasks. This is particularly critical for applications like collision avoidance, which require immediate feedback. Additionally, the processing of aggregated point clouds results in repetitive processing of the individual detections, significantly increasing memory and computational demands.

Addressing these challenges, this thesis focuses on extracting meaningful semantic information from measurements taken at a single time step, such as a radar point cloud of the environment. Furthermore, we leverage radar-specific properties such as direct motion information to identify moving agents within the vehicle's surroundings and overcome the limitations of camera and LiDAR data processing. This highlights the importance of the processing of radar data for safe and reliable functionalities.

Overall, we propose dedicated algorithms to enhance point-based radar processing and harness their potential to improve scene understanding. The goal is to develop radar-based algorithms that work safely and reliably under adverse weather and compensate for the limitations of other sensor modalities. Furthermore, radar sensors are already available in series production vehicles due to their low cost, making these advancements practical, widely accessible, and particularly relevant to society by reducing accidents. We split the remainder of the thesis into two parts. In Part I, we address the segmentation of radar point clouds focusing on dynamic objects. In Part II, we exploit the predictions and utilize them to solve subordinate tasks, enhancing the scene understanding.

1.1 Main Contributions

The main contributions of this thesis are novel approaches that utilize single-scan radar data to improve the perception of autonomous vehicles, resulting in improved scene understanding. The proposed methods incorporate radar-specific information within advanced algorithms to leverage the full potential of the sensor modality and improve overall performance. Before we elaborate on the individual approaches in detail, we introduce the basic techniques of radar signal processing and deep neural networks, which are key techniques to understand our contributions and the foundation of our approaches in Chapter 2 and Chapter 3, respectively. In Chapter 4, we introduce the related work to derive central aspects of existing methods in the field of point cloud processing for autonomous vehicles. We divide our approaches into two parts to explore the different aspects of the perception tasks. In Part I of our main contributions, we elaborate on the segmentation of sparse radar point clouds to improve the performance of radar-based perception tasks. Starting with Chapter 5, we focus on multi-class semantic information of moving objects. We present an optimized network to address the difficulties of sparse and noisy radar data. The improvements focus on extracting valuable information from individual scans, where a single point can represent whole instances, such as pedestrians or distant vehicles. Therefore, the learning process of the neural network for single-scan processing requires the extraction of meaningful features to enhance the performance. The proposed method generates more precise semantic segmentation results by leveraging the full potential of radar data within the optimized modules.

On top of exploiting the general semantic information from sparse radar point clouds, in Chapter 6, we address the problem of long-tailed data distribution and answer the question of which type of semantic information is more useful for different tasks. Specifically, we differentiate between moving and non-moving objects, such as driving cars and moving pedestrians, from static or non-moving objects, such as buildings, vegetation, and parked cars. Compared to other sensor modalities, radar sensors provide unique advantages, such as direct Doppler velocity measurements, which allow us to overcome the limitations of aggregated input data processing, which induce latency. By leveraging these strengths, we propose a single-scan approach for more efficient perception. We incorporate dedicated modules to explicitly utilize the Doppler velocity information within the neural network to enhance moving object segmentation.

Since radar point clouds are prone to noise from multi-path propagation, sensor noise, and interference, exploring the Doppler velocity information alone is insufficient for advanced moving object segmentation. Temporal information is helpful to identify noisy detections as these points fluctuate over time. Therefore, temporal information is essential to improve performance, but the aggregation of scans over time is not desired due to latency issues. In Chapter 7, we propose an efficient module to enrich the current scan with temporal information to improve the moving object segmentation and reduce latency. Furthermore, we introduce advanced instance segmentation modules to incorporate the instance information within our network and identify the individual moving agents within the surroundings of the autonomous vehicle. The difficulty of moving instance segmentation is that all objects belong to the same semantic class. Hence, the differentiation of the instances has to work reliably without relying on class-dependent information. Our optimized approach addresses this challenge and leads to superior segmentation performance.

Based on predicting the semantic classes from the sparse and noisy input point clouds, Part II focuses on exploring the moving instance predictions to solve additional tasks necessary for the overall scene understanding. In Chapter 8, we concentrate on tracking moving instances over time, a central capability of the perception algorithms to enable safe and reliable path planning. The difficulties of tracking moving agents primarily result from the sparse input data and occlusion artifacts, making the correct association of instances over time challenging. We propose combining the geometric and appearance features of individual agents to derive precise representations and improve temporal associations. Our optimized neural network learns temporal displacements of the instances and generates accurate association scores to ensure reliable tracking.

Despite the exceptional results for moving object and moving instance segmentation for sparse and noisy radar data, the actual semantic class is essential for scene understanding, particularly for interacting with other traffic participants. We introduce our method in Chapter 9 to predict the semantic classes of the moving instance predictions. We propose a dedicated algorithm to extract meaningful features from the filtered point clouds and refine the segmentation results leveraging the advantage of the semantic information. Our two-step approach combines the benefits of moving instance segmentation and the subsequent processing by an improved neural network to enhance the overall performance.

In Chapter 10, we conclude and provide an outlook for future work to further improve scene understanding for autonomous vehicles. Overall, this thesis presents novel approaches to processing single-scan, sparse, and noisy radar data, tackling multiple challenges to enhance scene understanding. We study the properties of radar data in detail, propose dedicated modules, and solve various tasks to improve performance, making an essential step toward reliable and safe algorithms for autonomous vehicles.

1.2 Publications

Parts of this thesis were published in the following peer-reviewed journal and conference articles, for which I was the main contributor:

- M. Zeller, J. Behley, M. Heidingsfeld, and C. Stachniss. Gaussian Radar Transformer for Semantic Segmentation in Noisy Radar Data. *IEEE Robotics and Automation Letters (RA-L)*, 8(1):344–351, 2023. DOI: 10.1109/LRA.2022.3226030
- M. Zeller, V.S. Sandhu, B. Mersch, J. Behley, M. Heidingsfeld, and C. Stachniss. Radar Velocity Transformer: Single-scan Moving Object Segmentation in Noisy Radar Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023. DOI: 10.1109/ICRA48891.2023.10161152
- M. Zeller, V.S. Sandhu, B. Mersch, J. Behley, M. Heidingsfeld, and C. Stachniss. Radar Instance Transformer: Reliable Moving Instance Segmentation in Sparse Radar Point Clouds. *IEEE Trans. on Robotics (TRO)*, 40:2357–2372, 2024. DOI: 10.1109/TRO.2023.3338972
- M. Zeller, D. Casado Herraez, J. Behley, M. Heidingsfeld, and C. Stachniss. Radar Tracker: Moving Instance Tracking in Sparse and Noisy Radar Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024. DOI: 10.1109/ICRA57147.2024.10610198
- M. Zeller, D. Casado Herraez, B. Ayan, J. Behley, M. Heidingsfeld, and C. Stachniss. SemRaFiner: Panoptic Segmentation in Sparse and Noisy Radar Point Clouds. *IEEE Robotics and Automation Letters (RA-L)*, 2024. DOI: 10.1109/LRA.2024.3502058

This thesis contributes to different collaborations, which address additional aspects of radar processing and resulted in the following peer-reviewed conference and journal articles:

- D. Casado Herraez, M. Zeller, L. Chang, I. Vizzo, M. Heidingsfeld, and C. Stachniss. Radar-Only Odometry and Mapping for Autonomous Vehicles. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024. DOI: 10.1109/ICRA57147.2024.10610311
- D. Casado Herraez, L. Chang, M. Zeller, L. Wiesmann, J. Behley, M. Heidingsfeld, and C. Stachniss. SPR: Single-Scan Radar Place Recognition. *IEEE Robotics and Automation Letters (RA-L)*, 9(10):9079–9086, 2024. DOI: 10.1109/LRA.2024.3426369

Chapter 2

Fundamentals of Automotive Radar

The sensor suites of autonomous vehicles are versatile to increase redundancy and work under all driving conditions. Radar data signal processing is fundamental for safe and reliable autonomous mobility due to the robustness under adverse weather, as explained in Chapter 1. Consequently, we aim to leverage radar data to enhance the overall scene understanding of autonomous vehicles. We derive an overview of the radar sensor data in the following chapter to understand the properties and advantages of the sensor modality. For completeness, we examine the radar signal processing to obtain the fundamentals that influence the final radar point cloud. We focus on the basic techniques related to the automotive industry. In Section 2.1, we introduce the automotive radar sensor in detail, including the basic specifications. In Section 2.2, we explain the radar equation, which combines all parameters and factors that need to be considered to detect an object within the surroundings of a self-driving vehicle. Section 2.3 focuses on radar signal processing to derive the range, Doppler velocity, and angle measurements. Finally, we elaborate on noise in radar data in Section 2.3.5 and the point cloud extraction in Section 2.3.6.

2.1 Automotive Radar Sensors

Radar is an acronym for radio detection and ranging, which means radars emit electromagnetic waves to detect and locate objects. Compared to other sensor modalities, the significant advantages are robustness under adverse weather, penetration of material, long-range sensing capabilities, and radar-specific sensor information. Consequently, radar sensors have a long-lasting history in the automotive industry. The first ideas for radar-based collision avoidance systems emerged in the 1960s [139, 149] resulting in increasing interest and research and development [64, 137, 149]. The first operational sensors were deployed in the 1990s to reduce severe accidents [52]. Today, radar sensors are an integral part of advanced driver assistance systems, including versatile applications such as adaptive cruise control, lane changing assistant, collision mitigation, and rear cross-traffic alert, increasing safety in real-world traffic scenarios. Due to the low cost compared to LiDARs, multiple radars are already installed in today's vehicles to enable a 360° coverage of the environment. The sensor setup includes short-range radars on the side of the vehicle sensing up to 30 m, mid-range radars at the corners of the vehicle sensing up to 150 m, and front-facing long-range radars sensing up to 400 m.

Over the past decades, radar sensor design and signal processing have developed. The commonly used frequency band within the automotive industry, for example, increased from 24.05-24.25 GHz to 76-81 GHz, resulting in a smaller wavelength, which typically reduces the required antenna size. As a result, the compact sensor design enables the integration of multiple sensors. The resulting wavelength of the emitted electromagnetic wave is around 3.8 mm, which influences, among others, atmospheric attenuation. In contrast to cameras and LiDARs, where the wavelength is in the nanometer range, radar signals are less absorbed or scattered by small particles, allowing them to propagate through rain, fog, and snow. Besides the robustness under adverse weather, the specification and design of the radar sensor determine the performance of the radar sensors, which we elaborate on in this chapter. In the following section, we introduce the radar equation to explain the fundamental properties of radar signal transmission.

2.2 Radar Equation

The radar equation [11] is the fundamental formula that describes the relationship of the received power dependent on the characteristics of the transmitted signal, the environment, and the detected object. The overall process starts with the transmission of a signal, where the radar sensor includes at least one transmitting antenna that emits electromagnetic radiation. The objects in the field of view of the radar, the so-called targets, reflect the radar waves in multiple directions. The energy that is reflected towards the radar sensor is collected by the receiving antenna and processed in signal processing to extract relevant information.

The process starts with the emission of the power P_t by an isotropic radar antenna that emits uniformly in all directions and distributes energy over the surface of $4\pi R^2$ of an imaginary sphere where the radius corresponds to the range R to the sensor. Since radar antennas of automotive radar sensors are directive, the power increases in a particular direction. Therefore, the gain G_t measures the increased power radiated towards the desired direction compared to an isotropic antenna. The radar beam represents the sector with the highest power concentration, where a narrower, more sharply defined beam has a higher gain. The size of the beam depends on several variables, including the wavelength and the aperture diameter. Since the wavelength of automotive radars is fixed to a frequency band, the aperture diameter is essential, where a larger diameter results in a smaller beam and, thus, higher gain. However, the size of the aperture is limited due to design constraints within the vehicle, resulting in a trade-off. Furthermore, we need to consider that a larger radar beam results in a wider field of view, which is important to cover larger areas of the surroundings but reduces the antenna gain. The resulting power density at a target from a directive automotive antenna is $P_t\, G_t / 4\pi R^2$.

The target intercepts only a portion of the incident power and reflects it. The power density of the echo signal depends on numerous factors such as size, geometry, surface, material, and the direction of incidence of the radar beam summarized in the radar cross section σ of the target. The radar cross section quantifies the detectability of the respective target. Consequently, large objects, such as trucks, typically have larger radar cross section values compared to smaller objects, such as pedestrians. We assume in the simple case that the reflected signal also emits uniformly in all directions, resulting in the distribution over the surface of a sphere $4\pi R^2$. The received reflected power P_r depends on the effective area of the receiving antenna, denoted as A_e. Since the signal is reflected, refracted, or scattered in various directions, the receiving antenna captures only a fraction of the transmitted energy. The resulting basic radar equation summarizes the signal transmission for the received power P_r as follows:

$$P_r = P_t \left(G_t A_e\right) \sigma \left(\frac{1}{4\pi R^2}\right)^2, \tag{2.1}$$

where the first term, P_t, represents the transmitting power of the radar. The second term $(G_t A_e)$ covers the characteristics of the radar sensor. The third term, the radar cross section σ, describes the object, and the fourth term represents the outward and return path of the signal. This basic radar equation emphasizes the important aspects of radar sensors relevant to this thesis. For practical applications, different aspects, including sensor noise, attenuation caused by atmospheric effects, multi-path propagation, and the penetration of the material, including the dome of the sensor, need to be considered to derive a comprehensive sensor model.

The major takeaway from the radar equation is that the received power decreases to the power of four by the range R. Consequently, distant targets are challenging to detect. Furthermore, the radar cross section, which describes the detectability of the object, is independent of the distance. Therefore, the radar cross section can help to classify targets because it depends on the properties of

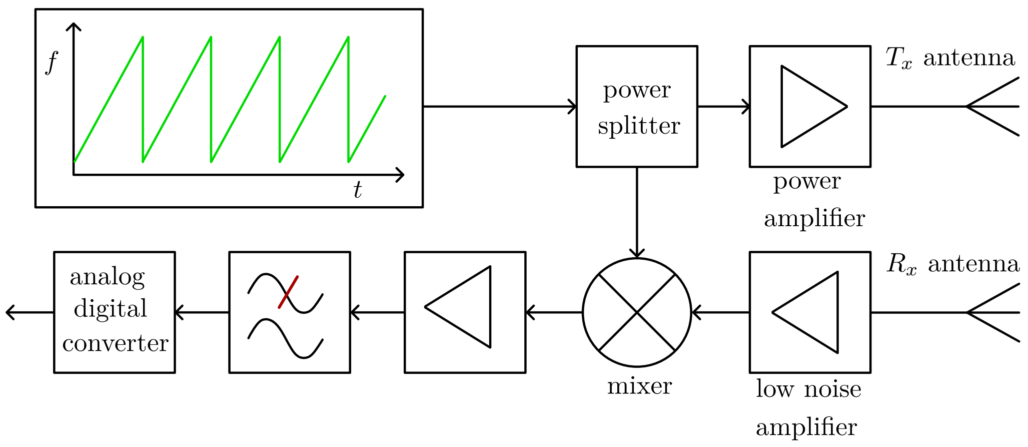


Figure 2.1: Overall working principle of a radar sensor. The transmitted signal is generated by frequency modulation and passed through a power splitter and power amplifier before being emitted by the transmitting antenna. The received signal is processed by a low noise amplifier and mixed with the transmitted signal. The resulting output is first amplified and filtered before being converted by an analog-to-digital converter for consecutive signal processing.

the target, such as the material, the shape, and the size. As a result, the approximation of the radar cross section of the target dependent on the transmitted and received power plays a central role in radar signal processing [9]. Furthermore, we can conclude that distant objects with small radar cross sections are particularly difficult to detect. Moreover, the beam width increases with the distance to the sensor, resulting in larger spatial coverage and reduced angular resolution, making the differentiation of closely spaced objects harder. Consequently, the detection and separation of distant objects is more difficult. To extract additional information about the target, we process the received signal, which we expand on in the following sections.

2.3 Radar Signal Processing

Deriving critical information about objects in the surroundings of autonomous vehicles from radar signals involves complex signal processing. To understand the central principles, we first introduce the range and Doppler estimation for a single target in Section 2.3.1 to derive the basic formulas before presenting signal processing for multiple targets in Section 2.3.2. In Section 2.3.3, we present the angle estimation. Additionally, we elaborate on the specific properties of the radar data in Section 2.3.4 and Section 2.3.5.

2.3.1 Range and Doppler Estimation: Single Target

Signal processing is necessary for locating the radar target and extracting additional properties of the target, such as the Doppler velocity [225]. The processing

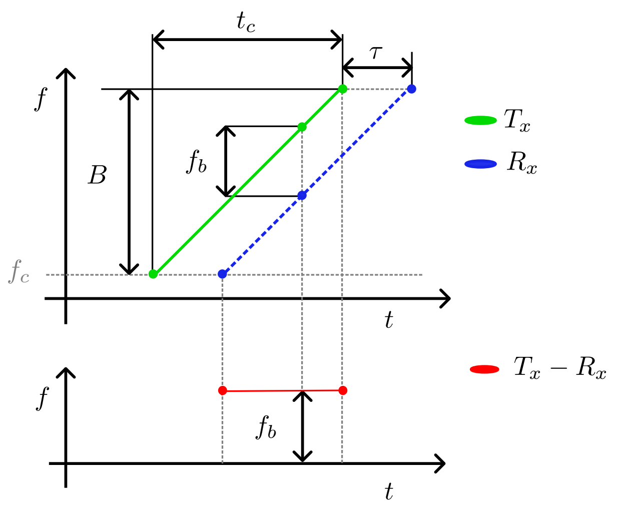


Figure 2.2: Transmitted and received signal frequency for the simple case of the detection of a single static object (top) and the frequency difference between these signals (bottom) for one chirp.

depends on the emission of the radar signal itself. On a basic level, we differentiate between pulsed and continuous wave radars. A pulsed radar transmits a short pulse and pauses the emission to listen for an echo signal, whereas a continuous wave radar continuously emits a radar signal. Current automotive radars fall into the second category and use frequency modulations resulting in so-called frequency modulated continuous wave radars [224].

Automotive radar sensors nowadays include multiple transmitting and receiving antennas. To understand the concepts, we first introduce the basic building blocks of the radar sensor for one transmitting and receiving antenna. The frequency modulation often increases the frequency linearly, resulting in so-called chirps, as depicted in the overall system diagram in Figure 2.1. To generate the modulated signal, radar sensors utilize chirp generators and voltage-controlled oscillators. The signal passes an amplifier before being radiated through the transmitting antenna. The receiver antenna listens to the echo signal. After the low-noise amplifier, the signal is mixed with the current signal in the mixer. The mixed signal is amplified and passed through a low-pass filter. The analog-to-digital converter transforms the signal. We process the output signal to obtain information on the radar targets.

We first assume the simple case of detecting a single static object without a relative radial velocity. Therefore, we consider a single chirp of a sawtooth signal to derive a detailed explanation of the dependencies of the measurements, as visualized in Figure 2.2. We obtain the functions for the transmitted and received signals, which form the basis for the signal processing procedure. The frequency modulation of the transmitted signal starts with the carrier frequency f_c, which is the lowest frequency in the signal. The linear modulation, which is the most common in automotive radar sensors, is defined by the frequency bandwidth B and the duration of a chirp t_c. The frequency will increase over time t by the bandwidth and reach the maximum after the duration of the chirp, resulting in the frequency function $f(t)$ as follows:

$$f(t) = f_c + \frac{B}{t_c} t, \tag{2.2}$$

where the maximum frequency is defined by the sum of the carrier frequency and the bandwidth. The instantaneous phase $\Phi(t)$ [5] of the transmitted signal, which is the phase of the signal at any given moment in time, depends on the initial phase ϕ_{T_x}, and the frequency resulting in:

$$\Phi(t) = 2\pi \int_0^t f(\hat{t})\, d\hat{t} + \phi_{T_x} = 2\pi \left(f_c t + \frac{B}{2t_c} t^2 \right) + \phi_{T_x}. \tag{2.3}$$

Combining the information, the transmitted signal with an amplitude $\hat{u}_{T_x}$ can be expressed as:

$$u_{T_x}(t) = \hat{u}_{T_x} \cos(\Phi(t)), \tag{2.4}$$

where we can substitute the instantaneous phase according to Equation (2.3). The transmitted and received signals have the same frequency as displayed in Figure 2.2. The static case does not include a relative movement and, hence, no change in frequency due to the Doppler effect. Due to the time of flight to the object and back to the sensor, the signal is received after a time difference τ. In terms of the distance R, τ can be calculated as:

$$\tau = \frac{2\,R}{c_0}, \tag{2.5}$$

where $c_0 = 3 \cdot 10^8\,\mathrm{m/s}$ is the speed of light, and due to the fact that the signal needs to travel to the object and back, we add the factor of two. It is worth mentioning that radar applications traversing the atmosphere need to consider atmospheric effects that influence the speed of light, which is not the case for automotive radars. We can express the frequency of the received signal f_{R_x} based on the time difference τ and the frequency of the transmitted signal, as follows:

$$f_{R_x}(t) = f_{T_x}(t - \tau). \tag{2.6}$$

Since the time delay results in a frequency difference between the transmitted and received signal, we can substitute τ by the so-called beat frequency $f_b(t)$:

$$f_{R_x}(t) = f_{T_x}(t) - f_b(t). \tag{2.7}$$

The constant beat frequency during the chirp is known as f_b, visualized in Figure 2.2. These beat frequencies f_b and the time delay τ are related to the bandwidth and the duration of a chirp as:

$$\frac{f_b}{\tau} = \frac{B}{t_c}, \tag{2.8}$$

which can be derived from the geometrical relations of the triangle. We substitute τ, following Equation (2.5) and solve the equation for R resulting in:

$$R = \frac{f_b\, c_0\, t_c}{2\, B}. \tag{2.9}$$

Therefore, we can calculate the distance of the radar detection based on the beat frequency and the parameters of the radar sensor. To determine the beat frequency, we mix the transmitted and the received signals. The signal obtained at the receiver antenna is identical to the emitted signal except for the time delay and the amplitude. According to Equation (2.4), the instantaneous phase Φ is affected by the time delay, resulting in the function for the received signal:

$$u_{R_x}(t) = \hat{u}_{R_x}\ \cos\left[2\pi\left(f_{T_x}(t) - f_b(t)\right)t + \phi_{R_x}\right]. \tag{2.10}$$

The resulting mixed signal $u_m(t)$ of the transmitted and received signal is:

$$\begin{aligned} u_m(t) &= u_{T_x}(t) u_{R_x}(t) \\ &= \hat{u}_{T_x}\cos\left[2\pi f_{T_x}(t)t + \phi_{T_x}\right]\ \hat{u}_{R_x}\cos\left[2\pi f_{T_x}(t)t - 2\pi f_b(t)t + \phi_{R_x}\right]. \end{aligned} \tag{2.11}$$

The mixing results in the combination of addition and subtraction of the signals because of the cosine property of $\cos(\alpha)\cos(\beta) = \frac{1}{2}[\cos(\alpha-\beta) + \cos(\alpha+\beta)]$. Consequently, the resulting equation for the mixed signal is:

$$\begin{aligned} u_m(t) = \frac{1}{2}\,\hat{u}_{T_x}\,\hat{u}_{R_x}\ &\left(\cos\left[\phi_{T_x} + 2\pi\, f_b(t)\, t - \phi_{R_x}\right] + \right. \\ &\left.\cos\left[4\pi\, f_{T_x}(t)\, t - 2\pi\, f_b(t)\, t + \phi_{T_x} + \phi_{R_x}\right]\right). \end{aligned} \tag{2.12}$$

The mixed signal includes a low-frequency and a high-frequency component. The low frequency, which equals the difference of the frequencies known as the beat frequency, explained in Equation (2.7), is much smaller compared to the high-frequency component. Therefore, only the low-frequency component will remain after processing the signal by a low-pass filter. The resulting mixed signal after the filtering can be expressed as:

$$\begin{aligned} u_{m,lf}(t) &= \hat{u}_{T_x}\,\hat{u}_{R_x}\,\frac{1}{2}\ \cos\left[2\pi\, f_b(t)\, t + \phi_{T_x} - \phi_{R_x}\right] \\ &= \hat{u}_M\ \cos\left[2\pi\, f_b(t)\, t + \phi_M\right]. \end{aligned} \tag{2.13}$$

The filtered signal only includes the beat frequency, which is required to calculate the distance of the target, according to Equation (2.9). To determine the

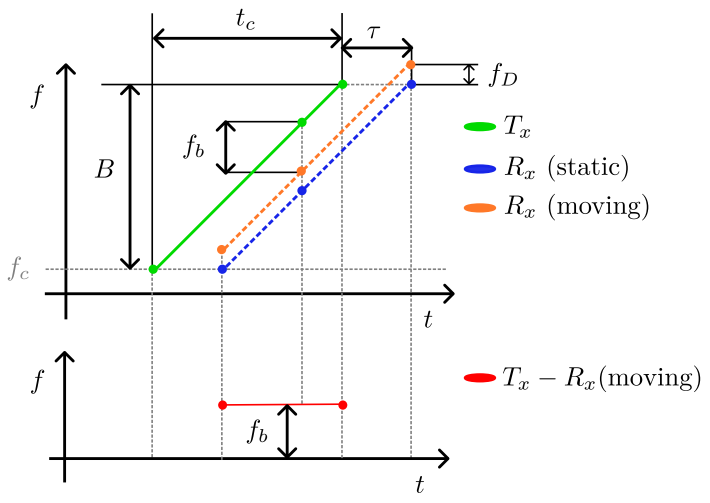


Figure 2.3: The transmitted and received signal frequency for the simple case of the detection of a static and moving single object with a radial velocity (top) and the frequency difference between the signals for the moving object (bottom) for a single chirp.

frequency based on the mixed and filtered signal, we can transform the function from the time to the frequency domain by a Fourier transform. The resulting frequency spectrum includes information about the frequency components of the signal and the respective amplitudes. Since the radar sensor signal processing results in a signal that mainly depends on the beat frequency, we can identify the frequency as the peak in the frequency spectrum after the Fourier transform and calculate the distance of the target.

Before further elaborating on the signal processing, we assume that the detected object is moving with a relative radial velocity with respect to the radar. The Doppler shift of the frequency directly influences the beat frequency, as depicted in Figure 2.3. The relative radial velocity causes a change in frequency due to the Doppler effect. The so-called Doppler frequency f_D combines the frequency change to the target and back to the sensor, resulting in:

$$f_D = 2\,\frac{v_r}{c_0}\,f_c, \tag{2.14}$$

where we assume $v_r \ll c_0$. The resulting beat frequency f_b, including a Doppler frequency, is:

$$f_b = \frac{B}{t_c}\,\frac{2R}{c_0} - f_D. \tag{2.15}$$

Replacing f_D with Equation (2.14) results in:

$$f_b = \frac{B}{t_c}\,\frac{2R}{c_0} - 2\,\frac{v_r}{c_0}\,f_c, \tag{2.16}$$

which illustrates that the beat frequency for a moving target depends on the Doppler velocity and the distance to the sensor. The Doppler velocity and ra-

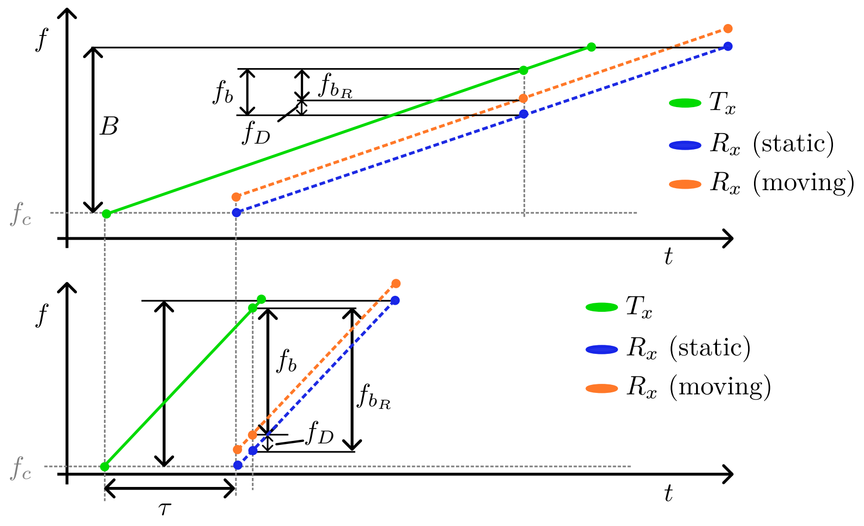


Figure 2.4: Transmitted and received signal frequency for the frequency modulated continuous wave radar (top) and the fast chirp adaptation (bottom), including a relative moving object and a static object. The beat frequency f_b comprises the frequency difference based on the distance f_{b_R} indicated by the time delay τ and the Doppler frequency f_D.

dial distance are coupled, resulting in the so-called range-Doppler coupling. The coupling prevents determining both range and Doppler velocity using a single chirp. When multiple targets are present, each target contributes a unique beat frequency corresponding to the specific range and Doppler velocity of the object. The superposition of the beat frequencies makes it challenging to decouple the range and Doppler velocity for each target, which complicates the resolution of range-Doppler coupling. However, the detection of multiple targets in the surroundings of a self-driving vehicle is essential for scene understanding. A common solution to resolve the coupling for multiple targets is the fast chirp sequence method, which we introduce in the following.

2.3.2 Range and Doppler Estimation: Multiple Targets

The detection of multiple targets is fundamental for radar signal processing in automotive applications. Therefore, we need to solve the range-Doppler coupling for multiple targets as mentioned in Section 2.3.1. One solution is the fast chirp sequence approach, which reduces the time of the individual chirps to get a near-stationary observation of the environment during the duration of the individual chirps. Hence, the influence of the Doppler effect is mitigated within one chirp since Doppler frequency f_D is much smaller than the part of the beat frequency which results from the distance of the object $f_{b,R}$ as depicted in Figure 2.4. To recover the important information on the Doppler velocity of the object, we need to utilize a sequence of chirps. The Doppler velocity causes a change in the phase

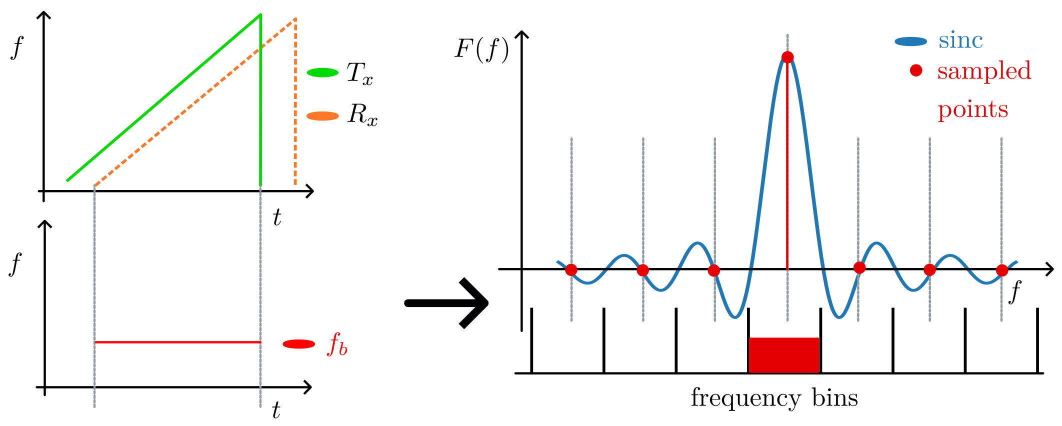


Figure 2.5: Illustration of the calculation of the distance based on a transmitted T_x and received R_x signal. The beat frequency f_b corresponds to a specific range of the target. To derive the information, we process the beat frequency by the range-FFT, resulting in the sinc function. The discrete values of the FFT relate to evenly spaced frequency bins that correspond to specific distances.

of the received signal over the sequence of chirps, enabling the effective resolution of the range-Doppler coupling.

We start the signal processing to determine the distance of the target based on the beat frequency. The first processing step focuses on the so-called fast time, the duration of an individual chirp. To obtain the distance R, we follow the same approach as in Section 2.3.1, including the mixing, the low-pass filtering, and the Fourier transform to determine the beat frequency. We utilize the fast Fourier transform (FFT) to efficiently compute the frequency of the discrete signal obtained after the analog-to-digital conversion. To illustrate the processing, we assume the simple case of one target. The mixed signal, described by the beat frequency, is a rectangular function. The Fourier transform of this signal $F(f)$ leads to the sinc function described as:

$$\operatorname{sinc}(x) = \frac{\sin(\pi\, x)}{\pi\, x}, \tag{2.17}$$

where $x = f_b\, t_c$ resulting in $F(f) = A \operatorname{sinc}(f_b\, t_c)$. The amplitude A depends on the received power. The signal includes a main lobe and several side lobes [227]. The main lobe is the frequency we are interested in since it relates to the corresponding beat frequency, which depends on the respective distances. To derive the frequency by the FFT, we sample s points during the duration of the chirp. The results of the FFT are s evenly spaced frequency bins. For the explained example, the resulting outputs of the FFT are the values of the sinc function at the center of the frequency bins, as visualized in Figure 2.5. Since the range depends on the beat frequency, the bins can be directly transferred into range bins, also known as range cells, according to Equation (2.9). As a result, we obtain the range measurement for the target as range cells. It is important to note that the

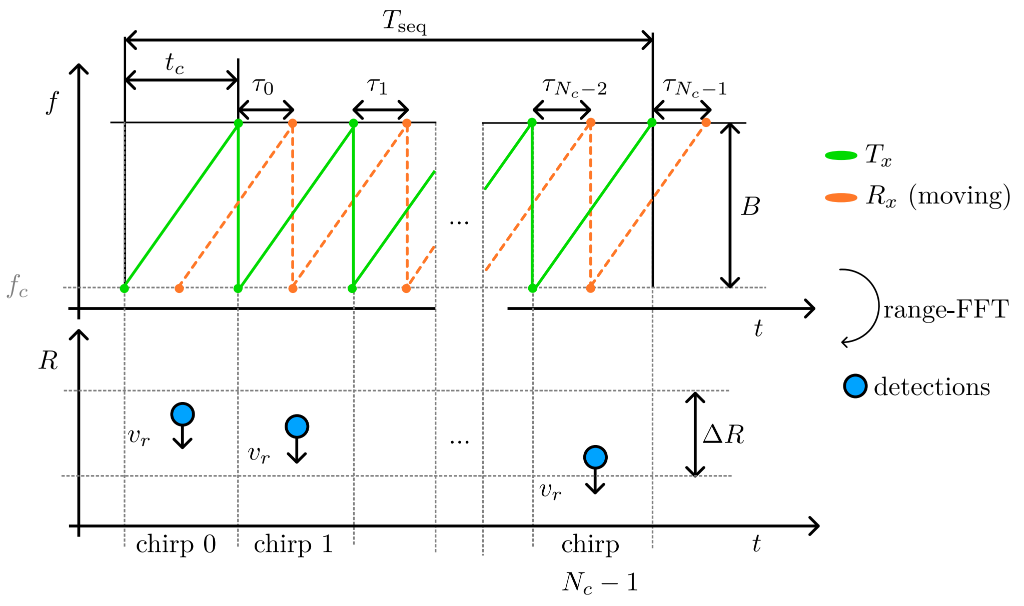


Figure 2.6: Radar signal processing in the fast time results in the range information for the individual radar targets represented in discrete range-cells. The real distance changes over time due to the Doppler velocity v_r.

case depicted in Figure 2.5 assumes an ideal scenario where no frequency component is spread into the neighboring bins, and only a single target is considered.

We compute the range-FFT for each chirp in the sequence, as depicted in Figure 2.6. The target appears in the same range cell over the sequence of chirps. However, the distance of the target changes due to the relative velocity. Therefore, we can extract the information about the Doppler velocity from a sequence of N_c chirps. The result of the range-FFT over the sequence of chirps for a target with relative velocity results in a phase shift between the consecutive chirps, which accumulates linearly over the sequence of chirps. This linear phase progression corresponds to a single frequency, the Doppler frequency. Consequently, the result of the range-FFT over the sequence of chirps varies due to Doppler frequency. Based on the Doppler frequency, we can determine the resulting radial velocity following Equation (2.14). Consequently, we apply a second FFT, the Doppler-FFT, which focuses on slow time, the duration of the sequence of chirps $T_{\text{seq}} = N_c t_c$, to determine the Doppler velocity of the object. We perform the Doppler-FFT over N_c chirps, resulting in N_c frequency bins. The resulting range-Doppler matrix has s rows for the range cells and N_c columns for the Doppler velocity, the speed cells. The consecutive processing resolves the range-Doppler coupling, and the fast chirp sequence approach enables the detection of multiple targets, resulting in a range-Doppler matrix, as depicted in Figure 2.7.

While range and Doppler information provide important information for scene understanding, they are insufficient for fully localizing targets within the environ-

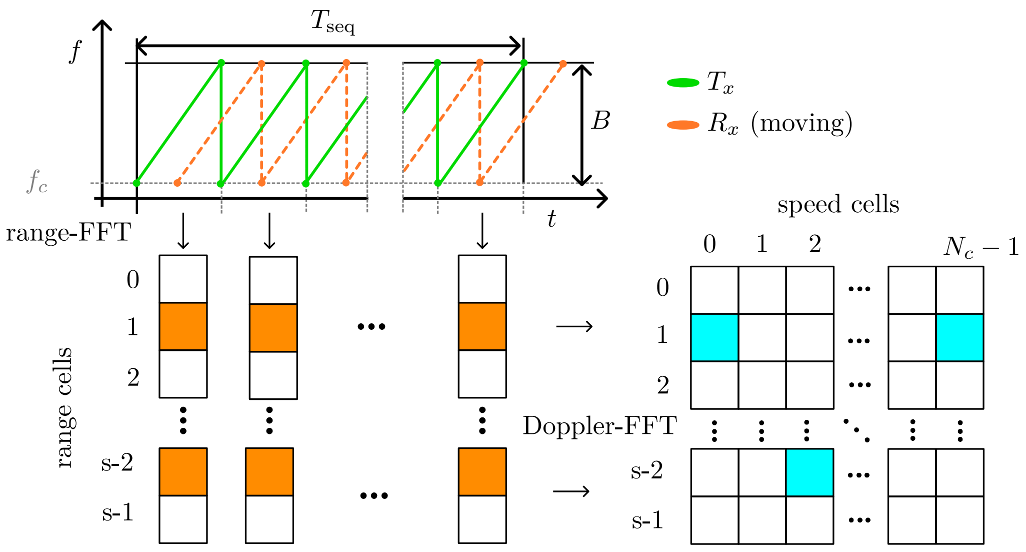


Figure 2.7: Signal processing of the sequence of chirps includes the fast time over the individual chirps to obtain the range and the slow time over the whole sequence of chirps to determine the Doppler velocity of the individual targets.

ment. Doppler velocity is particularly valuable for identifying moving objects, but without angular information, the exact position of a target cannot be resolved. Angle estimation is necessary to locate targets precisely within the scene. In the following section, we introduce the methods for estimating angular information, extending the sensor data for comprehensive scene understanding.

2.3.3 Angle Estimation

The signal processing to determine the Doppler and range requires a single transmitting and receiving antenna. To receive precise angular measurements, the common approaches in the automotive domain [11] rely on multiple receiving antennas, as depicted in Figure 2.8. We assume that the radial distance R to the object is much larger than the distance of the individual antennas d. Therefore, the transmitting and receiving signals can be modeled as planar waves. The closest receiving antenna to the object first receives the echo signal. This means that, for the second receiving antenna, the received signal needs to travel an additional distance Δs, which depends on the incident angle α of the signal and the distance d between the two antennas, as follows:

$$\Delta s = d \sin(\alpha). \tag{2.18}$$

The distance Δs increases for the consecutive receiving antennas. As a result, the received echo includes an additional phase difference ϕ^a, which can be derived

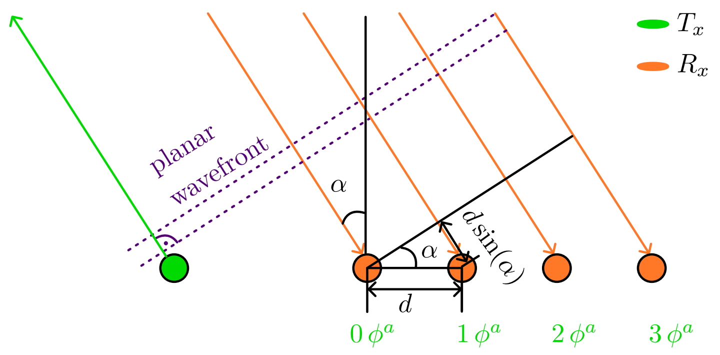


Figure 2.8: Radar sensors include multiple receiving antennas to derive the angle based on the additional distance $\Delta s = d\sin(\alpha)$ of the emitted signal to the antennas. The planar wavefront assumption is valid if the distance of the target is much larger than the distance of the antennas d. The angle α is the angle of incident.

as follows:

$$\phi^a = \frac{2\pi}{\lambda} d\sin(\alpha), \tag{2.19}$$

where λ is the wavelength of the signal. Therefore, we can measure the phase difference ϕ^a and calculate the angle of the received signal, resulting in:

$$\alpha = \sin^{-1}\left(\frac{\phi^a\lambda}{2\pi d}\right). \tag{2.20}$$

The measured phase difference ϕ^a and the angle α have a non-linear relationship due to $\sin^{-1}$, which results in a decreasing sensitivity if α increases. Hence, the angle estimation is more error-prone if the angle to the line of sight increases. Furthermore, the phase difference is linear proportional to the distance d of the antennas, resulting in $[0, \phi^a, 2\,\phi^a, \ldots, (N_{R_x}-1)\,\phi^a]$ for the respective N_{R_x} receiving antennas. Consequently, we can extract the spatial phase difference over N_{R_x} receiving antennas. We perform a third FFT to capture the spatial frequency corresponding to the phase shift over the range-Doppler bins of multiple antennas to obtain the direction of arrival, which is the angle of the target. The angular resolution is directly proportional to the number of sampling points, corresponding to the number of receiving antennas, which we elaborate on in detail in Section 2.3.4. To increase the resolution, the radar sensors require more receiving antennas, which usually results in a bigger sensor. However, larger sensors face packaging problems because of the limited space within the vehicle, reducing the resolution.

The standard approach is the multiple-input multiple-output setup to increase the angular resolution. Instead of transmitting the signal by one antenna, the setup uses multiple transmitting and receiving antennas as depicted in Figure 2.9. The distance of the transmitting antennas, in this example, is four times the distance of the receiving antennas. The signal of the second transmitting antenna

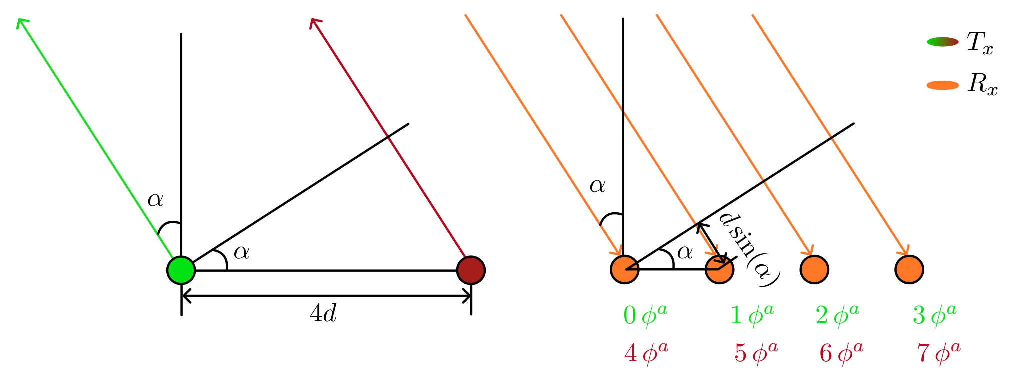


Figure 2.9: Automotive radar sensors combine multiple receiving and transmitting antennas in a multiple-input multiple-output setup to increase angular resolution.

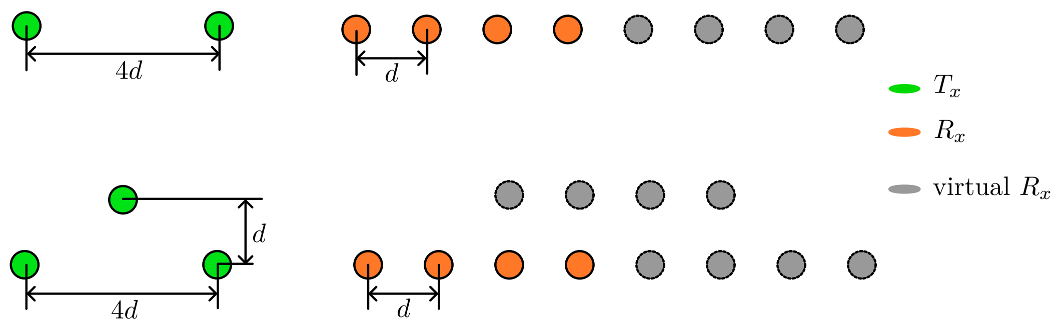


Figure 2.10: The multiple-input multiple-output setup is extended to a 2D array to measure elevation and azimuth angles.

travels an additional distance to the target of $4d\sin(\alpha)$ before the closest receiving antenna receives the echo signal. Hence, this results in an additional phase difference at the receiving antennas of $[4\,\phi^a, 5\,\phi^a, 6\,\phi^a, 7\,\phi^a]$. The result is the same as in a setup with one transmitting and eight receiving antennas, which is why the multiple-input multiple-output setup results in so-called virtual antennas. If the antennas are appropriately placed, we can add a virtual antenna for each receiving antenna, which depends on the echo signal of the second transmitting antenna. From a hardware perspective, we still need to integrate more antennas, but the resolution can be increased by a larger factor than by implementing more receiving antennas. However, we need to differentiate between the received signals to assign them to the corresponding transmitting antennas. Therefore, signal modulation, such as binary phase modulation [56] or time division multiplexing [264], is required to generate separable signals.

The multiple-input multiple-output setup can be extended to a 2D antenna array, enabling the measurement of azimuth and elevation angles, as depicted in Figure 2.10. The same principles apply where an additional FFT is required to measure both angles. The resolution of the radar sensor is still limited by the number of antennas and, hence, the size of the radar sensor itself. Therefore, a trade-off between azimuth and elevation resolution is often made. To gain a detailed understanding of the factors influencing the resolution, we outline the dependencies in the following.

2.3.4 Resolution and Ambiguities

The signal processing of radar sensors, including the Doppler velocity, the range, and the angular measurements, provides precise information about the environment for automotive applications. To understand the nature of radar data, we need to consider the measurement resolution and ambiguities in the signal processing. The resolution describes the ability to differentiate between closely related objects in the respective measurement dimensions. Furthermore, the cyclic properties of the FFT can result in ambiguities. Therefore, the sensor parameters and the signal processing directly influence the detection capabilities. To obtain a detailed understanding, we introduce the range resolution, the maximum unambiguous range, the velocity resolution, the maximum unambiguous velocity, and the angular resolution in the following. We focus on the final resolution and ambiguities as these are key aspects of understanding radar data.

There are two major concepts that determine the resolution and ambiguities. First, the Nyquist-Shannon theorem [191], which states that a sampling rate f_{sample} of at least two times the frequency f is required to capture all parts of the signal with a maximum frequency $f_{\max}$ such that $f_{\text{sample}} > 2\, f_{\max}$. Second, the bin resolution of the FFT is given by the ratio of the sampling frequency f_{sample} and the number of sampling points N_{sample}.

For the range measurement, we collect s samples over the duration of a chirp t_c, resulting in a sampling frequency $f_{\text{sample}} = s/t_c$. Since we utilize $N_{\text{sample}} = s$ sampling points for the range-FFT, the resulting frequency bin resolution is $1/t_c$. Two objects are, hence, separable in distance if the difference of the corresponding beat frequencies is separated by at least one frequency bin. Consequently, to resolve two objects, the frequency difference Δf_b must be greater than the bin width of the FFT. We combine Equation (2.9) with the derived formulas, resulting in:

$$\Delta f_b = \frac{2B\,\Delta R}{c_0\, t_c} > \frac{1}{t_c}. \tag{2.21}$$

We simplify this inequality to derive the resulting range resolution:

$$\Delta R > \frac{c_0}{2B}, \tag{2.22}$$

which depends on the bandwidth B of the transmitted signal. Hence, radar frequency bands that enable higher bandwidths are preferable. Therefore, the current radar sensors for automotive applications operate in the frequency band of 76-81 GHz because the possible bandwidth is larger compared to other more restricted frequency bands to enhance the range resolution. For example, the resolution of automotive radars with a bandwidth B of 1 GHz is 15 cm.

Besides the resolution, it is important to consider the maximum unambiguous range in signal processing. The maximum unambiguous range depends on the sampling rate of the analog-to-digital converter. Following the Nyquist-Shannon theorem, the sampling frequency has to be larger than twice the maximum frequency in the spectrum $f_b^{\max}$. The resulting maximum unambiguous range is:

$$R_{\max} < \frac{c_0\, t_c}{4B} f_{\text{sample}}^{\text{ADC}}. \tag{2.23}$$

Increasing the chirp duration t_c and the sampling frequency of the analog-to-digital converter $f_{\text{sample}}^{\text{ADC}}$ increases the maximum unambiguous range. However, the maximum range is also restricted by the transmission and reception of the radar signal itself, summarized in the radar equation in Section 2.2. Since the received signal power is reduced by the factor R^4, the difficulty is to separate the received signal from background noise for distant objects. Therefore, the maximum unambiguous range should be close to the physical detection limit so as not to restrict the range further.

To derive the Doppler velocity resolution, we follow the same principles as for the range resolution. The sampling frequency is defined by the chirp duration resulting in $f_{\text{sample}} = 1/t_c$. The number of sampling points N_{sample} equals the number of chirps N_c. Consequently, the bin resolution of the Doppler-FFT is $1/N_c t_c$. Therefore, the radial velocity of two targets is distinguishable if the difference of the corresponding Doppler frequencies Δf_D is separated by at least one frequency bin. Following Equation (2.14), we derive the expression for the Doppler velocity resolution as follows:

$$\Delta f_D = 2\,\frac{\Delta v_r}{c_0}\, f_c > \frac{1}{N_c\, t_c}. \tag{2.24}$$

Thus, the Doppler velocity resolution simplifies to:

$$\Delta v_r > \frac{c_0}{2 f_c\, N_c\, t_c}, \tag{2.25}$$

where we can summarize the duration of the sequence of chirps $T_{\text{seq}} = N_c\, t_c$. Therefore, the Doppler velocity resolution depends on the carrier frequency f_c, the number of chirps N_c, and the chirp duration t_c. To improve the velocity resolution, we can increase the carrier frequency, increase the number of chirps N_c, or increase the chirp duration t_c. The resulting velocity resolution for an automotive radar with a carrier frequency of 79 GHz and a sequence duration of 10 ms is, for example, $\Delta v_r = 0.19$ m/s. However, different approaches exist to further increase

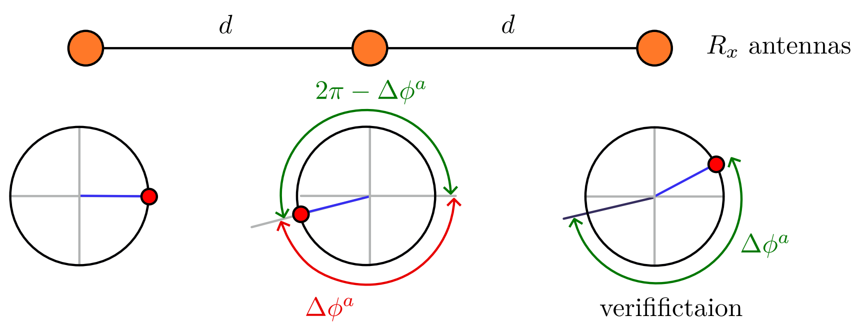


Figure 2.11: The result of the angle-FFT can be displayed as a rotating phasor. The angle estimation is unambiguous as long the phase shift is smaller than π.

the range and the velocity resolution, such as the fast orthogonal search [130] to estimate frequency components at a higher resolution.

The signal processing based on the FFT inherently imposes a maximum unambiguous Doppler velocity. Following the Niquist-Shannon theorem, the sampling frequency $f_{\text{sample}} = 1/t_c$ must be twice as big as the maximum Doppler frequency ensuring $f_{\text{sample}} > 2\, f_{D,\text{max}}$. Hence, we can derive the maximum unambiguous Doppler velocity by substituting $f_{D,\text{max}} = 1/2\, t_c$ in Equation (2.14) resulting in:

$$v_{r_{\text{max}}} < \frac{f_{D,\text{max}}\, c_0}{2\, f_c} \tag{2.26}$$

$$v_{r_{\text{max}}} < \frac{c_0}{4\, f_c\, t_c}. \tag{2.27}$$

This aligns with the explanation that to cover higher frequencies resulting from higher velocities, the sampling frequency, determined by $1/t_c$, must be higher, resulting in a smaller chirp duration. The maximum unambiguous velocity of modern automotive radar sensors often lies in the range of $[-25, 25]$ m/s. Since oncoming traffic easily exceeds the unambiguous velocity, signal processing often incorporates counter measurements, such as tracking algorithms [120], to verify the correct velocity. The velocity measurements are derived over multiple time steps to confirm the actual radial velocity by correlating the motion and previous positions.

The signal processing to calculate the angle depends on the phase shift between the individual antennas, as explained in Section 2.3.3. We determine the unambiguous angle based on the requirement that the assignment of the phase shift to the corresponding angle is clear, which is given as long $\Delta\phi^a < \pi$. If the phase difference is larger than π, we cannot unambiguously resolve the angle because a positive or negative angle from the boresight might cause the same phase difference, as depicted in Figure 2.11. As a result, following Equation (2.19), the largest unambiguous angle $\alpha_{\max}$ is:

$$\alpha_{\max} < \sin^{-1}\left(\frac{\lambda}{2d}\right). \tag{2.28}$$

The unambiguous angle depends on the spacing of the antennas where the spacing of $d = \lambda/2$ results in the maximum unambiguous angle of $\alpha_{\max} = \pm 90°$. Therefore, the spacing of the antennas is important for the maximum unambiguous angle, which also influences the dimensions of the radar.

For the angular resolution, the signal processing includes the angle-FFT, and hence, the spatial sampling frequency influences the resolution, as explained before. Two objects with an angle of arrival of α and $\alpha + \Delta\alpha$ result in a phase difference of:

$$\Delta\phi^a = \frac{2\pi d}{\lambda}(\sin(\alpha + \Delta\alpha) - \sin(\alpha)), \tag{2.29}$$

according to Equation (2.19). For small $\Delta\alpha$, we can use the following approximation:

$$\sin(\alpha + \Delta\alpha) - \sin(\alpha) \approx \cos(\alpha)\Delta\alpha. \tag{2.30}$$

According to this approximation, we derive the expression for $\Delta\phi^a$ as:

$$\Delta\phi^a \approx \frac{2\pi d}{\lambda}\cos(\alpha)\Delta\alpha. \tag{2.31}$$

The Nyquist-Shannon theorem must be fulfilled to separate the two signals, resulting in $\Delta\phi^a > 2\pi/N_{R_x}$, where N_{R_x} is the number of receiving antennas. The resulting angular resolution is:

$$\Delta\alpha > \frac{\lambda}{N_{R_x}\, d\, \cos(\alpha)}, \tag{2.32}$$

where the resolution of the angle depends on the angle itself. The highest resolution is achieved when $\alpha = 0°$. For the specific angles, the resolution further depends on the antenna size given by $N_{R_x}\, d$. Assuming the spacing of $d = \lambda/2$ for the maximum coverage, the resolution is proportional to the number of receiving antennas. Therefore, we need to increase the number of receiving antennas to increase the angular resolution. As a result, the maximum antenna size within autonomous vehicles restricts the maximum resolution. However, besides the number of physical antennas, virtual antennas are included in the multiple-input multiple-output setup, enhancing the effective number of antennas and making the advanced setup valuable. To further improve the angular resolution, algorithms such as ESPRIT [178], 2D ESPRIT [284], and MUSIC [185] exist. The angular resolution of the FFT does not depend on the distance. However, the spatial resolution decreases with the distance to the target because of the increasing beam width as introduced in Section 2.2. The final angular resolution of the radar depends on both the distance and the angle.

Different aspects need to be considered to mitigate ambiguities and improve resolution within each measurement dimension. Automotive radar sensors vary widely in design and parameter setups, and hence, the data is versatile with different resolutions and ambiguities. Compared to LiDARs, the range and angular resolution are much lower, leading to less precise information and making the interpretation of radar data more challenging.

2.3.5 Clutter and Noise

Clutter and noise are key challenges in radar signal processing because they degrade the overall system performance. Noise has various sources, such as sensor noise and interference of signals. The term clutter [198] refers to all unwanted radar returns that may limit the performance of the desired application, including multi-path propagation. The term clutter varies with applications such as weather, surveillance, and automotive radars, considering different radar returns as clutter. For automotive applications, for example, ground reflection can be considered as clutter. However, ground reflections can be important for free space detection and the identification of static parts of the environment. We introduce the most important sources of noise and clutter for our targeted applications in the following.

Each sensor measurement is affected by noise [211], which is usually a random, unwanted disturbance within the original signal. Noise comprises internal noise, such as thermal noise, and environmental noise, including interference with other systems. Random fluctuation in the signals negatively influences the signal processing. Therefore, noise can lead to false positives and false negatives by increasing or decreasing the signal power. Consequently, noise affects range, Doppler, and angular measurements, posing challenges to signal processing.

Radar signals which interact with an object are reflected, refracted, or scattered as mentioned in Section 2.2. The reflected radar signal is often not directly backscattered to the radar sensor itself. These signals interact further with objects within the surrounding of the vehicle, causing multiple reflections before being scattered back to the radar sensor, resulting in so-called multi-path propagation. Metal surfaces such as guardrails, fences, and gates are good reflectors that can cause strong reflections from other objects, resulting in ghost detections. The reason is that we can only determine the incident angle of the received signal and not the exact propagation. As laid out earlier, the signal processing estimates an object in the direction of the received echo signal. Due to the correspondence between the time of flight from emitting the signal and receiving the echo, the

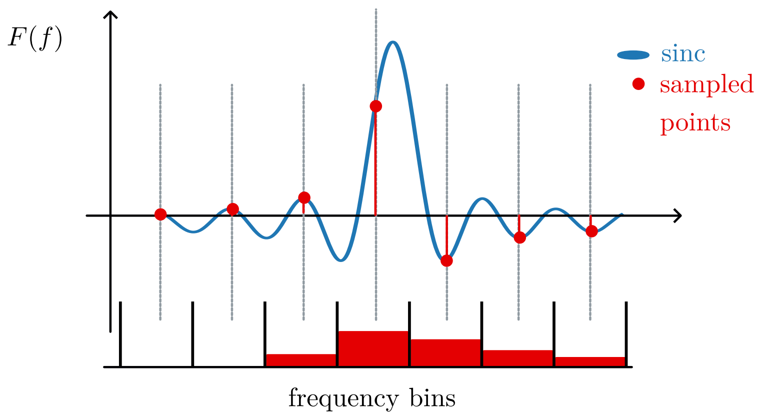


Figure 2.12: Signal processing noise for the fast Fourier transform arises from side lobes and the misalignment between the beat frequency and the center of the frequency bin. The neighboring bins include noise and do not correspond to real targets.

estimated distance is prone to error because only the direct path of a signal is considered in the signal processing, leading to clutter. On the one hand, an issue arises from the fact that these ghost objects have properties similar to the actual detected object, such as the Doppler velocity or radar cross section values, making their identification challenging. On the other hand, multi-path reflections can have benefits, such as the detection of occluded objects, and thus provide additional information to enhance scene understanding.

The radar signal processing based on the FFT is another source of noise. To highlight this issue, we visualize the Fourier transform of the mixed signal from a detected object in Figure 2.12. As explained in Section 2.3, the Fourier transform of the beat frequency leads to the sinc function, denoted as $\text{sinc}(x)$. The main lobe is the frequency we are interested in since it relates to the corresponding information we try to extract, such as the range. However, the main lobe often does not correspond perfectly with the resolution of our frequency bins, resulting in detection in neighboring bins. Additionally, the side lobes can result in detections that do not correspond to actual targets. Especially if more than one reflected signal is present, interference can result in higher side lobes, making differentiation even more difficult. The problem is more severe if one of the reflected signals is strong, making the identification of true targets challenging. As a result, the bins include noise and clutter, and only a fraction relate to true targets. This noise directly results from the limitations of the signal processing itself.

Additionally, the interference of multiple signals can lead to amplification or destruction of the echoes, which can, for example, increase the power but also lead to missing detection. The ambiguities introduced in Section 2.3.4 can lead to errors and may persist in the final radar data if the signal processing does not resolve the correct values. As a result, radar data contains noisy and ambiguous information, requiring optimized algorithms to achieve good performance.

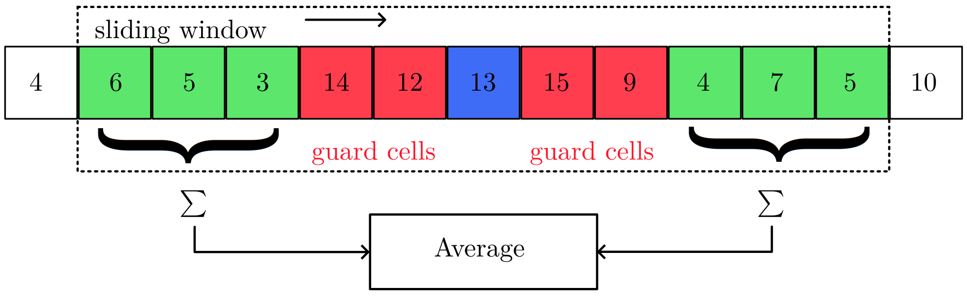


Figure 2.13: Example of the cell-averaging constant false alarm rate algorithm. We evaluate the cell under test (blue), surrounded by the guard cells (red), to estimate the level of noise from the neighboring cells (green) in a sliding window approach.

2.3.6 Radar Point Cloud Extraction

The result of the radar signal processing is a multi-dimensional data cube that includes the range, the Doppler velocity, and the angle estimation. However, the data cube covers the whole field of view, and as introduced in the previous section, most of the cells are occupied by noise instead of true targets. Therefore, we process the data cube to reduce the data stream, focusing on the essential radar data, which is necessary for downstream tasks such as scene understanding. To filter the data and extract point-wise detection, the most common algorithm is the constant false alarm rate [175].

The extraction aims to identify peaks within the data that have a high amplitude compared to background noise since these likely correspond to radar returns from real objects. One approach to setting a detection threshold is to reduce the threshold with increasing distance, accounting for attenuation effects described in Section 2.2, where the received power decreases proportionally to R^4. However, the noise levels vary, and more advanced algorithms are required to identify the radar targets.

The cell-averaging constant false alarm rate [10], for example, evaluates the cells of the data cube individually in a sliding window approach depicted in Figure 2.13. We evaluate the cell under test against the noise level around the cell. To cover the broad peaks of the main lobe, the direct neighboring cells, the so-called guard cells, are excluded from the evaluation of the noise level to prevent incorrect estimates of the noise. The other cells are then averaged to calculate the noise level within this local area. The cell under test is extracted as a target if the peak is higher than the expected noise level of the neighboring cells. As the name already states, there are still false alarms within the extracted targets, resulting in noisy detections. This issue arises from strong multi-path reflections, interference, and sensor noise, as stated in Section 2.3.5.

The results of the constant false alarm rate algorithm are the extracted peaks, which can be transferred into point clouds. Each peak typically corresponds to one point. The points are represented within the polar coordinate frame of the respective sensor because the results of the FFTs are the range, the azimuth, and the elevation information. A two-dimensional antenna array is required to measure the elevation of the object. Consequently, sensors with a one-dimensional antenna array only measure the azimuth angle and the range, providing 2D spatial information.

A common approach is to transfer these polar point coordinates into Cartesian coordinates x^C, y^C, and z^C within the radar sensor frame. Depending on the measurement dimension, we differentiate between different radar sensors. The radar sensors that cover 3D coordinates are called 4D radar sensors, whereas sensors that do not measure elevation angles are called 3D radar sensors, where the additional dimension covers the Doppler velocity. Furthermore, each point, often referred to as a target or detection, usually includes radar cross section values. These values provide information about the shape, material, and size of objects, which influence their detectability, as explained in Section 2.2. The resulting radar point cloud provides rich information about the environment. However, the field of view of individual sensors is limited. To comprehensively cover the surroundings of the vehicle, integrating multiple sensors is essential for providing the necessary information required by advanced driving functions. In the next section, we introduce the concept of combining point clouds of individual sensors to create a unified representation of the environment.

2.3.7 Automotive Radar Data

Automated driving functions require precise information about the environment. Since the field of view of individual sensors is limited, multiple sensors are required to cover the surroundings. The extracted radar point clouds of the individual sensors are represented in the sensor coordinate system, which limits consistent spatial interpretation. Therefore, we need to transform the radar point clouds into a common coordinate system to combine and process data from multiple sensors and ensure spatial alignment, as depicted in Figure 2.14. Moreover, we can utilize ego-motion measurements to compensate for the motion of the vehicle itself, ensuring more accurate velocity estimations. We introduce both concepts in the following.

The two central approaches to combine data within one coordinate system are the transformation into a car coordinate system and a global coordinate system. The origin of the car coordinate system is normally placed at the center of the rear axle, with the x-axis pointing in the forward direction. The y-axis points to the left of the vehicle, resulting in a z-axis pointing upwards for a right-handed coordinate system. The origin of the global coordinate system is often aligned with the car coordinates system after the start of the vehicle. The car coordinate system then moves through the global coordinate system. We determine the position of the vehicle within the global coordinate system by the odometry and the information from the global navigation satellite system.

In general, the transformation from one coordinate system into another can be done by homogeneous transformation where the transformation matrix H is defined as:

$$\mathsf{H} = \begin{pmatrix} & & & t_x \\ & \mathsf{R}_{\text{rot}} & & t_y \\ & & & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{2.33}$$

where R_{rot} is the product of the rotation matrices for the roll, the pitch, and the yaw angle. The advantage of the homogenous transformation is that it combines translation and rotation in a single matrix, enabling the combination of multiple transformations through matrix multiplication. We can derive the transformation matrices from the sensor coordinate system and the car coordinate system by calibration. Furthermore, global navigation satellite system data, online simultaneous localization and mapping algorithms [2, 15, 28, 266], and wheel encoders can be combined to derive the transformation from the car coordinate system into the global coordinate system. Moreover, the sensors provide the transformation between successive poses in the trajectory of the vehicle enabling accurate temporal data alignment.

The representation within the car coordinates system is fundamental to derive a meaningful understanding of the sensor data. Additionally, the transformation between the individual scans in the global coordinate system is important to align temporal scans because the car moves over time. Therefore, precise information about the individual transformations is essential to achieve valuable scene understanding.

The global position information and the odometry information typically include the speed in the driving direction and the yaw rate around the center of the rear axle. The information about the movement of the vehicle is valuable to identify the moving and static parts of the environment because ego-motion results in Doppler velocity in the radar signal processing. We compensate for ego-motion to generate a point cloud in which all static points ideally have a Doppler velocity of zero, while moving objects, such as cars, retain their true radial velocity. For the basic derivation of the ego-motion compensated Doppler velocity, we include the yaw rate $\dot{\psi}$ and the velocity in the x-axis of the car v^x_{car}. The radial velocity of static objects is the projection of the velocity of the sensor in the radial direction of the target. We first need to derive the sensor velocity v^x_{sens} and v^y_{sens} based on the velocity of the vehicle. The sensor velocity depends on the offsets $x^{\text{off}}_{\text{sens}}$

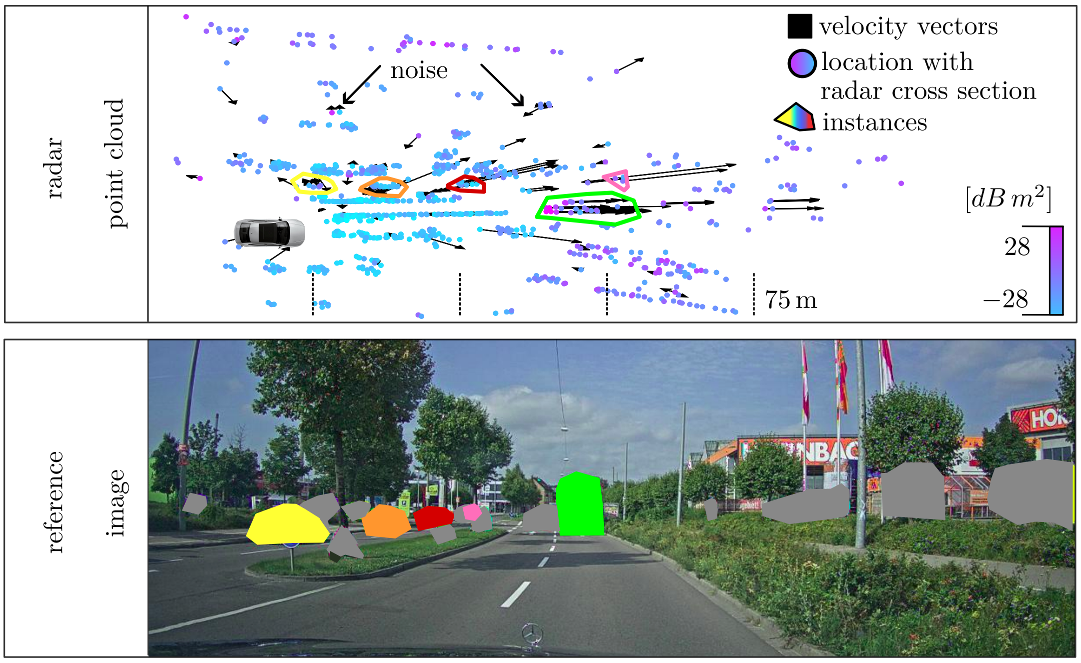


Figure 2.14: Overview of real-world radar data of four radar sensors combined within one common coordinate system. The reference image illustrates the scene and includes privacy-preserving colored masks. In the image of the point cloud, the points are colored according to the radar cross section values. The polygons correspond to the objects. The colors of the objects in the images correspond if the object is visible. Data from the RadarScenes[189] dataset.

and $y^{\text{off}}_{\text{sens}}$ of the sensors to the origin of the car coordinate system. We use the calibration information to obtain the values and calculate the sensor velocities, resulting in:

$$v^x_{\text{sens}} = v^x_{\text{car}} - y^{\text{off}}_{\text{sens}}\,\dot{\psi}, \tag{2.34}$$

$$v^y_{\text{sens}} = x^{\text{off}}_{\text{sens}}\dot{\psi}, \tag{2.35}$$

where the x- and y-axis are parallel to the car coordinate system. The velocity of the individual points is derived by subtracting the apparent radial velocity of the car v^{ego}_r from the measured Doppler velocity $v^{\text{comp}}_r = v_r - v^{\text{ego}}_r$. Since the points appear at different azimuth angles α, we have to calculate the compensation for each point separately, projecting the ego-motion onto the radial velocity. Furthermore, we need to consider in the calculation if the sensor coordinate system is rotated by an angle β in comparison to the car coordinate system. The projected radial velocity for the individual points is given by:

$$v^{\text{ego}}_r = -(v^x_{\text{sens}}\cos(\alpha+\beta) + v^y_{\text{sens}}\sin(\alpha+\beta)), \tag{2.36}$$

where the minus accounts for the fact that the radar moves toward the static object. However, the compensation is often not perfect, and the resolution of the azimuth measurement influences the final result. Furthermore, the radar point cloud includes noise, which, for example, are points that have a Doppler velocity vector unrelated to actual objects as visualized in Figure 2.14. For simplicity, we refer to the ego-motion-compensated Doppler velocity as v in the remainder of this thesis.

The resulting radar point cloud of multiple sensors provides the necessary information about the surroundings, including ego-motion compensated Doppler velocity and radar cross section values. However, these point clouds are sparse, include noise, and have limited resolution, posing challenges for processing, and traditional heuristic-based methods often fall short of addressing the complexity of signal processing. To overcome these limitations, advanced algorithms are required to harness the potential of radars and ensure reliable scene understanding. In the following chapter, we introduce the basics of machine learning that are essential to this thesis.

Chapter 3

Fundamentals of Deep Learning

Machine learning drives development across various industries, leading to advancements from natural language processing up to autonomous driving [113]. The ability to utilize large amounts of data to learn patterns from data and infer the rules themselves without explicit programming is key for successful applications of machine learning approaches to solve dedicated tasks [143]. Data-driven algorithms are able to solve difficult tasks that are too complex for rule-based systems, including human-designed heuristic methods. Therefore, state-of-the-art scene understanding approaches use machine learning algorithms utilizing vast amounts of sensor data to address challenging tasks such as the identification of objects and instances within point clouds. We introduce an overview of machine learning techniques in the following to understand the foundations of this thesis.

Machine learning algorithms can be categorized into supervised learning, unsupervised learning, and reinforcement learning methods, depending on the task and application. We focus on supervised learning techniques [148], where the training data $\mathcal{D}$ consists of input features $\mathbf{x}_i$ and output values $\mathbf{y}_i$, resulting in the dataset $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i) \mid i = 1, 2, \ldots, N_D\}$ for N_D input-output pairs. The ground truth mapping of the input and output values combines the input data with the corresponding output labels. For example, to segment a vehicle within an image or point cloud, a human annotator selects the pixels or points corresponding to the respective object. Machine learning aims to derive the desired mapping from the given data. Hence, the model infers a function from the labeled dataset during the training. The goal is to solve the desired task and learn the accurate mapping to perform well on unseen data, resulting in a good generalization. To evaluate such behavior, we split the labeled dataset into three parts: the training set $\mathcal{D}^{\text{train}}$, the validation set $\mathcal{D}^{\text{val}}$, and the test set $\mathcal{D}^{\text{test}}$. We train the algorithm on a training dataset and evaluate the performance of the adaptations of our approach during the development on the validation set to derive advanced algorithms. The final performance evaluation is conducted on the unseen test dataset. In supervised learning, we differentiate between classification and regression approaches [24]. For classification, the outputs $\mathbf{y}_i$ are discrete labels or categories, such as the corresponding class of a pixel or a point, whereas for regression methods, the outputs are continuous numerical values. However, both methods require an annotated training dataset to derive the mapping function for the desired task.

Conventional machine learning algorithms first transfer the raw input data into a suitable internal representation, often called a feature space, where individual training samples are feature vectors, which the system then utilizes to predict the output. Therefore, human experts design rule-based features to extract meaningful representation from the input data, including methods such as clustering of point clouds, which hopefully are useful for the algorithm to solve the desired task and generalize well. However, the performance is limited due to hand-crafted feature engineering and the rule-based design of the feature representation, which requires domain knowledge to capture expressive representations [146]. From the early days, research focused on replacing hand-engineered feature extraction with algorithms that automatically learn suitable representations from the data itself and leverage the advantages of data-driven methods.

Inspired by the theory of biological networks, Rosenblatt [177] proposed the perceptron, which uses mathematical functions to process the input data and directly predict a binary output. The perceptron is one of the first methods to utilize data to infer the rules directly from the input to solve the given task by minimizing the difference between the desired and actual output. However, the representation learning capability of these early approaches [177] is limited, and hand-crafted features are often mandatory to achieve acceptable performance. Consequently, research focused on advanced algorithms that overcome the limitations and improve accuracy. Major breakthroughs, including the backpropagation of errors to calculate the gradients and update the parameters of the algorithms [114, 115, 179, 180], the introduction of dedicated architectures [80, 116], and optimized training methods [78, 79] paved the way for so-called neural networks. Neural networks consist of hierarchically organized simple processing units, the neurons, that extend the capabilities of the perceptrons and are effective in directly learning suitable representations from the given input data. As a result, these approaches overcome the shortcomings of rule-based approaches and outperform other machine learning algorithms [78] and even surpass humans in large-scale image recognition [75, 181]. Consequently, these methods became the primary solution for various tasks, including perception algorithms. To derive a detailed understanding of the architecture of the methods, we present the basic concepts of neural networks in the following section.

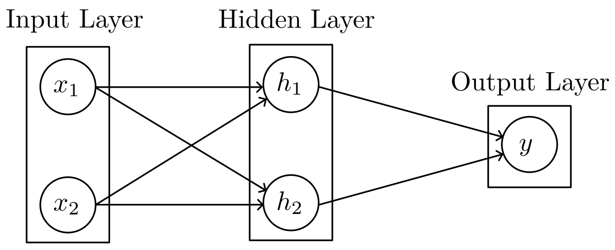


Figure 3.1: Illustration of a simple neural network with two neurons h_1 and h_2 to extract the feature from the two input values x_1 and x_2. The data is processed to obtain the prediction y_1 in the output layer.

3.1 Neural Networks

Neural networks are machine learning models that process the raw input data and derive meaningful hierarchical representations to solve complex tasks. Internal parameters are updated during the training process to learn the mapping between inputs and outputs. The foundation for the development of neural networks includes the pioneering work of Rumelhart et al. [179], which introduced the multi-layer architecture [179] consisting of multiple perceptrons and non-linear functions to extract suitable internal representations. This groundbreaking work is the foundation of modern neural networks with hierarchical architectures, including multiple layers, to extract increasingly abstract and complex features to solve non-linear real-world tasks. A neural network consists of an input and output layer and at least one hidden layer, as illustrated in Figure 3.1. The input layer operates as an interface and provides the data to the neurons in the hidden layers. The neurons are the central building blocks, performing the computations of the network and include the parameters that are adjusted during training to map the input to the desired output. The output layer produces the final predictions of the neural network. The architecture of the neural network defines the interconnections of the layers and the neurons. The combination of the architecture and the processing within the individual neurons defines the function of the neural network. To understand the overall concept and derive a solid understanding, we start by introducing the calculation for a single neuron.

For simplicity, we assume the processing of the inputs $x_1 \in \mathbb{R}$ and $x_2 \in \mathbb{R}$ to determine a single output value $y_1 \in \mathbb{R}$, as visualized in Figure 3.2. We first multiply the input values with the corresponding weights $w_1 \in \mathbb{R}$ and $w_2 \in \mathbb{R}$ and add the bias $b \in \mathbb{R}$ before passing the result through a non-linear activation function $a : \mathbb{R} \to \mathbb{R}$ to determine the output. To capture diverse representations within the network, the layers incorporate multiple neurons. We can summarize the operation of the layer l consisting of multiple neurons by efficient matrix-based computations. We multiply the input vector $\mathbf{x}^{l-1} = [x_1, \ldots, x_{d_{in}}]^\top \in \mathbb{R}^{d_{in}}$, which

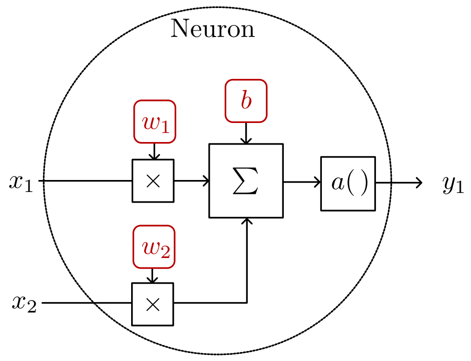


Figure 3.2: Illustration of a single neuron with the inherent weights w_1 and w_2 and the bias b to process the two inputs x_1 and x_2. The sum of the linear operations is followed by a non-linear activation function $a : \mathbb{R} \rightarrow \mathbb{R}$ to determine the output.

is the output of the previous layer $l-1$, with the weight matrix $\mathbf{W}^l \in \mathbb{R}^{d_{in} \times d_{\text{out}}}$ and add the bias vector $\mathbf{b}^l \in \mathbb{R}^{d_{\text{out}}}$, where the weight matrix and the bias vector summarizes all individual weights and biases of the respective neurons in the layer l. The bias vector is important to shift the activation function and extend the learning capabilities of the layer. The non-linear activation function a is applied element-wise to derive the output features $\mathbf{x}^l \in \mathbb{R}^{d_{\text{out}}}$, resulting in:

$$\mathbf{x}^l = a\left(\mathbf{x}^{l-1}\mathbf{W}^l + \mathbf{b}^l\right). \tag{3.1}$$

The non-linear property of the activation function is fundamental since linear functions can only represent linear transformations. Therefore, a cascade of linear functions within multiple layers can be represented by a single affine function, resulting in the collapse of the network into a single linear transformation incapable of solving complex real-world tasks. Two popular activation functions [62] are the sigmoid function defined as $f(x) = (1 + \exp(-x))^{-1}$, the rectified linear unit specified by $f(x) = \max(0, x)$. Additionally, state-of-the-art networks use a modification of the rectified linear unit activation function, such as the Gaussian error linear unit [77] to improve the generalization performance.

The non-linear activation functions in the hidden layers are fundamental for regression and classification tasks. However, to build the neural network for the specific task, the output layer utilizes different activation functions to derive continuous values for regression and discrete values for classification. For regression tasks, the linear activation function is convenient for predicting continuous values. For classification, we aim to obtain a probability distribution over the classes to map the input data to the corresponding output. Therefore, we want to have for each class $1, \ldots, C$ a probability p_i where $\sum_{i=1}^{C} p_i = 1$ and $0 \leq p_i \leq 1$. State-of-the-art networks use the softmax function that exactly does this. The softmax function σ processes the input vector $\mathbf{z} = (z_1, \ldots, z_C)$, to derive the output vector $\mathbf{s}$ resulting in $\sigma : \mathbb{R}^C \rightarrow \mathbb{R}^C$, i.e. $\sigma(\mathbf{z}) = \mathbf{s}$ where the individual components s_i

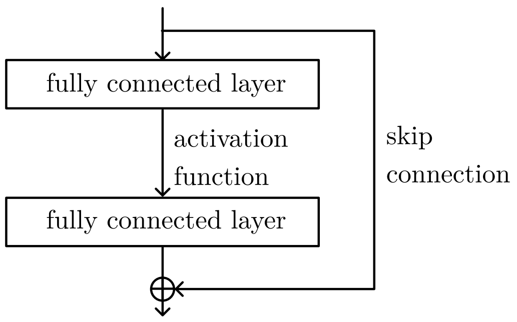


Figure 3.3: Architecture of a residual building block. The features of the skip connection are added to the processed output features of the fully connected layers.

are calculated as follows:

$$s_i = \frac{\exp(z_i)}{\sum_{j=1}^{C} \exp(z_j)}. \tag{3.2}$$

The output vector **s** can be transformed into hard class prediction by the $\arg\max$ function, which selects the class based on the highest probability. As a result, we can map the input values to a class prediction using a neural network. The overall network with multiple layers can be summarized as a function f to process the input $\mathbf{x}$ to derive the desired output $\mathbf{y}$ resulting in $\mathbf{y} = f(\mathbf{x}; \boldsymbol{\theta})$ where $\boldsymbol{\theta}$ are the parameters of the network, including the weights and the bias of all neurons within all layers. The selection of the number of parameters is important because it influences the capacity to learn complex patterns in the data. To scale the parameters, we can adjust the number of neurons within the individual layer, which is known as the width of the model or the number of hidden layers resembling the depth of the network. Therefore, the scaling of the depth and the width plays a central role in the architecture design of a neural network. Theoretical results support the development of deep architectures by showing that deep representation can be exponentially more efficient than shallow architectures that are insufficiently deep [18, 19, 71]. The idea of deep architectures is that we extract slightly more abstract features within the consecutive layers to derive detailed representations to solve complex tasks. However, to leverage the potential of deep architectures, different architectural adaptations [23, 75] are important. One of the fundamental developments for deep architectures is residual building blocks with skip connections [75], depicted in Figure 3.3. The skip connection passes information across layers within the residual block to preserve the features and enable the network to combine the information of multiple representations.

Besides the architecture of the individual building blocks, the interconnections of the layers influence the overall function of the neural network. Different approaches exist, e.g., the fully connected neural network connects all neurons of the previous layer with all neurons of the following layer. We emphasize that, while the simple architecture introduced in this section serves to illustrate the concepts, numerous advanced neural network architectures exist [113]. In Section 3.3, we present specific architectures for scene understanding that are relevant to this thesis.

The architecture is important to learn representative features. However, from the architecture itself, we do not derive a good mapping of the input and output values with a good generalization. Therefore, we need to optimize the individual parameters of the network to reduce errors in the mapping. The optimization of the parameters is performed during training, which makes the training essential to achieve good performance. To understand the difficulties and introduce state-of-the-art approaches, we draw the details of the training of deep neural networks in the next section.

3.2 Training of Deep Neural Networks

So far, we have introduced neural networks as a general class of parametric non-linear functions that map an input to an output depending on the architecture and parameters. The training of the neural network focuses on the key aspects of adjusting the parameters, such as the weights and biases of the neurons, to solve the given task accurately. Therefore, we require information on how to adjust the weights and biases of the neurons to reduce the error of the predictions of the network. The resulting training is an optimization procedure to learn the mapping $\mathbf{y} = f(\mathbf{x}; \boldsymbol{\theta})$ based on a predefined loss function which measures the error between the ground truth and the predictions. Consequently, the training of a supervised method requires an annotated dataset proving the ground truth for a specific input and the desired output, such as the class labels of a specific point within a point cloud. During the training, the network adjusts the parameters $\boldsymbol{\theta}$ in a way that minimizes the errors between the predictions and the ground truth values and, hence, improves the accuracy of the mapping.

The starting point is to define a loss function that accurately resembles the training target. Therefore, the loss function depends on the task and differs for classification and regression. The primary objective of the training is not only to minimize training errors but also to achieve a good generalization for unseen data. The training criterion functions as a surrogate for the generalization, and thus, an appropriate training criterion and parameterization related to the task is key for good performance. Therefore, the loss $l(f(\mathbf{x}_i; \boldsymbol{\theta}), \mathbf{y}_i)$ measures the error for the prediction and the ground truth value derived from the training dataset $\mathcal{D}^{\text{train}} = \{(\mathbf{x}_i, \mathbf{y}_i) \mid i = 1, 2, \ldots, N\}$, which depends on the neural network architecture and the parameters $\boldsymbol{\theta}$. The cost function $\boldsymbol{\mathcal{L}}$ summarizes the loss for the training dataset resulting in:

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} l(f(\mathbf{x}_i; \boldsymbol{\theta}), \mathbf{y}_i), \tag{3.3}$$

where N is the number of input-output pairs of the training set. For regression, the loss functions utilize a distance measure to calculate the error between the ground truth values and the outputs of the neural network. Standard loss functions are the mean absolute error loss and mean squared error loss. However, the primary goal for supervised learning to enhance scene understanding is the classification of the environment. For classification, the commonly used loss function is the cross-entropy loss, which minimizes the negative log-likelihood of the softmax output. Therefore, it is the maximum likelihood estimate [106] of the parameter given the training dataset. As a result, the cross-entropy loss penalizes predictions with a low probability for the correct class as follows:

$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{|\mathcal{D}^{\mathrm{train}}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}^{\mathrm{train}}} \frac{1}{C} \sum_{i=1}^{C} y_i \log(s_i), \tag{3.4}$$

where s_i are the individual output values of the softmax function $\mathbf{s} = \sigma\left(f(\mathbf{x}; \boldsymbol{\theta})\right)$, see Equation (3.2), and y_i are the individual ground truth values. For the ground truth, we utilize a one-hot encoding, which is a vector of the length of C where each element is zero except for the true class of the respective input. Additionally, optimized loss functions exist, including the focal Tversky loss [1] to deal with class imbalance in the dataset.

The common property of the loss functions is the measurement of the error between the predictions and ground truth value depending on the parameter of the network. The adaptation of the parameters of the network modifies the prediction and the corresponding loss. The minimization of the cost function hence results in an optimized parameter setting, which solves the task more accurately. To update the parameters and converge to an appropriate solution, we utilize gradient-based optimization algorithms [174], which we introduce in the following section.

3.2.1 Gradient Descent Optimization

Training deep neural networks aims to minimize the loss function by adjusting the parameters to achieve a solution that generalizes well to unseen data. Due to the large number of parameters with cascaded non-linear functions in the hidden layer, the loss function is non-convex [24]. The primary challenge of training deep neural networks lies in the cost function, which has an enormous number of local minima, plateaus, and equivalent minima due to non-unique solutions [46, 61], making a global optimization infeasible. Therefore, we adjust the network parameters based on gradient descent optimization. The idea is to take small steps in the direction of the steepest descent, which is the negative gradient of the loss function with respect to the parameters. As a result, we can update the parameters and minimize the loss. This is an iterative process with the goal of converging to a good minimum with high accuracy for the task.

The process starts with the initialization of the parameters to calculate the output of the network in the forward path, given the input from the training dataset. The resulting loss measures the difference between the prediction and the ground truth. We calculate the gradients $\nabla\mathcal{L}(\boldsymbol{\theta})$ of the loss with respect to all the parameters in the network to derive the individual parameter updates to minimize the loss, which is known as backpropagation [179]. Consequently, the backpropagation starts from the output and uses the chain rule to calculate the first-order partial derivatives of the loss function with respect to the individual weights and biases in the network. Based on the partial derivatives, we update all parameters according to the direction of the steepest descent as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma\nabla\mathcal{L}, \tag{3.5}$$

where $\gamma > 0$ is a hyperparameter specifying the step size known as the learning rate. The straightforward application of the gradient descent optimization requires the calculation of the cost function over the whole dataset as introduced in Equation (3.3). Consequently, for large datasets that are commonly used for scene understanding with thousands of training samples, the calculation of the gradient is computationally expensive. Therefore, different methods exist that reduce the number of samples to calculate the gradients efficiently, such as the mini-batch gradient descent [25]. In machine learning, this method is also known as stochastic gradient descent. However, the original stochastic gradient descent algorithm utilizes only one sample, corresponding to one input point cloud and the corresponding outputs, to calculate the gradients, whereas, in the training of deep neural networks, we normally use a mini-batch of B samples. The advantage of the mini-batch gradient descent is that the parameter updates are less noisy compared to the original stochastic gradient descent and, hence, lead to faster convergence and better generalization. In practice, we calculate the parameter updates based on a mini-batch, resulting in:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma\frac{1}{B}\sum_{i=1}^{B}\nabla l(f(\mathbf{x}_i;\boldsymbol{\theta}), \mathbf{y}_i), \tag{3.6}$$

where B is normally much smaller than N samples of the complete training set. As a result, for the training with a finite dataset, we iterate over all mini-batches within the dataset, which is called an epoch. The complete training of the neural networks requires several epochs to converge.

The advantage of the mini-batch compared to the vanilla gradient descent is that it has more frequent but noisier updates. The noisy updates especially help to escape local minima and saddle points. The appropriate selection of the batch size is important, as mini-batch sizes that are too small may limit the convergence. Furthermore, the parameter updates with respect to the loss depend on the central hyperparameter γ, which is why it is crucial to tune the learning rate for good performance [17]. If the learning rate is too large, it leads to oscillation around the minimum, and the training criterion will not converge, resulting in unsatisfactory results. In contrast, a learning rate that is too small leads to very slow learning or even no convergence. Additionally, a constant learning rate, which leads to a good convergence, is challenging to find. Therefore, state-of-the-art approaches utilize a learning rate scheduler that adopts the learning rate during the training process. The idea is to speed up the training in the beginning by a larger learning rate and later enable convergence to a minimum by a smaller learning rate. To enforce such behavior, different learning rate schedulers, including step decay, exponential decay, and cosine decay schedulers [20, 132] exist.

The selection of the learning rate decay depends on different hyperparameters, which need to be defined in advance, and the learning rate is fixed for all parameters. To overcome these limitations, adaptive gradient descent algorithms, including Adam [96] and AdamW [133], leverage the power of adaptive learning rates algorithms to identify individual learning rates for each parameter. The individual learning rate parameters are adjusted based on estimates of the first and second moments of the gradients, leading to faster convergence and higher efficiency.

Gradient-based optimization comes with several challenges, including vanishing and exploding gradients, which both make the adaptation of parameters challenging. Therefore, several techniques are important for the training of deep neural networks. One solution to induce a more stable behavior of the gradients and accelerate training is normalization [182], including batch normalization [88], which normalizes the activation at the specific layer for one mini-batch to enhance convergence. Furthermore, deep neural networks benefit from the residual connection and the rectified linear unit activation function introduced in Section 3.1. This activation function keeps the gradient for positive values intact, and the residual connection enables the gradient propagation into a deeper layer through the skip connection, reducing the problem of vanishing gradients.

3.2.2 Regularization

The optimization of the parameters during the network training does not necessarily lead to satisfactory results. The problem is that the number of parameters influences the learning capacity of the model. A model with too few parameters can fail to learn the desired mapping to solve the task. The model cannot obtain a sufficiently low training error and does not generalize well, which is known as underfitting. Neural networks with a high capacity can learn the mapping, resulting in a low training error. However, they might memorize the training data rather than learn a generalizable solution, especially when the dataset is small, which is known as overfitting. The selection of the model with the best generalization is not trivial. Therefore, the common approach is to use a model with a higher capacity to solve the underlying task and apply techniques to prevent overfitting, collectively known as regularization [105]. The goal is to control the complexity of the model and learn the underlying mapping to solve the desired task accurately. We focus on the most important regularization techniques relevant to the remainder of this thesis. For a comprehensive overview, we refer readers to the excellent surveys about regularization techniques [105, 145].

One of the earliest regularization techniques is the weight norm penalty or weight decay [70, 103, 111, 144]. The underlying principle is that neural networks with large weights tend to overfit since small changes in the input data can result in large changes in the predictions. Therefore, the AdamW optimizer [133] incorporates the weight decay step and the adaptive gradient mechanism separately to utilize the advantage of both.

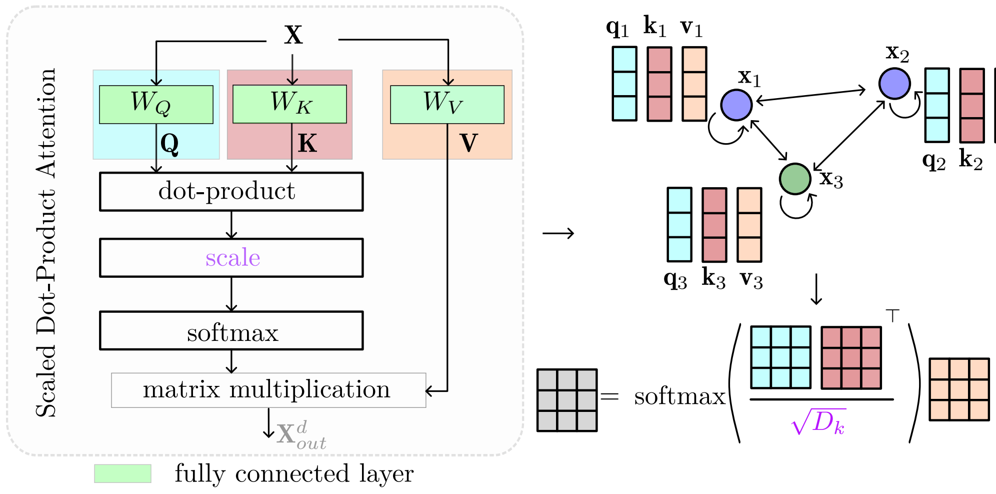
Early stopping [147, 162] is another form of regularization that monitors the training and validation error. Deep neural networks decrease the training error continuously. The validation error decreases first before increasing again due to overfitting to the training set. Therefore, the optimal solution with good generalization is achieved before the validation error increases again. At this point, the difference between the training and the validation error is small, with the best performance on unseen data. The early stopping selects this model as a regularization step.

Large models that are trained on insufficiently large datasets tend to overfit easily. Therefore, data augmentation is essential to generate synthetic but realistic training samples to extend the training dataset. Since labeling real-world data is tedious and often requires domain expertise, labeling additional training data comes with huge costs. Therefore, the goal is to augment samples to extend the training data, which represents the actual data distribution, to prevent overfitting and improve generalization. The standard data augmentation [199, 282] techniques include, among others, the rotation, scaling, flipping, and jittering of the input data. The selection of the data augmentations strongly depends on the application because not all modifications are suitable to generate synthetic but reasonable data. For example, flipping the input image in character recognition is not suitable because the label might not be valid anymore, e.g., think of the letters b and d. Therefore, the selection of appropriate modifications is essential to enhance generalization performance. Consequently, the neural network architecture, the training, and specific regularization techniques are important to improve the accuracy.

3.3 Transformers

Fully connected neural networks are the foundation of different successful deep learning models. However, the performance for feature extraction, focusing on local patterns and dependencies, is limited. Several works introduce specialized architectures to derive meaningful representations and solve complex tasks. One example is convolutional neural networks [112], which efficiently capture spatial representations in image data, resulting in successful practical applications such as image classification [102]. Convolutional neural networks are highly effective in extracting features from a local receptive field and exhibit excellent performance in various computer vision applications. However, the capabilities to capture global dependencies and contextual information are limited. Therefore, in recent years, transformer-based methods have achieved remarkable results within various tasks, from natural language processing [26, 212] to computer vision [29, 36, 50, 97, 254] and point cloud scene understanding [65, 160, 187, 196, 242, 262, 271].

Complex tasks require understanding the context, such as spatial and relational information, and learning dependencies within the input data to achieve high accuracy. The core mechanism of the transformer, the self-attention [212] proposed by Vaswani et al., enables the network to weigh the importance of the input data and extract meaningful features to learn dependencies and relationships. For the processing of point clouds, the relationship of which point belongs to the same object or the same class is essential to achieve good performance in scene understanding. The self-attention mechanism comes with various advantages compared to previous state-of-the-art approaches. Two important benefits are that the mechanism is permutation invariant and does not depend on the number of input elements. In comparison, the input of a fully connected neural network introduced in Section 3.1 is a one-dimensional feature vector. Therefore, the input data, such as the features of the point cloud, are flattened into a vector to process the data. The first processing layer of the fully connected neural network requires the same number of neurons as the dimension of the input feature vector. Since the number of parameters within the networks scales quadratically with the number of neurons, the number of parameters grows rapidly for large-scale input data, and the training is challenging [113]. Furthermore, the interconnections of neurons in a fully connected neural network are fixed. Consequently, the neurons focus on the processing of specific features, where the order of the input element matters, and thus, the processing is not permutation



Scaled Dot-Product Attention
X
W_Q
W_K
W_V
Q
K
V
dot-product
scale
softmax
matrix multiplication
X_{out}^d
fully connected layer
$\mathbf{q}_1$ $\mathbf{k}_1$ $\mathbf{v}_1$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$
$\mathbf{q}_2$ $\mathbf{k}_2$ $\mathbf{v}_2$
$\mathbf{q}_3$ $\mathbf{k}_3$ $\mathbf{v}_3$
= softmax
$\sqrt{D_k}$

Figure 3.4: Transformer layer with scaled dot-product attention for the processing of point clouds. The transformer layer processes the point-wise features $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times D}$ to obtain the queries $\mathbf{Q} \in \mathbb{R}^{N \times D_k}$, keys $\mathbf{K} \in \mathbb{R}^{N \times D_k}$ and values $\mathbf{V} \in \mathbb{R}^{N \times D_k}$. The attention scores calculated from the queries and keys weight the information of the values to derive the output features $\mathbf{X}^d_{\text{out}} \in \mathbb{R}^{N \times D_k}$.

invariant. However, point clouds are unordered data, and the number of points can vary, limiting the performance of fully connected networks. Therefore, the processing of point clouds requires advanced algorithms such as transformers to solve complex tasks. We first introduce the transformer layer, including the self-attention mechanism in the following, as it is the essential building block of the transformers.

The transformer layer first encodes the inputs into three representations: the queries, keys, and values. The idea is similar to a retrieval process, where the queries are mapped against the set of keys. We retrieve the information associated with the keys in the form of values that correspond to the best mappings. The goal is to learn meaningful internal representations and relationships based on the attention mechanism such that points belonging to the same class exchange information and derive similar features, resulting in reliable classification.

The calculation starts with the encoding of the input features $\mathbf{X}$ into the queries $\mathbf{Q} \in \mathbb{R}^{N \times D_k}$, keys $\mathbf{K} \in \mathbb{R}^{N \times D_k}$, and values $\mathbf{V} \in \mathbb{R}^{N \times D_k}$, visualized in Figure 3.4. We assume that the input is a point cloud with N points and the resulting point-wise features $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N \times D}$. We derive the encoding of the input features as follows:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \qquad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \qquad \mathbf{V} = \mathbf{X}\mathbf{W}_V, \tag{3.7}$$

where the matrices $\mathbf{W}_Q \in \mathbb{R}^{D \times D_k}$, $\mathbf{W}_K \in \mathbb{R}^{D \times D_k}$, and $\mathbf{W}_V \in \mathbb{R}^{D \times D_k}$ are learned linear projections or the weight matrices of fully connected layers. The advantage of encoding the features of individual inputs in the matrix format is that the processing is independent of the number of points, and we can parallelize the computations. To relate the queries and keys, the scalar attention calculates the dot-product between the queries $\mathbf{Q}$ and keys $\mathbf{K}$ to calculate the attention weights $\mathbf{A}^d \in \mathbb{R}^{N \times N}$, resulting in:

$$\mathbf{A}^d = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_k}}\right), \tag{3.8}$$

where the factor $\sqrt{D_k}$ is used to scale the attention weights, and the softmax function is applied row-wise. The scaling is important for the processing with a softmax function because the dot product grows large in magnitude for high-dimensional key and query vectors, pushing the softmax into regions with small gradients, resulting in vanishing gradients that can slow down the training process, as explained in Section 3.2.1. The softmax function converts the scaled dot-product into the attention weights, which are probability values that connect the individual points within the point cloud. Consequently, the weights encode the relationship of the individual points within the point clouds. We calculate the output features $\mathbf{X}_{\text{out}} \in \mathbb{R}^{N \times D_k}$ by simple matrix multiplication of the attention weights and values to extract meaningful features and aggregate the information. The inputs to the transformer layer are the features of all points where the number of points does not change during the processing within the transformer layer. Hence, the dot-product attention obtains the relationship of all points to each other, and we derive updated features for the individual points.

The matrix $\mathbf{A}^d$ relates all the queries and the keys and derives an overall understanding of the input, independent of the set of input data and regardless of the order. However, an understanding of the local context, such as the relationship of nearby points, is important for scene understanding in order to relate objects. Therefore, the position information obtained from the cartesian coordinates is key to recognize spatial relationships within the input data. Transformers leverage positional encodings [212, 271] to integrate the position information and adapt the attention to local structures. The standard encodings are manually crafted based on sine and cosine functions or range values [270]. To directly integrate the provided 3D information of the point clouds, the positional encoding leverages the relative distance of the points to derive meaningful features and relationships. The input data therefore uses the position information within the features but also as additional position information $\mathbf{P} = [\mathbf{p}_1, \dots, \mathbf{p}_N]^\top \in \mathbb{R}^{N \times 3}$ of the individual N points. We process the relative positions $\mathbf{r}_{i,j} = \mathbf{p}_i - \mathbf{p}_j, 1 \leq i, j \leq N$ by an multi-layer perception (MLP), including two fully connected layers with weight matrices $\mathbf{W}^d_{R_1} \in \mathbb{R}^{3 \times 3}$ and $\mathbf{W}^d_{R_2} \in \mathbb{R}^{3 \times 1}$ an activation function a such as the rectified linear unit [150], resulting in:

$$\mathbf{R}^d_{i,j} = a((\mathbf{r}_{i,j}\,\mathbf{W}^d_{R_1})\mathbf{W}^d_{R_2}), \tag{3.9}$$

where $\mathbf{R}^d \in \mathbb{R}^{N \times N}$. The goal of processing the relative positions is to have a trainable and parametric positional encoding to learn a suitable internal representation and to extract valuable features to solve the task. The resulting positional encoding $\mathbf{R}^d$ is added to the dot-product of the queries and keys before deriving the final attention weights. The resulting weighting hence includes spatial relationships. The output features, which are the weighted information obtained from the values, result in the following:

$$\mathbf{X}^d_{\text{out}} = \operatorname{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{D_k}} + \mathbf{R}^d\right)\mathbf{V}. \tag{3.10}$$

The attention weights of all points within the position cloud result in the calculation of dependencies in the form of queries and keys from points in distances of more than a hundred meters. The problem of global attention is that the calculations scale quadratically with the number of input points, and hence, the processing leads to unacceptable memory consumption and computational cost. To reduce the calculations and focus on the important information within the individual attention blocks, we limit the self-attention to local areas. We derive the local area for each individual point within the point cloud by the k nearest neighbor (kNN) algorithm. The encoding of the queries, keys, and values remains the same. However, we sample and group the vectors according to the local areas with $k = N_l$ points. The resulting queries and key and values matrices include the local information resulting in $\mathbf{Q}^k, \mathbf{K}^k$, and $\mathbf{V}^k \in \mathbb{R}^{N \times N^l \times D_K}$.

The dot product attention focuses on the weighting of the information based on the encoded queries and keys by a one-dimensional attention weight $\mathbf{A}^d_{i,j} \in \mathbb{R}$. However, the weighting of all feature channels by the same attention weight can limit the ability to capture rich interaction and detailed relationships of the input data. For example, we want to differentiate between static and moving vehicles within the environment. The encoded features about the shape and location of the vehicle can help to derive object information, but for the differentiation, the velocity information is key and should be waited differently. Therefore, vector attention incorporates attention weights for each individual feature channel, resulting in attention weights $\mathbf{A}_{i,j} \in \mathbb{R}^{D_K}$. Consequently, we replace the dot-product in the transformer layer with a relation function such as subtraction, as visualized in Figure 3.5. Since the dimension of the attention weight corresponds to the dimension of the values, we modify the positional encoding to add the information to the individual weightings. Since the weighting of the feature channels comes with additional computational costs, the restriction to local areas is important. We determine $\mathbf{Q}^k, \mathbf{K}^k$, and $\mathbf{V}^k \in \mathbb{R}^{N \times N^l \times D_K}$ by kNN search followed by sample and grouping. We utilize the same indices of the local areas to calculate the positional encoding. Based on the relative positions $\mathbf{r}_{i,j}$, we compute the positional encoding for all points $\mathbf{R} \in \mathbb{R}^{N \times N^l \times D_K}$ by an MLP [271], including

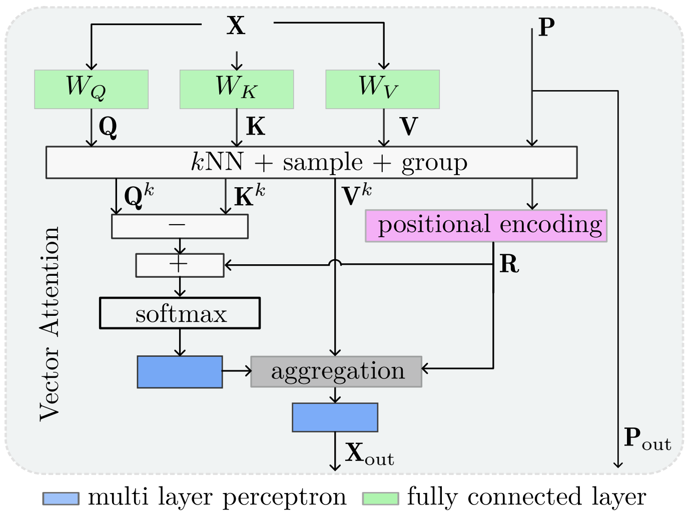


Figure 3.5: The transformer layer with vector attention and positional encoding for local neighborhoods. The transformer layer processes the point-wise features $\mathbf{X} \in \mathbb{R}^{N \times D}$ to obtain the queries $\mathbf{Q} \in \mathbb{R}^{N \times D_k}$, keys $\mathbf{K} \in \mathbb{R}^{N \times D_k}$ and values $\mathbf{V} \in \mathbb{R}^{N \times D_k}$. We sample the encodings according to the local areas determined by kNN search followed by sample and grouping. The relation function of the vector attention is subtraction to obtain an individual weighting of the feature channels. The positional encoding $\mathbf{R}$ is added to the attention weights and the values to include fine-grained position information. The output features $\mathbf{X}^{\text{out}}$ and point coordinates $\mathbf{P}^{\text{out}}$ are passed to the consecutive layer.

two fully connected layers with weight matrices $\mathbf{W}_{R_1} \in \mathbb{R}^{3\times3}$ and $\mathbf{W}_{R_2} \in \mathbb{R}^{3\times D_K}$, batch normalization [89], and rectified linear unit activation function [150]. The processing of the positional encoding adopts the feature dimensions and enables the learnable adaptation of the weighting for each feature channel. The combination of the vector attention in the form of subtraction and the positional encoding results in the attention weights, which we determine as follows:

$$\mathbf{A}_{i,j} = (\mathbf{Q}^k_{i,j} - \mathbf{K}^k_{i,j}) + \mathbf{R}_{i,j}. \tag{3.11}$$

We apply the softmax function to each feature dimension within the local area individually and process the attention weights by an MLP with two fully connected layers, where each layer is followed by batch normalization [89] to derive the final attention weights $\hat{\mathbf{A}}_{i,j}$. The processing of the attention weights by an MLP enables the network to adapt the weighting to capture complex relationships. The weighting controls the aggregation of the information from the values. Instead of directly multiplying the weights with the values, we also add positional encoding, which is an additional feature that can support scene understanding. We calculate the output features by the sum of the element-wise multiplication, indicated by $\odot$, as:

$$\mathbf{X}^{\text{out}}_i = \sum_{j=1}^{N^l} \hat{\mathbf{A}}_{i,j} \odot (\mathbf{V}^k_{i,j} + \mathbf{R}_{i,j}), \tag{3.12}$$

where we aggregate the information within the local areas. The aggregated features $\mathbf{X}^{\text{out}}$ are often processed by additional learned linear projections such as MLPs to derive the final output feature representations of the transformer layer. The output features include the relationship of all points within the local neighborhood. Therefore, compared to dot-product attention, local vector attention reduces the receptive field of the individual point to local areas. The features of all points are updated and enriched with information. Besides the features, the transformer layer outputs point coordinates to calculate the positional encoding within consecutive layers. The self-attention mechanism is essential to extract valuable features and solve complex tasks. However, the local restriction limits the respective information within the individual layers. Therefore, the architecture of the network plays an important role in deriving meaningful hierarchical representations for scene understanding. In the following section, we elaborate on the most common approach, the encoder-decoder architecture.

3.3.1 Encoder-Decoder Architecture

The feature extraction within the transformer layer is the foundation of the strong learning capabilities of transformers. The encoder-decoder architecture is key to deriving hierarchical features and combining the information to solve complex tasks. The encoder first transforms the input into a compact but expressive internal representation. Therefore, the dimension of the input is reduced, which is, in our case, the number of points. To capture detailed information within the reduced number of points, we commonly increase the feature channels. We perform the encoding in consecutive steps within the different levels of the architectures, known as stages $S_1, \dots S_4$, as illustrated in Figure 3.6. The stages consist of at least one transformer block. However, multiple transformer blocks can be stacked within the individual stages to increase the depth of the network. The transformer block includes as a central building block the transformer layer, which is often embedded into two fully connected layers, including a skip connection resulting in a residual block introduced in Section 3.1. The residual connection reduces vanishing gradients for deep neural networks, allowing the gradient to propagate into deeper layers.

The number of points, represented as features and point coordinates, remains the same during processing in the transformer layers. For the downsampling of the point cloud, the encoder commonly uses farthest point sampling to derive a subset of points based on the distance to cover the geometric space evenly. To derive the features for the subsampled point cloud, we utilize kNN search with $k = N^d$ and max pooling [166] to combine the local features. From stage S_L to stage S_{L+1}, most approaches sample $N_{S_{L+1}}$ points from N_{S_L} points where $N_{S_{L+1}} = N_{S_L}/4$. The cardinality of the point cloud, the number of points, is often reduced by a

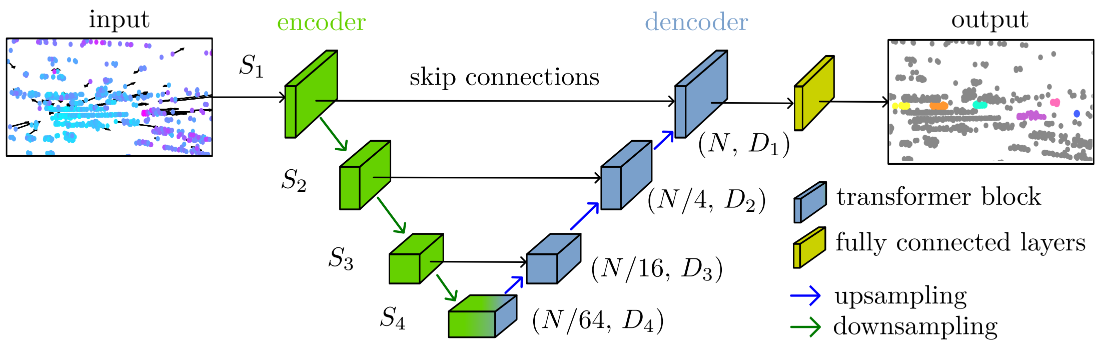


Figure 3.6: Encoder-decoder architecture of a transformer with skip connection to process point clouds. The final fully connected layers process the features and predict the output. The tuples denote the number of points and the feature channels within the individual stages.

factor of 4 [166, 271], resulting in $[N, N/4, N/16, N/64]$ points for four stages, where N is the number of input points. Due to the downsampling of the point cloud, the consecutive transformer layer within the next stage captures a larger area within the point cloud. Therefore, we can extract slightly more abstract features within consecutive layers to derive an overall understanding of the scene. We commonly increase the number of channels D_i within consecutive stages to encode this information.

To solve point-based perception tasks, the encoded features need to be projected to the original number of points. Hence, during upsampling, we need to project the features of the downsampled point cloud onto the higher-resolution point cloud of the encoder provided through skip connections, as visualized in Figure 3.6. The features of the encoder are important to extract a versatile representation combining features from different layers. However, the cardinality of the point clouds differs because the point cloud of the skip connection includes more points compared to the point cloud, which needs to be upsampled. Hence, the difficulty is aligning the information and combining the respective features of the individual points. For the upsampling, the common approach is trilinear interpolation based on an inverse distance weighted average [166]. We determine $k = 3$ neighbors of the points of the encoder of the stage S_{L-1} within the points in the decoder of stage S_L based on the kNN algorithm. For each point, the features of the corresponding neighbors are weighted and added to the features of the encoder stage. The final point-wise features after encoding and decoding include the combined information of the individual stages for each point in the input point cloud. Intuitively, the idea of the encoder is to provide higher semantic information within each encoder stage. The decoder gradually recovers the spatial information within the stages while refining the information for each point. The encoder and decoder form the core of the network architecture, often referred to as the backbone, important for extracting features and generating meaningful internal representations. To derive the final class prediction or continuous values for regression, the architecture includes several fully connected layers that reduce the feature channels to the desired output size, e.g., for classification, the output channels correspond to the number of classes. The training of the transformer then follows the common methods introduced in Section 3.2. Additionally, regularization techniques such as data augmentation play a critical role in training deep transformer architectures.

Transformers offer a robust approach to point cloud processing by combining flexibility, adaptability, and strong feature extraction capabilities to enhance accuracy for point cloud-based scene understanding. However, the processing depends on the specific input data. Hence, dedicated architectures are essential to enhance scene understanding and achieve reasonable accuracy. In the following chapter, we present state-of-the-art approaches for point cloud processing and elaborate on the advantages and limitations of the respective methods.

Chapter 4

Related Work

There is extensive research in the field of autonomous mobility focusing on scene understanding. Spatial and semantic information about the environment is the foundation for many applications, making autonomous driving more safe and reliable. Over the past years, significant advancements have been made in different sensor modalities, such as cameras, LiDARs, and radars, mainly due to deep learning-based approaches. Starting with AlexNet [102] in 2012, neural networks gained attention, surpassing traditional methods in image processing by leveraging data to learn complex relationships as explained in Chapter 3. Spatial understanding especially benefits from depth information. Therefore, there has been tremendous progress in point cloud processing [66, 152, 268], focusing on LiDAR and radar sensors in recent years. The primary focus of research was LiDAR data [22] due to data availability [13], spatio-temporal consistency, and high angular resolution, which enhance human interpretability. However, with the improvements in radar sensor technology [202], open-source datasets [158, 189], and the advantage to work under adverse weather, radar technology gained attention, and learning-based radar signal processing evolved vastly. Since LiDARs and radars primarily provide point-wise data, breakthroughs within the individual modalities can often enhance the overall scene understanding.

We divide this chapter into two sections to structure the work relevant to our approaches in Part I and Part II, respectively. In Section 4.1, we first introduce approaches that focus on point-wise perception tasks, such as segmentation and classification. Section 4.2 builds upon the previously introduced works and utilizes their predictions for tracking and enhancing segmentation performance by additional refinement modules.

4.1 Point Cloud Scene Understanding

The scene understanding for autonomous vehicles includes various tasks, such as semantic segmentation, instance segmentation, scene flow estimation, and panoptic segmentation. The individual solutions for point cloud processing often benefit from each other and contribute to the enhancement of scene understanding. To provide a complete overview and structure the extensive related work, we differentiate between projection-based, voxel-based, point-based, transformer-based, and hybrid approaches. While we target radar-based scene understanding, the research on point-based scene understanding includes a large body of work for LiDAR processing. To provide a complete overview, we discuss methods of point cloud processing [67, 265] for LiDAR and radar data. Furthermore, different methods benefit from each other, and hence, we do not specifically differentiate between the topics covered in this thesis, such as moving object segmentation, moving instance segmentation, or panoptic segmentation, but cover a variety of the research landscapes in the subject of automotive applications. However, we concentrate on point cloud-based methods using LiDAR and radar data, and for an overview of camera-based approaches, we refer to the following excellent survey [142].

4.1.1 Projection-Based Approaches

The early approaches for point cloud processing are projection-based methods [33, 43, 95, 100, 107, 141, 229, 204, 201] projecting the point clouds into two-dimensional grid representations. Hence, these input representations can be processed by successful convolutional neural networks [116]. As a result, the advancements and efficacy of these methods in image processing are directly applicable to point cloud processing without further adaptations. One of the first methods proposed by Wu et al., SqueezeSeg [230], projects the LiDAR point clouds onto a sphere to derive a dense, grid-based representation for the frontal view. The height of the grid representation corresponds to the number of LiDAR channels, and the final point-wise predictions are obtained by backprojecting the grid representations to the point cloud. Milioto et al. [141] propose to first project the de-skewed point cloud into a range representation to address the shortcomings of the previous approaches, including discretization errors and blurry outputs. Therefore, this approach includes a post-processing step to recover consistent semantic information. Cortinhal et al. [43] adopted the post-processing steps and improved the modules of SalsaNet [4] by residual dilated convolution stacks. The optimized architecture reduces the runtime and enhances segmentation performance. However, these approaches do not include temporal information, which is important when identifying moving objects in LiDAR data. Therefore, Chen et al. [33]

addressed this issue by first projecting the LiDAR point clouds into 2D range images and determining the residual images of previous scans as multi-scan input to perform moving object segmentation on the SemanticKITTI [13] dataset. Kim et al. [95] adopted the approach and improved moving object segmentation in LiDAR data utilizing an attention-based fusion module to combine semantic and motion features. Sun et al. [204] extend the residual image processing into a dual approach to deal separately with appearance features from the range image and motion information from residual images. However, iteratively processing multiple range images results in repeating computations of individual scans throughout the whole network, which is computationally disadvantageous. Furthermore, the range representation results in occlusion artifacts among instances, which limits the performance in 3D scene understanding.

To keep the range information intact, Zhang et al. [263] project the point cloud along the gravitational axis into a Cartesian bird's-eye-view representation. The resulting input image is processed by a U-Net [176] before predicting point-wise semantic classes obtained by nearest neighbor interpolation. Yang et al. [247] employ a similar bird's-eye-view input representation for object detection, including an optimized decoding loss for object localization. Since the grid cells combine the information of multiple points, Lang et al. [110] first extract learned representations before processing the bird's-eye-view image to enhance performance. Baur et al. [12] process two consecutive scans to include temporal information and propose a self-supervised training approach to extract scene flow information and to improve generalization to previously unseen data. Wu et al. [233] extend the temporal information to five scans and add dedicated loss functions to preserve spatial and temporal consistency. For scene flow estimation and moving object segmentation, especially in LiDAR data, temporal information is crucial since most sensors do not provide velocity information. However, the projection into bird's-eye-view results in a sparse representation compared to range view images, where most of the grid cells are not occupied. Therefore, Chen et al. [34] propose a hybrid approach to combine multiple representations to incorporate the complementary information of the different views to enhance performance. Qui et al. [168] adopt the input representation and explore the complementary information from a geometric perspective during the representation learning by proposing a geometric flow network for semantic segmentation. The features of the range view and bird's-eye-view are first aligned using geometric information of the corresponding views and then combined by an advanced fusion scheme to continuously exchange the features within the backbone to enhance accuracy. However, the point distribution of point clouds changes along the radial axis, which is not covered by the Cartesian bird's-eye-view representation, resulting in unevenly distributed points in the occupied cells. Therefore, Zhang et al. [267]

proposed PolarNet, which uses a polar bird's-eye-view representation to enhance performance and account for the changing distribution of the points along the radial axis. The Panoptic-PolarNet [277] approach by Zhou et al. adopts the representation and incorporates an adaptive point cloud pruning technique and instance augmentation to address panoptic segmentation.

However, all projections-based methods face several limitations, including discretization artifacts and backprojection errors, which can harm accuracy. Since the target processing of sparse and noisy radar data includes limited information, the additional loss of information is impractical for radar-based scene understanding. Furthermore, the range image representation comes with several limitations for radars, including the missing elevation information of 3D radars and the limited coverage of sparse point clouds. Consequently, projection-based methods often face limitations for radar-based scene understanding.

4.1.2 Voxel-based Approaches

Voxel-based approaches address the shortcomings of projection-based methods by preventing backprojection errors and keeping the 3D information of the point clouds intact. Zhou and Tuzel [276] propose one of the first approaches to perform object detection in LiDAR point clouds by dividing the point cloud into an equally spaced voxel representation, preserving the height information. To reduce the computational burden, the number of points within the voxels is reduced by random sampling. The resulting input is processed by 3D convolutions to enhance performance over projection-based methods. Tchapmi et al. [208] process point clouds by a 3D fully convolutional network [131] to predict point-wise semantic classes. However, the point cloud is represented as coarse voxels to reduce the computational burden. Therefore, additional processing steps are required to derive point-wise prediction, such as trilinear interpolation. Furthermore, the predictions are processed by a connected conditional random field to enhance consistency and provide fine-grained semantics at object boundaries.

To omit the information loss due to sampling strategies of the input voxels, Meng et al. [138] process the voxel-based input point cloud using a variational auto-encoder with a radial basis function to capture the local geometric distribution. Wang et al. [222] address this issue by voxelizing the point clouds as dense cuboids and leveraging atrous 3D convolutions and attention-based aggregation to enhance feature extraction. This preserves detailed structural information and adaptively selects informative features from different abstraction levels. However, the minimum voxel size is limited due to increased computational cost, and therefore, discretization artifacts constrain the accuracy. Since outdoor point clouds are sparse and vary in density, just a small percentage of voxels are occupied. In comparison to the projection-based method, the proportion of occupied cells is even smaller, making it inefficient to apply fully convolutional neural networks.

To reduce the computational burden, Graham et al. [63] propose submanifold sparse convolutional networks that only generate outputs for occupied voxels. Therefore, the voxel size can be reduced to enhance the accuracy by capturing fine-grained information. Chen et al. [32] adopt the optimized backbone architecture and propose a multi-task learning approach that predicts center shift vectors to support clustering in finding instance boundaries in a post-processing step. Vu et al. propose Softgroup [214], which improves instance predictions by an advanced grouping algorithm to avoid error propagation from hard semantic predictions. The processing of point clouds with sparse convolutions enhances scene understanding. However, for tasks such as moving object segmentation, temporal information is essential, as mentioned in Section 4.1.1. Therefore, Choy et al. [41] introduce the 4-dimensional spatio-temporal Minkowski convolutional neural network to process a sequence of point clouds. Mersch et al. [140] utilized the Minkowski engine and proposed a receding horizon strategy to incorporate new scans in an online fashion and refine predictions by Bayesian filtering to enhance LiDAR moving object segmentation. However, incorporating temporal information by aggregating scans over time and processing the point cloud throughout the network increases the computational cost. Consequently, other approaches exist to enhance the accuracy without relying on aggregated point clouds. Following Polarnet [267] by Zhang et al., Zhu et al. [283] introduce cylindrical partitioning, which does not alter the 3D topology compared to the 2D approach and processes the features by asymmetrical 3D convolutions. Furthermore, to alleviate the interference of lossy label encoding for voxels, the method optimizes the point predictions by a point-wise module. Hong et al. [82] adopt the cylinder convolution and propose a dynamic shifting module to cluster instances of varying size in complex autonomous driving scenes with non-uniform point cloud distributions. The final panoptic predictions are derived by consensus-driven fusion, unifying the semantic and instance segmentation predictions.

However, the refinement module and the consensus-driven fusion are required to compensate for false predictions often resulting from lossy feature and label encoding within the voxels. Even with the application of sparse convolutions, the voxel size is often limited due to increased latency. To mitigate the effect of lossy encoding, other methods [118, 168] directly incorporate multiple representations, such as bird's-eye-view representation with fine-grained voxel features, to reduce runtime and improve accuracy. Therefore, Liong et al. [124] utilize a vanilla late fusion to combine the prediction of the range view and bird's-eye-view LiDAR point cloud representations to leverage the individual strengths and provide more robust predictions. However, the late fusion does not include any information exchange within the network to improve the feature extraction. Therefore, Li et al.

propose Panoptic-PHNet [118], which combines the features of the voxel and bird's-eye-view representation within the instance and semantic segmentation branch to incorporate the complementary information within the feature extraction. The idea is to combine global and fine-grained information to enhance the accuracy. Additionally, this approach models the interaction between foreground objects to improve the offset regression for instance assignments. To enrich the features with fine-grained information, Tang et al. [207] propose a lightweight 3D module that combines the sparse convolutions with a high-resolution point-based branch to enrich the voxel representation with point-based features. Additionally, the network architecture is optimized by a neural architecture search to reduce inference time and preserve high accuracy.

Xu et al. [244] go one step further and combine the voxel-based method with point- and projection-based encodings, utilizing a gated fusion module to adaptively merge the features. Su et al. [203] adopt the backbone and propose a unified network for panoptic segmentation. The optimized method uses a set of point-level classifiers to directly predict instance groupings and the corresponding semantic class in an end-to-end manner, leading to state-of-the-art performance in the panoptic segmentation of LiDAR point clouds. Since voxel-based methods inherently introduce discretization artifacts and information loss, the hybrid methods [48, 128, 192] utilize point-wise information to alleviate the lossy encoding. The information loss is especially harmful to the processing of sparse radar point clouds [255], where the information is rare. Hence, these approaches focus on dense LiDAR point clouds. To overcome the limitations, point-based approaches directly process the unordered point cloud to extract valuable features, which we introduce in the following section.

4.1.3 Point-Based Approaches

Point-based approaches [164, 272] overcome the lossy encoding of aforementioned paradigms by directly processing the point clouds and keeping the spatial information intact. One of the first works proposed by Haykin and Deng [72] utilizes hand-crafted features to train a classifier to distinguish between several major classes of radar returns, including birds, ground, and aircraft. However, the preprocessing and feature selection for the training of the neural network are complex. Therefore, deep learning approaches nowadays directly extract features from the input data to enhance accuracy [155].

The pioneering work of Qi et al., the PointNet [164], utilizes shared multilayer perceptrons (MLPs) to process the input point cloud and aggregate nearby information by max pooling operations. Wang et al. [219] process the input point cloud by multiple PointNet layers and extends the approach by instance fusion for semantic segmentation and semantics awareness for instance segmentation to benefit from the advantages of the individual task and enhance accuracy. PointNet does not capture local structures, which limits the ability to recognize fine-grained patterns and the ability to generalize to complex scenes. The successor proposed by Qi et al., PointNet++ [166], addresses these shortcomings and introduces a hierarchical feature extraction to learn local features with increasing contextual scale. The features are combined from multiple scales based on local point densities to account for the non-uniform distribution of the points. The overall architecture improves the segmentation performance, especially for sparse point clouds.

Hence, several approaches [163, 167, 193, 221] adopted the method, such as Schumann et al. [188], who optimized the architecture to process sparse and noisy radar point clouds. Since radar data is sparser than downsampled indoor point clouds, the parameters of the individual modules are adjusted to capture valuable features for objects comprising only a few radar points. Furthermore, to increase the density of the input, the radar scans are aggregated over 500 ms. However, the ability to capture local structures of sparse radar point clouds is limited. To circumvent this, Schumann et al. [190] include additional features and exploit strong temporal relationships by adding a memory module that iteratively updates the information about the past point clouds. The temporal information further enhances segmentation performance even though the aggregated scans already include points from different time steps. However, for sparse outdoor data, the aggregation of multiple scans is key to including temporal information and extracting valuable features from the surroundings [269]. Nevertheless, processing aggregated point clouds through the whole network comes with an increased computational cost and memory consumption.

Puy et al. [163] propose to use the PointNet++ backbone to process two consecutive point clouds in parallel to estimate the scene flow on point clouds. The correspondence of the points within the two point clouds is determined by graph matching via optimal transport. The scene flow predictions can be leveraged to segment moving objects by selecting points based on a threshold. Liu et al. [126] utilize supervised learning based on simulated data to ensure correct correspondences within the dataset, leading to good generalization capability on real-world data. Wang et al. [221] adopt the approach and propose additional loss functions to enhance the scene flow prediction by considering point-to-plane matching in LiDAR point clouds. Since LiDAR point clouds usually do not provide the Doppler velocity, the moving object segmentation requires at least two point clouds, which increases the computational cost. In contrast, Dubey et al. [51] adopt the approach of Danzer et al. [45] and process single-scan radar point clouds to perform instance segmentation of moving objects. The multi-task learning framework uses uncertainty [92] to combine multiple tasks, including the classification and direction field estimation for each point. However, the approach drastically restricts the region of interest to the near field to simplify the task by omitting sparse regions of the radar data, as explained in Chapter 2.

Using encoder-decoder structures, the point-based methods benefit from effective sampling strategies [85, 234, 248, 249] to enhance feature extraction and reduce computational cost. The most frequently used methods for small-scale point clouds are farthest point sampling [166] and inverse density sampling [234]. However, these methods do not include a learnable weighting to combine the features to learn meaningful representations. Hence, Yang et al. [248] propose a new feed-forward neural module to overcome the limitation and aggregate an arbitrarily sized feature vector based on learnable weights. Hu et al. [85] address the latency issue and utilize random point sampling to achieve remarkably high efficiency for large-scale point cloud segmentation. However, random sampling eliminates points, which is unacceptable for sparse radar data because instances might comprise only a single point.

Besides the direct processing of point clouds by MLPs, another way to exploit stronger connections of the individual points is graph-based methods [119, 109, 200], which conduct message passing on the constructed graphs. The graphs capture the organization of 3D point clouds efficiently, and the resulting representation can be processed by graph convolutions [122]. Wang et al. [220] construct a local neighborhood graph and process the edge features by MLPs, which connect the neighboring pairs of points. Te et al. [209] construct a graph by connecting each point with all other points in the point cloud to extract global relationships. The graph Laplacian matrix that describes the connectivity of features is updated in each layer according to the corresponding learned features, which enables it to adaptively capture the structure of the graph. The interconnection of each individual point increases the computational costs. Therefore, to process large-scale outdoor point clouds, Landrieu and Simonovsky [109] propose super-point graphs that include rich edge features that encode the contextual relationship between object parts. The compressing of the data into super-points is essential to utilize learning-based edge-conditioned convolutions [173] and optimized gated recurrent units [40] that are otherwise infeasible for dense outdoor scenes. The construction of the super-points and the encoding of the features is not trivial. Hence, Shi and Rajkumar [194] replaced the super-point embedding by processing the voxel downsampled point cloud to reduce computational cost. Han et al. propose OccuSeg [68], which directly utilizes a voxel-based approach to process the input point cloud before grouping the input voxels into super-voxels using a graph-based segmentation approach [57]. To enhance the grouping for instance segmentation, the graph incorporates the occupancy information. Razani et al. [171] cluster only foreground classes, leveraging the semantic predictions to construct a graph for instance segmentation. The grouping of the clusters depends on the edge weights predicted by a graph convolutional neural network. Super-voxel or super-point clusters are essential for reducing the computational burden. However, these representations are often inappropriate for sparse point clouds, where single detection can represent complete instances.

Therefore, kernel-based convolutions [37, 210, 245] process point clouds on a per-point basis and extract features for individual points. Wu et al. [234] use an MLP to approximate a weight function and compute translation-invariant and permutation-invariant features on any point cloud. Thomas et al. propose KPConv [210], which directly defines an explicit convolution without an intermediate representation where the kernel weights of KPConv are located in Euclidean space defined by kernel points. The location of the kernel can be learned by the network, enabling deformable convolutions that can adapt to the point cloud geometry. Nobis et al. [153] extend the KPConv [210] approach and exploit the time dimension of multiple radar scans to perform object detection. The resulting features are processed by long short-term memory cells [195] to incorporate the time dependencies on the global features. The encoded features are then upsampled to the original point resolution by three-nearest neighbor upsampling and processed by MLPs to derive the final predictions. Gasperini et al. [59] extend the KPConv backbone with class-agnostic segmentation and learnable clustering to achieve panoptic segmentation. However, to achieve competitive results, the approach adds class-dependent post-processing, including the clustering of split instances of the same semantic class.

Xu et al. [245] aim to address the shortcomings of the KPConv approach and propose a flexible method for kernel design and weight learning. In contrast to KPConv, the weight matrices are derived from a weight bank without requiring the location estimation of the kernel points. Furthermore, the proposed ScoreNet associates relative positions with different weight matrices, which enhances flexibility. To capture strong relationships within local areas, different approaches [156, 213, 241, 249] utilize the self-attention mechanism [212] to leverage the limitations and further improve the accuracy. Fan et al. propose P4Transformer [55], which combines 4D point-based convolutions with video-level self-attention to merge related local areas spatially and temporally. The point-based method benefits from the transformer-based module since the self-attention mechanism is invariant to permutation and is thus inherently suitable for capturing strong local and global dependencies and extracting valuable features in point clouds, as introduced in Section 3.3. Therefore, the point-based approaches are often outperformed by attention-based approaches [271], which extract robust dependency information that is helpful for sparse and noisy radar point clouds, which we elaborate on in the following section.

4.1.4 Transformer-Based Approaches

Transformer-based approaches exploit the self-attention mechanism [212, 241, 249], as introduced in Section 3.3, and achieve remarkable results within various tasks, from natural language processing [26, 212] to computer vision [29, 36, 97, 170, 254] and point cloud understanding [65, 129, 160, 169, 196, 242, 262, 278]. The exceptional results within different point-based perception tasks [134] benefit from the strong feature extraction capability.

Guo et al. propose PCT [65] to enable attention-based point cloud processing. The input point cloud is first processed by a neighbor embedding to map the point cloud into a high-dimensional feature space and incorporate local information. The resulting features are processed by global transformer blocks to learn semantic information. Furthermore, PCT optimized the dot-product attention mechanism inspired by the Laplacian matrix in graph convolutional networks to reduce the influence of noise and to sharpen the attention weights. The positional encoding is removed since the input data already comprises the position information detailed in the coordinates. In contrast to the single-scale processing of PCT, Han et al. [69] utilize cross-level cross-scale cross-attention to capture interactions between and within different scales of the point cloud to enhance the feature representation. Therefore, the input point cloud is split into three subsets with decreasing resolution, which increases the receptive field. The combined output features, thus, include different geometric and semantic information to enhance the accuracy. Hui et al. [87] follow a similar approach and suggest a pyramid point transformer network that adaptively learns the spatial relationship of different sets of neighboring points by grouped self-attention. The grouping mechanism includes multiple keys and queries to derive different attention maps to enhance the discrimination of local features.

To leverage the full potential of transformed-based approaches, Yu et al. [253] propose a pre-training strategy inspired by BERT [49]. The point cloud is first divided into several local point patches, which are encoded into discrete point tokens containing meaningful local information. The pre-training task is to predict the corresponding tokens of randomly masked-out patches. The resulting transformer is then fine-tuned on the downstream task, leading to better generalization compared to transformers without pre-training. The encoding of patches and the resulting global attention induces large computational costs depending on the number of patches, where smaller patches are desired to enhance accuracy.

Zhao et al. [271] propose an encoder-decoder architecture with local attention based on k nearest neighbors (kNN). The optimized architecture utilizes vector attention [270], which enables an individual weighting of the feature channels by subtracting keys and queries. This enables a more fine-grained weighting compared to commonly used dot-product attention. Furthermore, relative positional encoding is added to enhance accuracy and keep position information throughout the network [121]. The approach uses max pooling to downsample the point cloud to increase the receptive field. The features in the decoder are then mapped to the higher-resolution point cloud via trilinear interpolation to derive semantic segmentation results. Wu et al. [236] optimize the approach and propose group vector attention to reduce the number of parameters by dividing the channels into several groups. Moreover, the improved positional encoding and partition-based pooling enhance the segmentation accuracy. The succeeding approach of Wu et al. [235] adopts the sampling strategy and leverages the power of scale by reducing the complex design and accuracy of the algorithm and increasing the number of learnable parameters. Therefore, the neighbor search by kNN is replaced with a point cloud serialization to efficiently increase the receptive field and enhance accuracy. To further optimize the runtime, Park et al. [160] replace the sampling algorithms with centroid-aware voxelization and devoxelization techniques. Furthermore, the network includes a lightweight local self-attention module, which incorporates continuous positional information. The neighboring voxels can be quickly derived via voxel-hashing, reducing the complexity compared to the kNN search.

Zhang et al. [260] adopt the voxel representation and add an additional point branch. The voxels are processed by sparse window attention within non-overlapping 3D voxels. The point branch enriches the voxel features by capturing global context information to enhance the overall performance. To increase the receptive fields of the attention mechanism within the voxels, Lai et al. [242] utilize a stratified key-sampling strategy within the grid representation inspired by the work of Liu et al., the Swin Transformer [127]. For the attention mechanism, nearby points are sampled densely and distant points sparsely to include long-range contexts at a low computational cost. Furthermore, the first layer point embedding by a KPConv layer improves accuracy and facilitates convergence. Contextual relative positional encoding [232] is added to the queries, keys, and values to adaptively capture position information to enhance semantic segmentation performance. To transfer the cubic window approach to large-scale outdoor point clouds, Wang et al. [215] extend the approach of He et al. [73] and partition the sparse voxels into a series of size-equivalent and window-bounded subsets to calculate the attention in parallel. He et al. [74] optimize the runtime of the approach and address the computational overhead in sorting and padding the voxels by directly performing attention on voxel sets with variable lengths, resulting in a better trade-off between accuracy and speed.

Grid representations do not cover the varying sparsity properties of outdoor point clouds. Therefore, Lai et al. propose SphereFormer [108], which utilizes a radial window self-attention that partitions the space into multiple non-overlapping narrow and long windows. The narrow window directly aggregates information from dense close points to the sparse distant ones. The radial windows use an exponential splitting approach to yield fine-grained positional encoding. The dynamic feature selection enhances feature extraction, improving semantic segmentation accuracy. Sun et al. [206] utilize sparse voxel attention, including cross-window correlation with multi-scale feature fusion. Additionally, the method uses a modified CenterNet head [252, 275] to perform object detection. Mask3D [187] by Schult et al. directly predicts instance masks from 3D point clouds to replace the hand-selected voting schemes based on center predictions. The instance queries are iteratively learned from point cloud features at multiple scales. The final mask module predicts a semantic class and an instance heatmap for each query. To enhance the panoptic segmentation, Xiao et al. [239] propose position-aware segmentation to guide the mask prediction and query update processes. This enables the queries to concentrate on specific positions to distinguish small and geometrically similar instances, leading to an improved panoptic segmentation performance. Kolodiazhnyi et al. [99] go one step further and unify semantic, instance, and panoptic segmentation within one network. The method includes a novel super-point query selection to enhance accuracy. Furthermore, the commonly used bipartite matching strategy [187, 205] based on a Hungarian algorithm [104] is replaced by a disentangled matching to reduce training time and make it more stable. Despite the tremendous progress and improved segmentation performance, one serious drawback of mask-based approaches is that the number of masks needs to exceed the maximum number of instances within the scene, which leads to a computational overhead.

Overall, transformer-based approaches enable strong feature extraction from unordered point clouds, enhancing the overall segmentation performance. However, state-of-the-art methods often focus on dense input data and do not account for the specific properties of radar data, neglecting important information. Since radar data is sparse compared to LiDAR point clouds, handling individual points is essential. This highlights the need for dedicated approaches that leverage the potential of radars, incorporate additional information, and propose dedicated modules to improve attention-based processing and scene understanding.

4.2 Exploiting Predictions to Enhance Scene Understanding

The prediction and classification of the environment are essential tasks for scene understanding, as elaborated in the previous section. These approaches provide an accurate prediction of the semantic classes and the instances and include the differentiation of moving and static objects. However, the precise information about how many agents are present and the differentiation of static and moving objects are often insufficient to perform complex driving tasks. Therefore, it is important to process predictions to provide additional information. In the following section, we focus on two common methods, including the tracking of instances and additional refined semantic and instance predictions, which are both useful in enriching the information to enable safe driving behavior. We introduce the respective related work in the following sections.

4.2.1 Tracking Approaches

The tracking of objects within the surroundings of a self-driving vehicle is important to reliably interact with other traffic participants. The primary solution to track instances is the tracking-by-detection paradigm [58], which utilizes instance or object prediction to perform the tracking within a second step. To provide a comprehensive overview, we include methods that address tracking in an end-to-end manner.

Early approaches utilize multiple point clouds to associate objects and perform the tracking. Wang et al. [218] take two consecutive point clouds and predict the point-wise association tracking displacement to capture the movement of the bounding boxes of objects. This method includes a refinement module to improve the association features and enhance the predictions. The final associations of the bounding boxes are determined by the intersection over union. To limit the possible location, several approaches define a template matching to match a template and a respective search area to localize the object and perform the tracking within this predefined region. Qi et al. [165] employ an end-to-end network and operate on sampled seeds instead of 3D bounding boxes to reduce search space by a large margin. Cui et al. propose LTTR [44], which utilizes the self-attention mechanism to capture the attention changes of the object during tracking and improve feature fusion. The cross-attention mechanism determines the similarity among regions from search and template point clouds to capture the relations between the two point clouds. The approach includes voxel-based random sampling to reduce the number of points and capture meaningful features. Zhou et al. [273] replaced the random sampling with relation-aware sampling to preserve the relevant points. Furthermore, the transformer-based feature fusion is extended with a coarse-to-fine matching of the bounding boxes to enhance tracking accuracy.

Another approach to improve feature representation is Siamese networks, which share the weights for the encoding of the point clouds. Giancola et al. [60] propose one of the first Siamese trackers for 3D object tracking in LiDAR point clouds. Therefore, the individual objects are encoded into a high-dimensional latent space. To determine the tracking IDs, the cosine similarity between the respective latent vectors is calculated before assigning the most similar object to the tracks. The decoder part of the approach includes a shape-completion network to ensure a meaningful latent representation and regularize the Siamese tracker. However, the approach struggles to capture complex geometric structures of the potential target because the templates are limited and do not align with the large variations of the potential search areas. To overcome this limitation, Hui et al. [86] propose to directly learn the target completion model from the samples of the search areas to enhance the feature extraction of the potential target and suppress the background information. Additionally, the approach uses the voxel to bird's-eye-view target localization to enhance tracking performance in sparse point clouds. However, for sparse radar data, shape completion is difficult, and intermediate representation induces information loss, limiting the accuracy.

To extend the information, including point-wise predictions, 4D panoptic segmentation unifies instance segmentation and tracking [94], incorporating spatial and temporal information about the environment. Aygün et al. [7] recently introduced LiDAR-based 4D panoptic segmentation by grouping points in the 4D continuum using clustering and assigning a semantic class to each point. The object instances are modeled via Gaussian probability distributions located at the predicted instance centers. The point embedding vectors are evaluated under these Gaussian probability distributions to assign the points to their respective instance. 4D-StOP [101] by Kreuzberg et al. replaces the Gaussian representation and models tracklets as spatio-temporal object proposals. The proposals are first generated by a center-based voting technique before utilizing learned geometric features to aggregate the proposals to form tracklets. The geometric feature association further enhances performance compared to simple geometric matching by incorporating additional information. Zhu et al. [280] extend the approach and propose rotation-equivariance predictions to learn more robust features. Therefore, the backbone utilizes E2PN [279] convolution layers, and the learning targets are formulated as equivariant vector fields, which enhance generalization capability. Hong et al. propose 4D-DS-Net [81], an optimized dynamic shifting clustering approach on predicted center offsets to handle non-uniform point cloud distributions and varying instance sizes. Additionally, this approach optimizes the semantic predictions by consensus-driven fusion, selecting the most frequent semantic label for all points within the instance. Agarwalla et al. [3] extend CenterPoint [252] by Yin et al. and replace the center offset predictions with point-wise velocity offset predictions to associate the objects in concatenated point clouds by greedy nearest-neighbor association. Despite the encouraging results, aggregated scan processing entails a computational burden and induces latency.

Tracking-by-detection algorithms are the most common approaches [35, 47, 159] to work on a sequential scan basis. These algorithms first obtain object detections in the current frame and associate them across time, which can be formulated as bipartite graph matching. The data association is often based on a cost matrix, which can be solved by the Hungarian method [104] or greedy-matching algorithms [252]. The cost matrix is a similarity matrix that compares existing tracks with newly identified objects based on appearance or geometric features. To incorporate motion information, filtering algorithms [16, 38, 223] such as the Kalman filter [91] or particle filter [6] utilize real-world physical models to estimate the state transition of instances. Weng et al. introduce AB3DMOT [223], which provides a compact baseline for multi-object tracking, utilizing a 3D Kalman filter to update the trajectory and determine the association based on the 3D IoU as the cost function. However, IoU-based association is inappropriate for radar signal processing because instances comprise single points where no overlap exists.

Chiu et al. [38] enhance performance by utilizing the Mahalanobis distance to determine the association and exploit the covariance matrices for state estimation. Wu et al. [231] combine appearance features from the backbone network, geometric features of the bounding boxes, and the motion cost by associating motion vectors to perform the association. To enhance the feature representation, CA-Net [136] by Marcuzzi et al. proposes a contrastive approach. The goal of the contrastive loss [93] is to learn that encodings of the same instance, at different time steps, lie close together and far from encodings belonging to other instances. Furthermore, the association module incorporates motion cues based on a center cost to associate instances across scans. Combining geometric and learned appearance features enhances accuracy. CXTrack [246] by Xu et al. and MotionTrack [261] by Zhang et al. utilize attention-based similarity features to track single and multi-objects, respectively. However, MotionTrack struggles to associate objects based on attention due to the sparsity of the point clouds, which is more severe for noisy radar data. Additionally, transformer-based approaches often neglect the valuable geometric features that can enhance tracking performance. To process sparse and noisy radar data, we need to explore the available information to reduce false associations.

4.2.2 Refinement

Prediction refinement includes algorithms to adjust bounding boxes, optimize semantic predictions, and correct instance segmentation. Precise perception algorithms are key to enhance scene understanding. The refinement of the prediction often helps to leverage the full potential of learning-based approaches. We introduce different refinement approaches to illustrate versatile applications. Besides the optimization of the predictions, the refinement also includes improved feature extractions by refining the internal representations. Jiang et al. [90] combine semantic, centroid-aware, and instance-aware features to enhance performance. The refined features are processed to perform point-wise instance assignments in an end-to-end manner. The candidates are masked with a suppression module to handle the problem of redundant instance assignment. Another approach by Zhou et al. is Refine-Net [274], which utilizes multiple refinement modules to combine additional information from multiple feature representations to enhance accuracy. Xiang et al. [237] transfer the feature refinement approach to semantic segmentation and propose an explicit and retrospective refining process that establishes semantic relationships across different decoding stages of the backbone pyramid layers. A cross-attention block summarizes the features to aggregate semantic contexts from nearby points. Furthermore, the introduced gating mechanism controls the information flow to prevent the propagation of erroneous information within the local areas and selectively retain valuable semantic features to enhance performance. Lui et al. [127] propose a similar approach to improve 3D object detection by refining the spatial encodings of an object candidate within each decoder stage. Therefore, the predicted location of the bounding box is updated within the stages to refine the spatial encoding of the same object utilizing the attention mechanism. These refined representations are updated within the consecutive stages to generate accurate object detection results.

The feature refinement comes with a larger computational burden since the whole point cloud is processed to enhance performance. To reduce latency, patch refinement [117] first extracts local areas in the bird's-eye-view representation to reduce the number of points and perform heavy computation within restricted areas. The individual patches are encoded with a higher voxel resolution to enhance the representation and perform the 3D object detection. Hou et al. [84] directly predict the 3D bounding boxes and only refine the prediction inside the boxes to reduce computations and improve instance segmentation. Yi et al. [251] adopt the axis-aligned bounding box predictions to extract region features within these areas. The processing of the regions of interest utilizes PointNet layers to perform classification, regression, and segmentation within different heads. Furthermore, the approach enforces geometric understanding and reduces the blind box proposals that do not correspond to a single object. The refinement of the bounding boxes includes predicting the relative center and size adjustments [172], and the segmentation is derived by predicting a point-wise binary mask for each category label [76]. STD [250] by Yang et al. further enhances the recall for proposal generation by utilizing spherical anchors based on points. Shi et al. [193] process the input point cloud to derive a small number of precise 3D proposals in a bottom-up manner via segmenting the point cloud into foreground and background points. The regions of interest, which consist of the foreground points, are utilized to derive relative residuals in the refinement step. The residuals update the size and location prediction of the input proposals to improve object detection. The refinement utilizes canonical coordinates to learn local spatial features, which are combined with point-wise semantic features of the backbone to achieve accurate box refinement and additional confidence prediction. Liang et al. [123] propose a graph representation to overcome shortcomings from grouping-based instance segmentation methods such as fragmented objects. Therefore, a semantic superposition tree is constructed based on the learned semantic features of the points. The refinement module prunes the super-points that may belong to other instances or the background to improve performance. ScoreNet [90] by Jiang et al. evaluates the generated proposals to derive the final instance predictions. Vu et al. [214] group instances using soft semantic scores to enhance accuracy and avoid error propagation from hard semantic predictions to instance segmentation. The additional top-down refinement module refines and classifies the instance proposals from the corresponding backbone features. Therefore, the false positive predictions introduced by wrong semantic predictions are suppressed, leading to better instance segmentation, especially for ambiguous objects.

Wang et al. [217] utilize the instance information to further enhance the semantic segmentation. The instance-based refinement algorithm refines the network predictions by incorporating additional information, such as the temporal consistency of the outputs. Furthermore, the point-wise predictions within an instance are aligned to derive optimized results. However, the refinement includes multiple hyperparameters which need to be optimized individually. More recently, Kolodiazhnyi et al. [98] utilized a fully sparse convolutional cluster-free network for 3D instance segmentation. The approach follows previous works [251] and selects the features based on predicted 3D bounding boxes to refine the predictions. The resulting regions of interest of the individual instances are processed by a tiny U-Net to solve a binary segmentation task to assign foreground and background labels. Therefore, the instance segmentation is optimized by the mask-based predictions to enhance accuracy. However, the performance is limited due to possible error accumulation from inaccurate box predictions to the refinement phase. Hence, Shin et al. [197] propose a coarse-to-fine approach based on spherical representation to avoid excessive size estimation of axis-aligned bounding boxes. Additionally, the approach handles coarse detection as a soft reference to enable more access to the refinement while excluding unnecessary background points. This allows the correction of both false positive and false negative predictions by utilizing spherical coordinates and predicting a single radial delta for each point to move it along the radial axis to the instance center. The resulting points within the centroid represent the optimized instance predictions.

Nevertheless, the refinement mainly focuses on optimizing the same objective, neglecting the fact that some tasks are inherently easier to perform. Furthermore, the refinement often utilizes voxel representations, which harm accuracy in radar-based scene understanding due to discretization artifacts. Addressing these limitations and integrating radar-specific modules is key to improving performance.

The perception tasks for scene understanding are diverse, and different functionalities need to focus on different aspects. The variety of related work highlights the importance of dedicated algorithms to enable safe autonomous driving. The major focus of research is the processing of LiDAR data, which are denser but face limitations under adverse weather. To address these gaps, we introduce novel approaches to enhance the scene understanding by leveraging the unique properties of radar data in the following chapters.

Part I

Learning-Based Segmentation of Moving Objects in Radar Data

Chapter 5

Semantic Segmentation of Moving Objects

Autonomous vehicles require to understand their surroundings accurately to safely navigate in dynamic, real-world environments across diverse scenarios, including adverse weather conditions. To achieve holistic perception and enhance safety, sensor suites of autonomous vehicles are versatile to explore redundant information of individual sensors such as cameras, LiDARs, or radars, as introduced in Chapter 1. Cameras and LiDAR sensors capture the environment precisely but face limitations under adverse weather such as fog, rain, and snow. Radar sensors work under these conditions and provide superior reliability. As a result, processing radar data is essential to ensure safe autonomous mobility across all scenarios.

The applications of radar sensors are versatile. In this chapter, we investigate the semantic segmentation of moving objects in radar point clouds, which is essential for safe path planning and navigation. Semantic segmentation provides important information for adapting to the motion of other objects in the environment and preventing collisions. Semantic segmentation of moving objects requires differentiating between detections of moving and static objects and assigning a class label to each radar detection, as illustrated in Figure 5.1. Therefore, the task differentiates, for example, between parked and moving cars and assigns the class labels static and car, respectively.

Compared to LiDAR point clouds, radar point clouds are inherently more sparse due to lower resolution and sampling density. They are also noisier, with sensor noise and multi-path propagation further degrading signal quality, as explained in Chapter 2. However, radar sensors provide additional information, such as the Doppler velocity, to directly indicate moving objects, making the sensor inherently suitable for single-scan processing. Furthermore, the radar cross section values support the differentiation and classification of the detections.

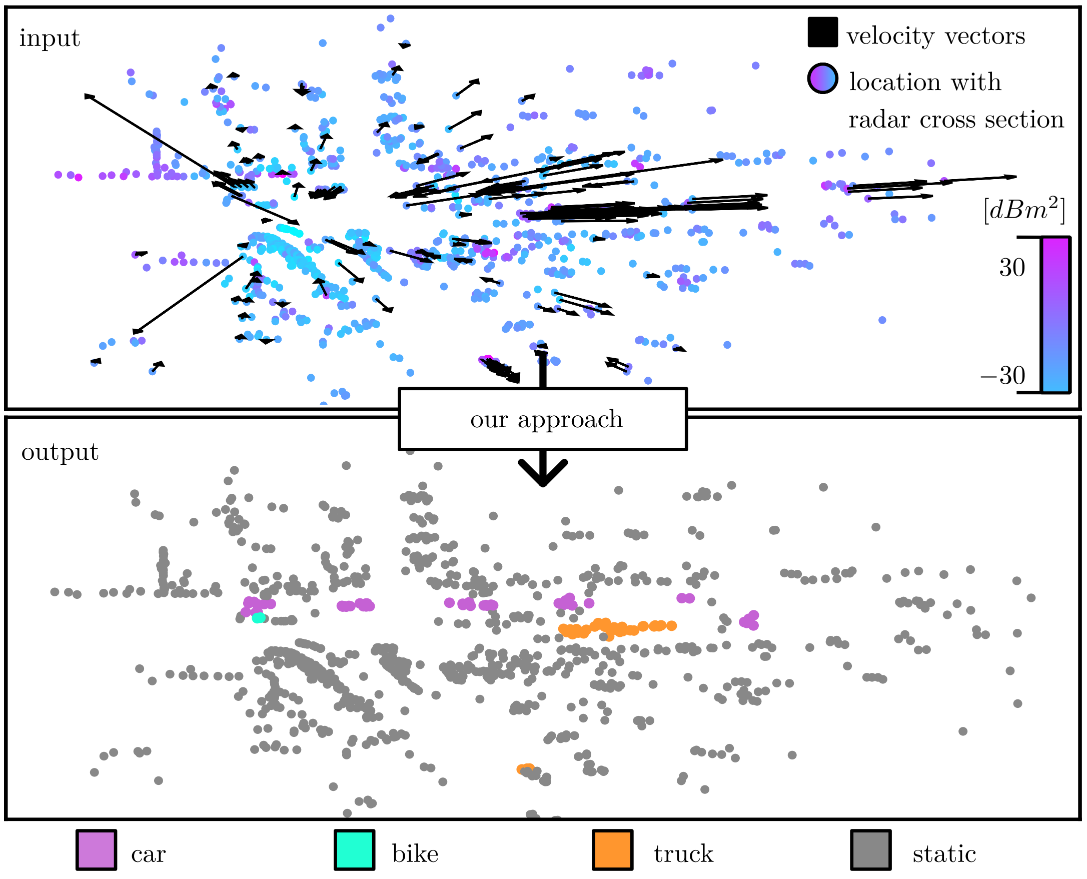


Figure 5.1: Our method performs semantic segmentation of moving objects (bottom) from sparse, single-scan radar point clouds (top), exploiting additional information, including the Doppler velocity and the radar cross section. In the bottom image, each color represents a different semantic class for moving objects (static is grey).

Most state-of-the-art methods for estimating semantics from radar data, such as approaches proposed by Scheiner et al. [184], Schumann et al. [190], and Zhang et al. [269], rely on the aggregation of information over multiple scans to accurately perform semantic segmentation. Additionally, other sensor modalities, such as cameras and LiDARs, strongly depend on aggregating input data to reliably differentiate between moving and static detections. However, aggregation inherently introduces latency, making it often unsuitable for tasks requiring immediate information about the vehicle's vicinity, such as collision avoidance. Li et al. [118] and Xu et al. [244] utilize voxel representations to process LiDAR data to reduce computational complexity. However, voxel-based methods inherently introduce discretization artifacts and information loss, which is especially harmful to the processing of sparse radar point clouds. To overcome these limitations, we investigate the point-wise processing of single radar point clouds in this chapter. Moreover, we exploit additional radar sensor information to enhance performance.

The main contribution of our approach is a new method for accurate, single-scan, radar-only semantic segmentation of moving objects. It takes sparse point cloud representations of radar scans as input and outputs a semantic label for each point. To extract discriminative point-wise features, we build on the self-attention mechanism, a fully attentive neural network with our novel Gaussian transformer layer, and our attentive up- and downsampling modules as central building blocks. We modify the transformer layer and enable the decoupling via the usage of a Gaussian function. Furthermore, our attentive sampling enables the capturing of complex local structures and progressively increases the receptive field of individual points. We combine these building blocks in our new backbone, called the Gaussian Radar Transformer, to enhance feature extraction on sparse and noisy radar point clouds.

In sum, we make three key claims in this chapter: Firstly, our approach demonstrates state-of-the-art performance for semantic segmentation of moving objects in sparse, single-scan radar point clouds without aggregating multiple scans and without exploiting temporal dependencies. Secondly, the Gaussian transformer layer and the attentive upsampling and downsampling modules improve feature extraction by decoupling individual points and enlarging the receptive field to enhance accuracy. Thirdly, our fully attentive network is able to extract discriminable features from additional sensor information, such as Doppler velocity and radar cross section.

5.1 Our Approach to Semantic Segmentation

The goal of our approach is to achieve accurate semantic segmentation of moving objects in single-scan, sparse radar point clouds to enhance scene understanding of autonomous vehicles. To accomplish this, we introduce a point-based framework to directly process the input point cloud, omitting information loss by building upon the successful self-attention mechanism throughout the network. Figure 5.2 depicts our Gaussian Radar Transformer. We adopt the encoder-decoder structure proposed by Zhao et al. in the Point Transformer [271]. We replace each module and use our Gaussian transformer layer, introduced in Section 5.1.1, as the central building block of each stage, which enables decoupled fine-grained feature aggregation. Furthermore, we introduce attentive up- and downsampling modules to enlarge the receptive field and extract discriminative features in Section 5.1.4 and in Section 5.1.3, respectively.

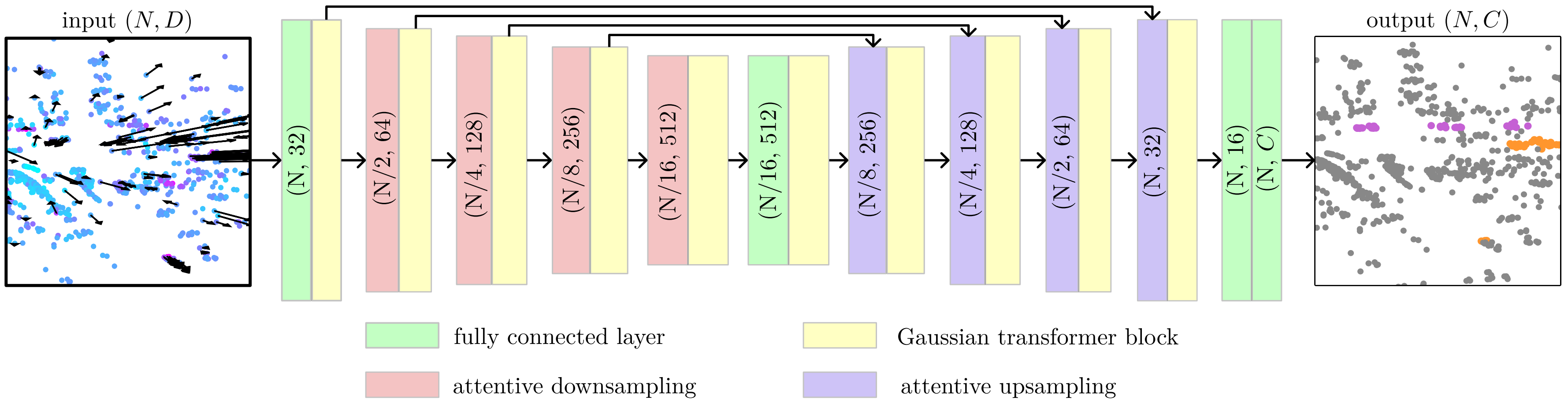


Figure 5.2: Architecture of our Gaussian Radar Transformer for semantic segmentation of moving objects. The fully connected layer first increases the dimension of the per-point features. The Gaussian transformer block incorporates our Gaussian transformer layer in a residual block to extract discriminative features from sparse radar data. Our optimized attentive downsampling and attentive upsampling further improve the feature extraction. The final fully connected layer predicts the semantic class for the individual points in the single-scan radar point cloud. The tuples denote the number of points and feature channels.

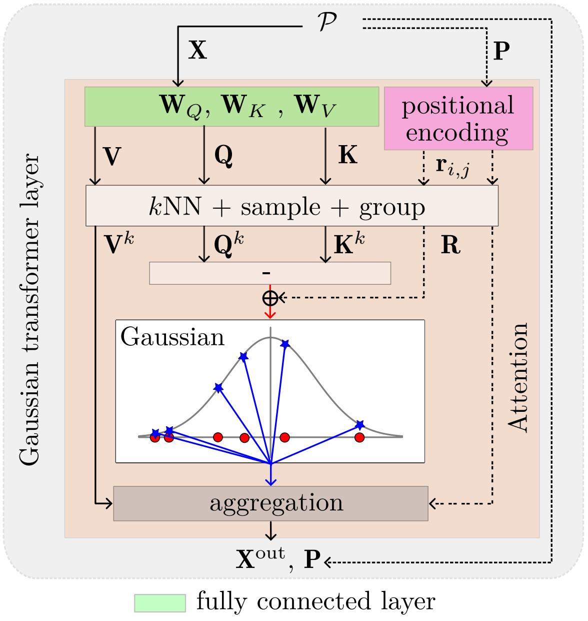


Figure 5.3: The design of the Gaussian transformer layer replaces the softmax function with a Gaussian function to enable an individual weighting of the points. The fully connected layer encodes the features as queries **Q**, keys **K**, and values **V**. The positional encoding **R** incorporates precise geometric information.

5.1.1 Gaussian Transformer Layer

In sparse radar point clouds, individual reflections contain essential information for downstream tasks such as semantic segmentation of moving objects. Small objects such as pedestrians might only comprise a single detection, making the point-based processing crucial. Therefore, the processing of sparse radar data must extract valuable features to enhance performance but also enable single-scan processing.

To tackle this problem and enable independent and precise feature aggregation, we introduce the Gaussian transformer layer based on the Point Transformer layer [271] including vector self-attention as illustrated in Figure 5.3. Contrary to other approaches, including the Point Transformer [271], which focuses on dense point clouds, we do not utilize the softmax function, introduced in Section 3.3, which is defined as:

$$s_i = \frac{\exp(z_i)}{\sum_{j=1}^{N_{\text{gtl}}} \exp(z_j)}. \tag{5.1}$$

The softmax function leads to a coupling of the points since individual outputs s_i, which can be interpreted as probabilities, are dependent on all inputs z_j with $j \in \{1, \ldots, N_{\text{gtl}}\}$. The size of the local neighborhood further influences the weighting, which leads to a dilemma since large local areas increase the receptive field but make the coupling more severe. Furthermore, the coupling of the individual weights is the reason why the softmax function is not scale-invariant, and the additional weighting factor for dot-product attention introduced in Section 3.3 is required.

The backpropagation of the loss $\mathcal{L}$ through the softmax function to obtain the partial derivative $\frac{\partial \mathcal{L}}{\partial z_j}$ to determine the gradients at the input is dependent on all output values. The calculation of the chain rule of derivatives for the softmax can be expressed by the Jacobian matrix $\mathbf{J}_{\text{softmax}}$ as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}} = \mathbf{J}_{\text{softmax}} \frac{\partial \mathcal{L}}{\partial \mathbf{s}}. \tag{5.2}$$

If the output values grow in magnitude, the gradients diminish since the Jacobian converges to a zero matrix. Hence, the error propagation is restricted, which can slow down the learning process. In contrast, we argue that points belonging to the same class should aggregate the information, whereas points belonging to different classes reduce the information aggregation to a minimum, both of which can lead to a close to zero Jacobian matrix. To overcome this limitation, we replace the softmax function with a Gaussian function g, which is executed on every dimension of the vector for vector self-attention.

The input to our Gaussian transformer layer contains the information about the individual points within the single current scan $\mathcal{P}$ at time t. This comprises the point coordinates $\mathbf{P} = [\mathbf{p}_1, \ldots, \mathbf{p}_N]^\top \in \mathbb{R}^{N\times 2}$, where $\mathbf{p}_i \in \mathbb{R}^2$ and the point-wise features $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]^\top \in \mathbb{R}^{N\times D}$, where $\mathbf{x}_i \in \mathbb{R}^D$ for N points in the scan. The input features are first encoded as queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$, as follows:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \qquad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \qquad \mathbf{V} = \mathbf{X}\mathbf{W}_V, \tag{5.3}$$

where $\mathbf{W}_Q \in \mathbb{R}^{D\times D}$, $\mathbf{W}_K \in \mathbb{R}^{D\times D}$ and $\mathbf{W}_V \in \mathbb{R}^{D\times D}$ are the weights of the fully connected layers. We follow the point transformer approach and arrange the query, keys, and values as a matrix, which are sampled according to the local neighborhoods to reduce computational cost. We utilize the k nearest neighbor (kNN) [166] algorithm with $k = N_{\text{gtl}}$ points to derive for each point the respective information, resulting in $\mathbf{Q}^k$, $\mathbf{K}^k$, and $\mathbf{V}^k \in \mathbb{R}^{N\times N_{\text{gtl}}\times D}$. To incorporate fine-grained position information within the attention mechanism, we process the relative positions $\mathbf{r}_{i,j}$ by two fully connected layers with weight matrices $\mathbf{W}_1^p \in \mathbb{R}^{2\times 2}$ and $\mathbf{W}_2^p \in \mathbb{R}^{2\times D}$ and the Gaussian error linear unit [77] activation function to determine the relative positional encoding $\mathbf{R} \in \mathbb{R}^{N\times N_{\text{gtl}}\times D}$ [271]. We adopt subtraction as a relation function to calculate the attention weights and replace the softmax function with the Gaussian g function:

$$\mathbf{A}_{i,j} = g((\mathbf{Q}^k_{i,j} - \mathbf{K}^k_{i,j}) + \mathbf{R}_{i,j}), \tag{5.4}$$

to assess fine-grained information flow for sparse radar point clouds. We introduce parameters of the Gaussian function, such as the mean and the variance in Section 5.2. Since the Gaussian function is applied to each dimension of the matrix individually, the points are decoupled, which enables precise information aggregation to enhance feature extraction and performance. As a result, the information of points belonging to the same class is ideally aggregated, and the information of points belonging to different classes is not mixed to improve feature extraction. For points belonging to the same class, the attention weight to weigh the features should be high, and for points belonging to different classes, the weights should be low.

Moreover, the partial derivative of the Gaussian function depends on a single output value s_j. Hence, vanishing gradients may influence individual points but not whole local areas, which can be seen by the chain rule of derivatives:

$$\frac{\partial \mathcal{L}}{\partial z_j} = \frac{\partial \mathcal{L}}{\partial s_j}\frac{\partial s_j}{\partial z_j}. \tag{5.5}$$

To derive the output $\mathbf{X}^{\text{out}}$, we calculate the sum of the element-wise multiplication:

$$\mathbf{X}_i^{\text{out}} = \sum_{j=1}^{N_{\text{gtl}}} \mathbf{A}_{i,j} \odot \mathbf{V}_{i,j}^k, \tag{5.6}$$

without further processing by a fully connected layer, reducing computational cost in contrast to other approaches [271]. We pass the output features and the point coordinates to the consecutive module. We do not modify the point coordinates to keep fine-grained position information in deeper layers.

5.1.2 Gaussian Transformer Block

Our Gaussian transformer layer is embedded into the center of the Gaussian transformer block, which is a residual block, similar to the Point Transformer block [271], with two fully connected layers processing the input and the output features. We replace the activation function with the Gaussian error linear unit after each fully connected layer and utilize batch normalization [88]. The Gaussian transformer block processes the single-scan point clouds, including the point coordinates and point-wise features. The features of the individual points are enriched by the information aggregation within the block enhanced by the Gaussian transformer layer. We add the processed output features to the features of the skip connection to reduce vanishing gradients. The point coordinates are utilized to calculate the positional encoding but are not further transformed to keep detailed position information within the whole network.

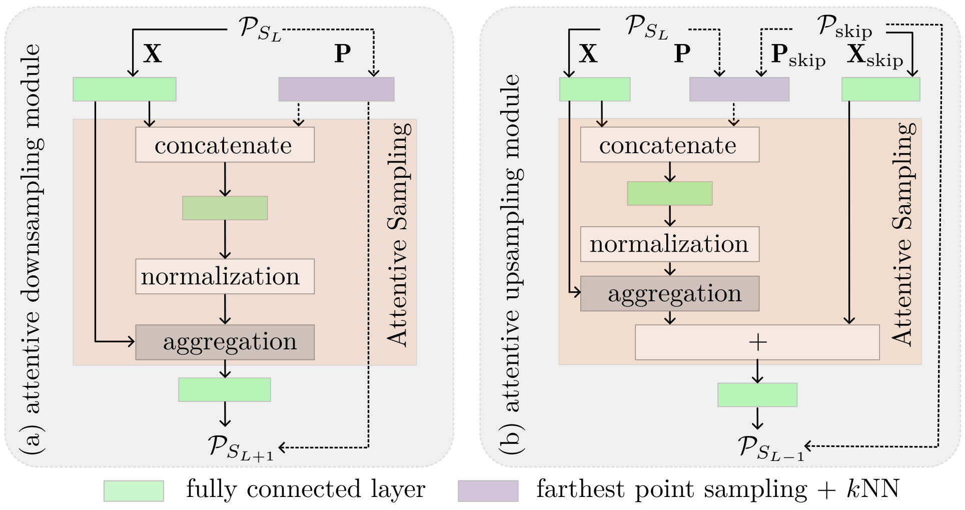


Figure 5.4: Detailed design of the attentive downsampling (a) and attentive upsampling (b) module of our Gaussian Radar Transformer. The downsampling samples a subset of the input point cloud $\mathcal{P}_{S_L}$ of the respective stage S_L. The attentive upsampling combines the information of the point cloud of the skip connection $\mathcal{P}_{\text{skip}}$ and the point cloud $\mathcal{P}_{S_L}$, which has to be upsampled.

5.1.3 Attentive Downsampling Layer

Our Gaussian Radar Transformer follows an encoder-decoder structure to derive meaningful fine-to-coarse feature representations, as elaborated in Section 3.3.1. The goal is to increase the receptive field within the encoder and combine the information of different levels within the decoder of the network. To reduce the cardinality of the point cloud $\mathcal{P}_{S_{L+1}} \subset \mathcal{P}_{S_L}$ and thereby the number of points N_{S_L}, we process the point cloud by the attentive downsampling layer in each stage S_L, depicted in Figure 5.4 (a). Our approach aims to enable adequate sampling and feature processing by applying the self-attention mechanism throughout the network. In contrast to commonly used max pooling [166] the idea is to keep the information within the point cloud intact. The max pooling leads to information loss, which can harm accuracy, especially for sparse radar point clouds. To reduce computational complexity, we follow Yang et al. [248] and calculate the attention weights by a single fully connected layer with the weight matrix $\mathbf{W}^f_{S_L} \in \mathbb{R}^{(D_{S_L}+2)\times D_{S_L}}$ and no direct representation of keys, queries, and values. Furthermore, the calculation does not include a softmax and Gaussian function to reduce computations. We concatenate the input features $\mathbf{X}$ and the point coordinates $\mathbf{P}$ to include positional information to calculate the attention weights $\mathbf{A}_{i,j}$. We normalize the attention weights over the whole point cloud within the local neighborhood to amplify the contribution of valuable points. We use farthest point sampling and the kNN algorithm to sample the point for the downsampling operation. The final weights are multiplied with the input features within local areas, with $k = N_d$ points resulting in:

$$\mathbf{X}_i^{\text{down}} = \sum_{j=1}^{N_d} \mathbf{A}_{i,j} \odot \mathbf{X}_{i,j}. \tag{5.7}$$

The features are fed into another fully connected layer with layer normalization [243] and a Gaussian error linear unit activation function. In contrast to Point Transformer [271], which utilizes farthest point sampling and max pooling [166], our attentive downsampling includes the information of nearby points, which we assume as valuable for sparse point clouds.

5.1.4 Attentive Upsampling Layer

To deduce discriminative features, we argue that the upsampling and feature concatenation of the skip connection are crucial to further enhance performance. The common method for upsampling, also utilized by Point Transformer [271], is an interpolation of the $k = 3$ nearest neighbors based on an inverse distance weighted average [166]. The interpolated points are concatenated with the features of the points, which are passed through the skip connection. The inverse distance weighted average does not include further feature-based information. Hence, the interpolation combines the features only based on their relative position. This is reasonable for dense point clouds because nearby points often belong to the same class.

However, this might be problematic for sparse point clouds, especially for small instances, which are represented by single points. Therefore, we consider upsampling as an important part to improve feature extraction and propose the attentive upsampling layer. The upsampling layer, which is illustrated in Figure 5.4 (b), first processes the features of the skip connection $\mathbf{X}_{\text{skip}}$ and the proceeding Gaussian transformer block $\mathbf{X}$ by two separate fully connected layers with layer normalization and Gaussian error linear unit activation function. To propagate the points from $\mathcal{P}_{S_L}$ to $\mathcal{P}_{S_{L-1}}$ where $\mathcal{P}_{S_L} \subset \mathcal{P}_{S_{L-1}}$ with $N_{S_L} \leq N_{S_{L-1}}$, we feed the position information of the two point sets and the corresponding features into our attentive upsampling layer. We calculate the k nearest neighbors of the individual points for the point set of the skip connection $\mathcal{P}_{\text{skip}}$ within the point cloud, which has to be upsampled $\mathcal{P}_{S_L}$. The attention mechanism enables information aggregation of larger local areas since the attention weights will control the information flow and not reduce the discriminability, which is possible if large local regions are interpolated. To integrate the positional information, we calculate the relative position of the kNN of the two point sets given by:

$$\mathbf{r}_{i,j} = \mathbf{p}_i - \mathbf{p}_j, \tag{5.8}$$

where $\mathbf{p}_j \in \mathcal{P}_{\text{skip}}$ and $\mathbf{p}_i \in \mathcal{P}_{S_L}$. The relative distances are further sampled according to the nearest neighbors, resulting in the relative distances $\mathbf{R}_{i,j}$, which are concatenated with the features. Following our downsampling layer, we calculate the attention weights directly by processing the concatenated features with a fully connected layer and normalizing the weights within the local neighborhood defined by N_u points over the whole point cloud. We combine the features by summation. The output is processed by a fully connected layer with layer normalization and Gaussian error linear unit activation function to derive the final output of the attentive upsampling layer.

The self-attention mechanism turns into an inter-attention between the two point clouds to enable attentive feature aggregation. The upsampling is repeated until we have broadcasted the features to the original set of points. We optimize the information aggregation by determining the weighting based on the relative position and the features. We emphasize that the sampling steps are essential for appropriate feature extraction of transformer architectures for sparse point clouds.

5.2 Implementation Details

The input is a sparse radar point cloud with N points, feature dimension D, and batch size b. Each point $\mathbf{p}_i$ is defined by two spatial coordinates x_i^C and y_i^C. Additionally, the radar sensors provide the ego-motion compensated Doppler velocity v_i and the radar cross section σ_i resulting in a 4-dimensional input vector $\mathbf{x}_i = \left[x_i^C, y_i^C, \sigma_i, v_i\right]^\top$. To simplify the batch processing for the varying sizes of the radar data, we utilize zero padding to derive a fixed number of input points of $N = 1024$.

We construct our architecture based on the self-attention mechanism. The central building blocks are the Gaussian transformer layer and the attentive down- and upsampling modules to extract discriminative features for point cloud understanding. The backbone adopts the U-Net architecture of Point Transformer [271] with an encoder-decoder architecture, including skip connections. First, we directly extract features of the sparse input point cloud by a Gaussian transformer block and increase the per-point feature dimension to 32. The resulting features are progressively down-sampled by four consecutive stages where each reduces the cardinality of the point cloud by a factor of two, resulting in $[N/2, N/4, N/8, N/16]$ points. The per-point features are further gradually increased to 64, 128, 256, and 512. The individual stages include the Gaussian transformer block and attentive downsampling modules in the encoder part, which are replaced by attentive upsampling modules in the decoder part of the network. The per-point features maps of the final decoder layer are processed by an MLP with two fully connected layers to obtain point-wise semantic classes $\mathcal{P}^{\text{sem}} = \{p_1^{\text{sem}}, \ldots, p_N^{\text{sem}}\}$, where $p_i^{\text{sem}} \in \{1, \ldots, C\}$.

We implement the Gaussian Radar Transformer in PyTorch [161]. To train the network, we utilize the stochastic gradient descent optimizer with an initial learning rate of 0.05, a momentum of 0.9, and a cosine annealing learning rate scheduler [132]. The batch size b is set to 32. The loss combines the Lovász loss [21] and weighted cross-entropy. We follow Schumann et al. [190] and set the weights of the cross-entropy loss for dynamic objects to 8.0 and for static to 0.5 to account for the class imbalance of the dataset. For the attentive sampling operations, we define $k = 9$ for the kNN operation, and for the Gaussian transformer layer, we restrict the local area to $N_{\text{gtl}} = 16$. We define $g(x)$ as:

$$g(x) = \exp\left(\frac{-x^2}{2}\right), \tag{5.9}$$

such that for $x = 0$ the attention weight is $g(x) = 1$. The idea to scale the Gaussian function is that weights smaller than one result in an attenuation effect, which we argue is not optimal for feature extraction of sparse radar point clouds. Therefore, the replacement of the softmax function includes the advantage that the sum of all attention weights within the local area is not bound to one. We apply data augmentation, including scaling, rotation around the origin, jitter augmentation of the coordinate features, and instance augmentation, to improve generalization.

5.3 Experimental Evaluation

The main focus of this chapter is to enhance the semantic segmentation of moving objects in sparse and noisy radar point clouds. We present our experiments to show the capabilities of our method and to support our key claims that our approach achieves state-of-the-art performance in semantic segmentation of moving objects in single-scan radar point clouds without exploration of temporal dependencies or the aggregation of scans. Moreover, we demonstrate that the Gaussian transformer layer and the attentive up- and downsampling modules improve feature extraction and contribute to the final performance. Our fully attentive network is able to extract valuable features from the Doppler velocity and radar cross section provided by the radar sensor.

5.3.1 Experimental Setup

We train and evaluate our method on RadarScenes [189], which is the only large-scale, open-source radar dataset including point-wise annotations for varying scenarios. The dataset consists of 158 annotated sequences. We use the recommended 130 sequences for training and split the remaining 28 sequences into validation (sequences: 6, 42, 58, 85, 99, 122) and test set. The RadarScenes dataset provides individual point clouds for the four radar sensors. The measurements are from the near-range mode of the 77 GHz automotive radar sensors, which cover detections in a range of up to 100 m. Two sensors are mounted at $\pm 85\,^\circ$ and two sensors at $\pm 25\,^\circ$ with respect to the driving direction. Since the field-of-view of the sensors is restricted to certain areas, we derive detailed information about the surroundings by merging the individual sensor data from the four sensors into a single radar point cloud. The measurement times and ground truth pose information of a differential global positioning system are given, which enables a transformation into a common coordinate system. We always aggregate four scans consecutive scans and compensate for the relative movement, resulting in the final input point clouds with transformed local coordinates. To evaluate the performance, Schumann et al. [189] propose the point-wise macro-averaged F_1 scores based on all five moving object classes and the static background class resulting in $C = 6$ classes. We further report the intersection over union (IoU) and mean intersection over union (mIoU) defined as:

$$\mathrm{IoU} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN} + \mathrm{FP}}, \tag{5.10}$$

$$\mathrm{mIoU} = \frac{1}{C}\sum_{i=1}^{C} \mathrm{IoU}_i, \tag{5.11}$$

where the true positives (TP), false positives (FP), and false negative (TP) predictions are summarized in these metrics, which are common for semantic segmentation tasks [13]. We train each network using its specific hyperparameters with two Nvidia RTX A6000 GPUs over 50 epochs on the training set and report the results on the test set. We conduct all ablation studies on the validation set. Since the Point Transformer does not provide semantic segmentation results for the RadarScenes dataset, we transfer the approach to the radar domain. For more details on the training regime, we refer to the original paper [271].

5.3.2 Semantic Segmentation of Moving Objects

The first experiment presents the performance of our approach on the RadarScenes test set to investigate the claim that we achieve state-of-the-art results for semantic segmentation of moving objects in sparse and noisy radar point clouds without the aggregation of scans or the exploration of temporal dependencies. In this experiment, we compare our Gaussian Radar Transformer with the recent and high-performing Point Transformer by Zhao et al. [271] as well as the baselines provided by Schumann et al. [188, 190]. We selected Point Transformer as a reference since the method meets the following requirements: (1) single-scan input

Method	Input	mIoU	F1	IoU						F_1					
				static	car	pedestrian	pedestrian group	bike	truck	static	car	pedestrian	pedestrian group	bike	truck
RadarPNv1 [188]	aggregation	61.0	74.3	98.7	58.2	36.0	58.7	58.4	56.1	99.4	73.6	52.9	74.0	73.8	71.9
RadarPNv2 [190]		61.9	75.0	98.7	63.8	**38.8**	58.5	51.0	61.0	99.4	77.9	**55.9**	73.8	67.5	75.8
Point Voxel Transformer [260]	single-scan	45.9	57.5	99.3	47.5	7.3	47.5	54.6	19.2	99.6	64.4	13.6	64.4	70.6	32.2
Point Transformer [271]		55.6	68.1	99.3	58.1	15.2	56.8	55.1	48.9	99.6	73.5	26.4	72.5	71.1	65.6
Gaussian Radar Transformer		**68.5**	**79.8**	**99.4**	**69.6**	36.3	**71.2**	**71.2**	**62.8**	**99.7**	**82.1**	53.2	**83.2**	**83.2**	**77.1**

Table 5.1: Semantic segmentation results of moving objects on the RadarScenes test set in terms of IoU and F_1 scores. The results of RadarPNv1 [188] and RadarPNv2 [190] are calculated based on the reported confusion matrix. Our method outperforms state-of-the-art segmentation approaches and enhances IoU compared to aggregation-based methods for most of the classes. The best results are in bold.

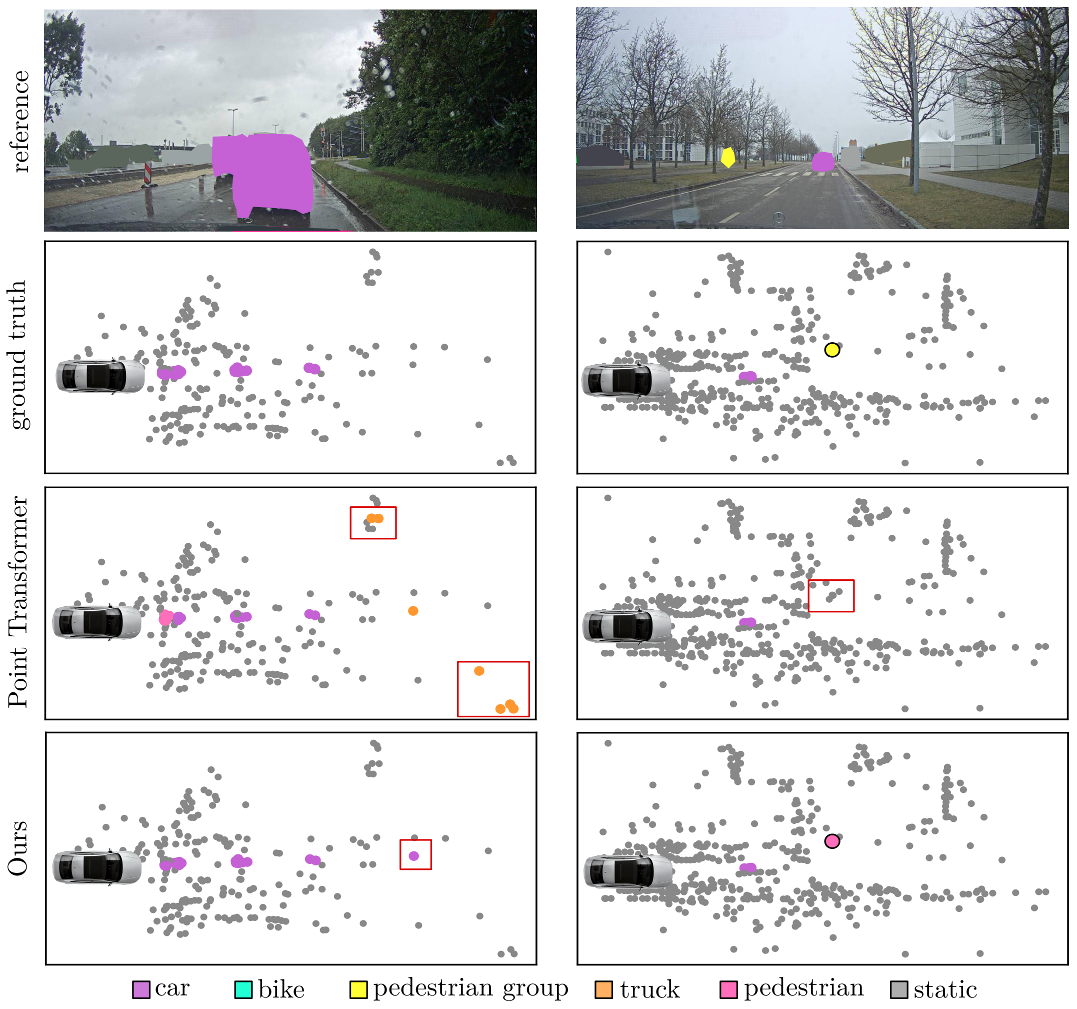


Figure 5.5: Qualitative results of the Point Transformer [271] and our Gaussian Radar Transformer on the test set of RadarScenes [189]. The left column is from sequence 93 (rain), and the right is from sequence 14 (fog). In the images of the predictions, the color represents the predicted semantic class of moving objects (static is grey). The colors in the images correspond with the ground truth if the object is visible. The camera images are anonymized and shown for reference.

for comparability; (2) point-based method, since the voxelization leads to discretization artifacts and hence a loss of information, see Point Voxel Transformer in Table 5.1; (3) very good performance on different benchmarks including semantic scene understanding. Furthermore, the Point Transformer [271] utilizes vector attention, which is beneficial for point cloud understanding because the weighting of individual channels enables a fine-grained weighting of the corresponding features.

Our Gaussian Radar Transformer outperforms the existing methods in terms of mIoU and F_1 score, as displayed in Table 5.1. We achieved superior performance in five of the six classes, except for pedestrians. We assume that individual detection in radar scans contains important information, which is why point-

#	Attentive downsampling	Attentive upsampling	Gaussian transformer layer	F_1	mIoU
A				74.0	61.0
B	✓			77.0	64.7
C		✓		77.3	65.5
D	✓	✓		78.8	66.8
E	✓	✓	✓	**79.4**	**68.3**

Table 5.2: Influence of the different components of the approach in terms of mIoU and F_1 score on the RadarScenes validation set. The Gaussian transformer layer, the attentive downsampling, and the attentive upsampling contribute to the improved performance. The best results are in bold.

based methods enhance performance compared to Point Voxel Transformer. The combination of the dual branch processing of the voxelized point cloud and the point-based transformer branch hence sufferers from information loss due to discretization artifacts. The baselines exploit temporal dependencies of consecutive radar scans within a memory feature map, utilize additional global coordinates, or densify the point clouds by aggregation. The baselines utilize an aggregation of multiple point clouds to incorporate the detection over 500 ms. Since the radar sensors operate at around 17 Hz, the aggregation includes up to 10 scans of the four individual radar sensors. However, the exact comparison of the results is difficult because Schumann et al. work on a subset of the officially released dataset. Nevertheless, the IoU for the class pedestrian indicates that the exploration of temporal information is beneficial for small instances. We suspect that the consistent detection of pedestrians over the whole sequence, which is difficult for strict single-scan approaches, further improves the performance. Despite that, the Gaussian Radar Transformer considerably improves the IoU for the class pedestrian as opposed to Point Transformer by more than 19 absolute percentage points. Figure 5.5 shows some qualitative results on the test set. Notably, our approach achieves superior performance under adverse weather, including rain and fog. Furthermore, we observe that the radar signal processing of single scan inputs is able to detect distant pedestrians comprising only one detection.

5.3.3 Ablation Studies on Method Components

The first ablation study presented in this section is designed to support our second claim that our proposed self-attention modules each contribute to the advancements in the segmentation performance of the Gaussian Radar Transformer. To assess the influence of the different components of our fully attentive backbone, we evaluate the performance in terms of mIoU and F_1 score on the validation set. To replace our proposed modules, we follow commonly used network designs. We substitute the Gaussian function with the softmax function and keep the rest of the Gaussian transformer layer as it is. For the attentive downsampling, we utilize local max pooling, and we exchange attentive upsampling using trilinear interpolation based on an inverse distance weighted average. Table 5.2 summarizes the influence of different components on the performance in terms of mIoU on the validation set.

In configuration (A), we replace each module with its substitute, which leads to a noticeable decrease in mIoU. We suspect that the commonly used modules are highly optimized for denser point clouds but struggle to capture fine-grained information from sparse and noisy radar point clouds. In (B), we add attentive downsampling, see Section 5.1.3, which introduces a smooth information exchange within the downsampling step of individual points, visibly improving the results. Furthermore, the information loss within deeper layers due to max pooling is disadvantageous for sparse point clouds. In (C), we add the attentive upsampling module to enlarge the receptive field and include encoded feature information to optimize the information aggregation, see Section 5.1.4. The larger receptive field resulting from the increased local area from three (trilinear) to nine points improves the F_1 score by 3.3 and the mIoU by 4.5 absolute percentage points. Although the attentive upsampling only affects the features of the decoder part, it leads to an additional improvement of mIoU by 0.8 absolute percentage points compared to attentive upsampling in (B). We assume that the interpolation of the features results in an exchange of features of different classes, which makes it difficult to reliably perform the semantic segmentation. In (D), we add the attentive up- and downsampling, which further enhance the performance. This shows the importance of attentive sampling modules for sparse radar point cloud processing. In (E), we utilize the fully attentive network to illustrate the improvement due to the usage of the Gaussian function by decoupling individual points, see Section 5.1.1, resulting in the best performance. In conclusion, the Gaussian function and the attentive up- and downsampling are essential to extract valuable features from sparse and noisy radar point clouds.

5.3.4 Ablation Studies on Input Features

The third experiment evaluates the performance depending on the provided information by the radar sensor and demonstrates that our approach is capable of capturing complex local structures within the features to enhance mIoU. For this experiment, we utilize our Gaussian Radar Transformer and add to the position information of x^C and y^C coordinates, the ego-motion compensated Doppler velocity v, the radar cross section σ, or both. Table 5.3 displays the influence

Input Features	F$_1$	mIoU
$\mathbf{x} = \left[x^C, y^C\right]$	56.0	43.7
$\mathbf{x} = \left[x^C, y^C, \sigma\right]$	63.7	50.1
$\mathbf{x} = \left[x^C, y^C, v\right]$	75.0	62.0
$\mathbf{x} = \left[x^C, y^C, \sigma, v\right]$	**79.4**	**68.3**

Table 5.3: Influence of the different input features in terms of mIoU and F$_1$ score on the RadarScenes validation set. The radar features, including the spatial location, radar cross section σ, and Doppler velocity v, are essential to enhance scene understanding. The best results are in bold.

of the input features $\mathbf{x}$ on the validation set performance. As we presume, the ego-motion compensated Doppler velocity is especially valuable for semantic segmentation of moving objects since the feature inherently distinguishes between moving and non-moving parts of the environment, resulting in an increase of mIoU of 18.2 absolute percentage points. Moreover, we further improve the mIoU if we add the radar cross section features σ, because it depends on the material, the surface, and the shape of the object, which is valuable to differentiate between different classes. However, it is remarkable that the network still identifies moving objects and correctly assigns a semantic class based on the spatial coordinates. We assume that most of the objects appear in similar places, and the attention mechanism is capable of detecting local clusters, such as cars driving in front of the ego-vehicle. This illustrates the strong generalization capability of our Gaussian Radar Transformer. Nevertheless, the best performance is achieved by including radar cross section, and ego-motion compensated Doppler velocity, suggesting that our approach extracts valuable features for the downstream task from additional sensor information.

5.3.5 Runtime

The primary focus of our approach is to achieve state-of-the-art performance for single-scan radar signal processing. However, the runtime and the memory consumption of the algorithm play an important role in real-world applications. We investigate the input representation of the radar point cloud to derive a detailed evaluation of the runtime and the performance in terms of mIoU. The processing of the point cloud in batches requires that the input vectors have the same size to concatenate the input data along the batch dimension. However, radar point clouds largely differ in the number of points N depending on the surroundings of the vehicle. Our approach utilizes zero padding to align the number of input points, but that increases the computational burden because artificial points are included in the processing steps. Therefore, we adopt the advanced batching pro-

RadarScenes [189]	training set	validation set	test set
number of scans	175, 918	7, 062	27, 849
number of moving objects	630, 851	21, 143	95, 392
average number of points per scan	569	559	534

Table 5.4: The number of scans, the number of moving instances, and the average number of points per scan in the RadarScenes training, validation, and test set.

posed by Zhao et al. [271] to always keep the original number of points as input data. The algorithm combines the individual scans within one large vector. The transition from one scan to the other scan is defined by a scalar, which corresponds to the number of points within each scan. As a result, we can use this information and split the concatenated scans into individual scans to calculate, for example, the nearest neighbors in parallel.

Furthermore, we investigate the proposed dataset split for the single scan processing. The RadarScenes dataset includes four radar sensors as explained in Section 5.3.1, and the data is ordered according to the recording time of the individual sensor. The sensors are equal and operate at the same frequency. However, due to internal processes or latencies in the processing of the packages, the resulting data stream is asynchronous at some points in time. Therefore, not for all time steps is the aggregation of four consecutive scans useful because this might include two point clouds of the same radar sensor. This is also important for the runtime because the latency between different sensors is ideally smaller than the time between two consecutive scans of the same sensor. Therefore, we aggregate, in this case, only the point clouds of different radar sensors, resulting in less than four point clouds leading to sparser input data. Since the resulting point clouds are recorded within a short time frame of around 50 ms, we do not compensate for additional movement. We summarize the number of scans and the number of moving objects for the individual dataset splits in Table 5.4. Additionally, 87% of the points with an absolute ego-motion compensated Doppler velocity larger than 0.1 m/s are noise and hence belong to the static class. The sparsity and noisiness demonstrate that reliable scene understanding in radar data is more challenging compared to typically used 3D LiDAR data.

We evaluate the performance regarding mIoU and runtime and report the results in Table 5.5. The sparse representation and the advanced batch processing both harm the accuracy of the semantic segmentation. We argue that the aggregation of four scans and the zero padding help to improve accuracy. The advantage of the padded point clouds is that during the downsampling, these points are also removed, and the resulting point clouds are more dense, resulting in a downsampling factor smaller than two within the individual stage. However, the

Model	parameters (M)	mean runtime (ms)	mIoU
Gaussian Radar Transformer	8.4	123.0	68.5
Gaussian Radar Transformer (optimized)	8.4	24.0	56.9

Table 5.5: Evaluation of the mean runtime, number of parameters, and performance in terms of mIoU of the Gaussian Radar Transformer on an Nvidia RTX A6000 GPU based on 1000 randomly sampled point clouds of the RadarScenes dataset. We optimize the Gaussian Radar Transformer in terms of the runtime to enable real-time processing.

runtime-optimized model still outperforms state-of-the-art approaches, which use dense representation and zero padding. More importantly, the runtime is reduced by utilizing advanced batching and the sparse input representation. The resulting approach is applicable for online processing, which is important for tasks requiring immediate feedback. As a result, we consider runtime to be an important factor and utilize the sparse representations within the following chapters.

In summary, our evaluation supports our claim that our method provides competitive semantic segmentation performance of moving objects in single-scan, sparse radar point clouds. At the same time, our method exploits self-attention modules, which enhance performance in multi-dimensional radar data processing, outperforming state-of-the-art approaches. Thus, we support all our claims with this experimental evaluation.

5.4 Conclusion

Scene understanding is crucial for autonomous vehicles in dynamic environments to make future state predictions, avoid collisions, and plan paths. Camera and LiDAR perception algorithms have made tremendous progress in recent years but face limitations under adverse weather conditions. To leverage the full potential of multi-modal sensor suites, radar sensors are essential for safety-critical tasks and are already installed in most new vehicles today. In this chapter, we presented a novel approach to perform semantic segmentation of moving objects in sparse, noisy, single-scan radar point clouds obtained from automotive radar sensors to enhance the perception of the environment. Instead of aggregating multiple scans to densify the point clouds, we propose a novel approach based on the self-attention mechanism to accurately perform single-scan segmentation. We propose the Gaussian transformer layer, which replaces the softmax normalization with a scaled Gaussian function to decouple the contribution of individual points. To tackle the challenge of the transformer to capture long-range dependencies, we incorporate attentive up- and downsampling modules. This approach enlarges the receptive field and captures strong spatial relations. The resulting approach exploits the self-attention mechanism throughout the whole network. This allows us to successfully segment moving objects and improve feature extraction. We implemented and evaluated our approach on the RadarScenes dataset, providing comparisons to other methods and supporting all claims made in this chapter. The experiments suggest that the proposed architecture achieves good performance on semantic segmentation of moving objects within single-scan point clouds and shows superior segmentation quality in diverse environments, even without exploiting temporal information. We assessed the different parts of our approach and compared them to other existing techniques. Overall, our approach outperforms state of the art both in F_1 score and mIoU, taking a step forward towards sensor redundancy for semantic segmentation for autonomous robots and vehicles.

Chapter 6

Moving Object Segmentation

The semantic segmentation of moving objects in sparse and noisy point clouds provides essential information for scene understanding. The differentiation of moving and static objects is especially important for safely navigating in dynamic, real-world environments. However, incorporating semantic information into the perception task comes with several limitations. Firstly, the number of points within the radar point clouds that belong to different semantic classes is unevenly distributed. Therefore, training deep neural networks is challenging since class imbalance can result in a bias in the trained model, which negatively impacts performance. Several methods exist to overcome the class imbalance problem, including optimized loss function and training strategies. Despite that, the segmentation performance of underrepresented classes is often limited, including the most vulnerable road users, like pedestrians. Secondly, the restricted number of semantic classes does not cover real-world long-tailed class distribution. State-of-the-art datasets [158, 189] typically include the most common classes, such as cars, trucks, pedestrians, and two-wheelers, which do not cover a real-world data distribution with a variety of classes of objects which might appear on the road such as strollers, animals, wheelchairs and scooters. Therefore, the limitation to specific classes poses a serious restriction for safety-relevant applications.

Additionally, the annotation process for a limited set of classes is not trivial, as the exact classification of objects within these classes is often unclear. For example, different annotators classify a medium-sized vehicle as a truck or car, which can result in inconsistencies within the training dataset, leading to performance degradation. To address both problems and overcome the inherent limitations, we address the task of moving object segmentation in radar point clouds. This task only requires the differentiation between the detection of moving and static objects. Moving detections comprise all possible moving objects within the surroundings of the autonomous vehicle, incorporating occasionally

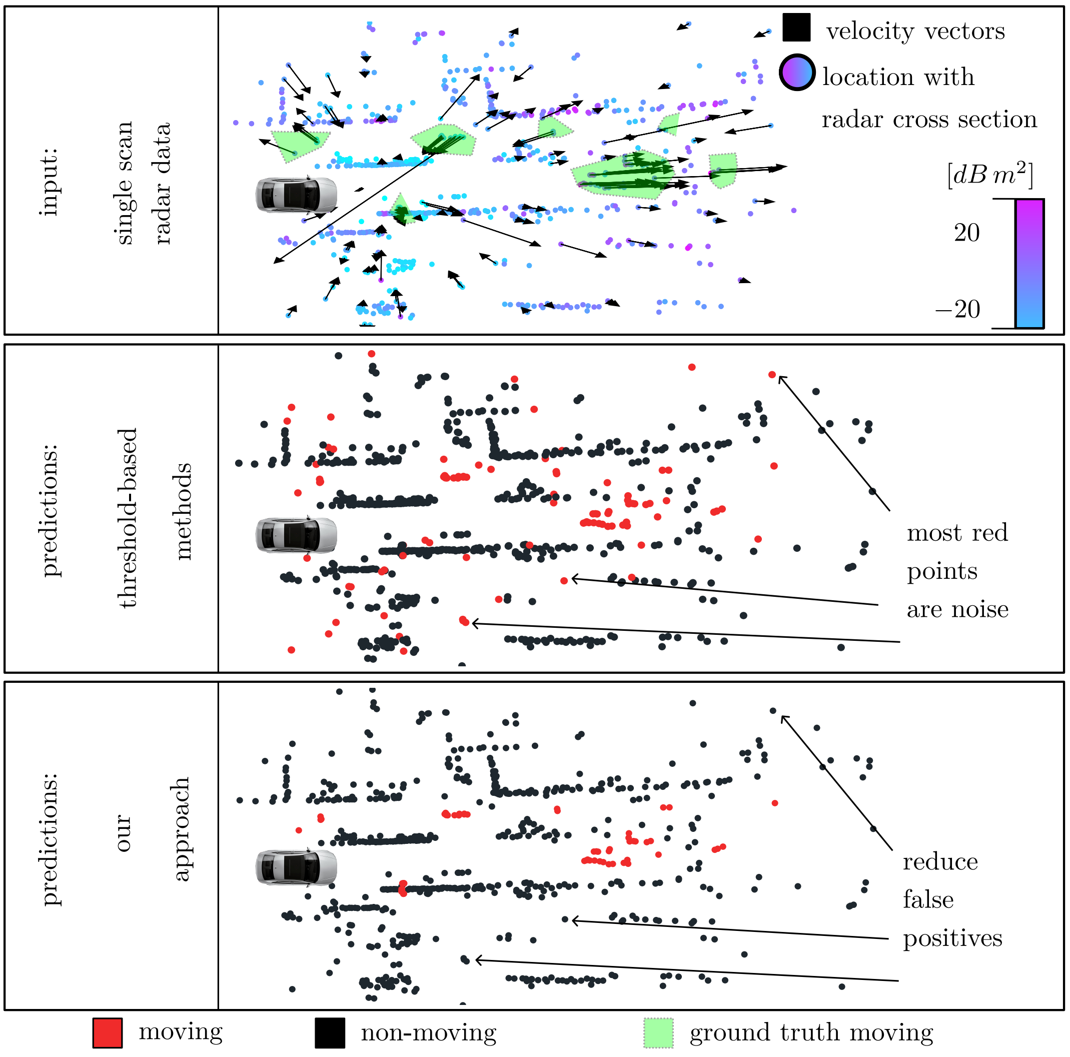


Figure 6.1: Our learning-based approach enhances moving object segmentation and reduces false positive predictions (bottom) from noisy, single-scan radar point clouds (top) compared to a velocity threshold determined on the validation set (middle).

appearing objects. Static or non-moving objects include the static environment, such as buildings and vegetation, but also parked vehicles. Additionally, the annotation process is simplified to derive the supervised labels for moving object segmentation because no differentiation between the semantic classes is required, reducing the labeling cost. Consequently, binary classification of moving and static objects comes with several advantages, and for most downstream tasks, such as collision avoidance, the correct semantic class of the object, such as car or truck, is not critical.

State-of-the-art methods for moving object segmentation for LiDAR and camera data rely on the extraction of temporal dependencies in videos [55] or aggregated residuals between previous scans [95]. Processing multiple scans, as done by Chen et al. [33] and Sun et al. [204], induces latency, which is unsuitable for a task requiring immediate information about the environment, such as collision warning. Furthermore, these approaches utilize 2D range image representations, which are inappropriate for radar sensors that do not provide height information. Mersch et al. [140] and Wang et al. [217] utilize a voxel-based approach to keep the 3D information of the point clouds intact and enhance moving object segmentation. However, these approaches also rely on the aggregation of multiple scans to identify moving objects reliably. We exceed these limitations and focus on the single scan processing of radar data. Moreover, the single-scan processing reduces the memory requirements, which is beneficial for real-world applications.

A drawback is that radar scans are affected by noise due to multi-path propagation, ego-motion, and sensor noise, as introduced in Section 2.3.5. The noisy measurements frequently lead to false positives and make simple methods that identify moving objects based on a Doppler velocity threshold [183] unacceptable, as visualized in Figure 6.1. In this chapter, we aim to investigate how additional sensor information, such as the Doppler velocity, can be exploited by learning-based approaches to reliably identify moving objects in the environment. Furthermore, the radar cross section, which depends on the material properties and the structure of the detection, supports the differentiation of closely connected objects. We investigate the processing of single, sparse radar point clouds by exploiting additional and valuable radar sensor information, including the Doppler velocity and radar cross section.

The main contribution of this chapter is a novel learning-based approach that accurately predicts moving objects in sparse, single-scan radar point clouds. Our approach, called Radar Velocity Transformer, predicts the semantic label of moving or non-moving for each point in the input radar scan. To classify the individual detection and extract valuable point-wise features, we introduce the velocity encoding in each module of our network. The encoding of the velocity enhances performance by injecting important information throughout the network. We optimize the feature aggregation in the decoder part with our transformer-based upsampling to adaptively merge features and capture complex local structures in sparse point clouds. Furthermore, we reorganized the RadarScenes [189] dataset, providing semantic classes for individual detection, which we transfer into moving and non-moving labels, establishing a single-scan benchmark.

For this approach, we make two key claims in this chapter: (i) Our approach is able to accurately perform moving object segmentation in single-scan, noisy radar point clouds and enhance the state of the art in moving object segmentation without exploiting temporal dependencies; (ii) The velocity encoding throughout the network and the transformer-based upsampling are essential to derive highly discriminative features and adaptively aggregate information to enhance accuracy.

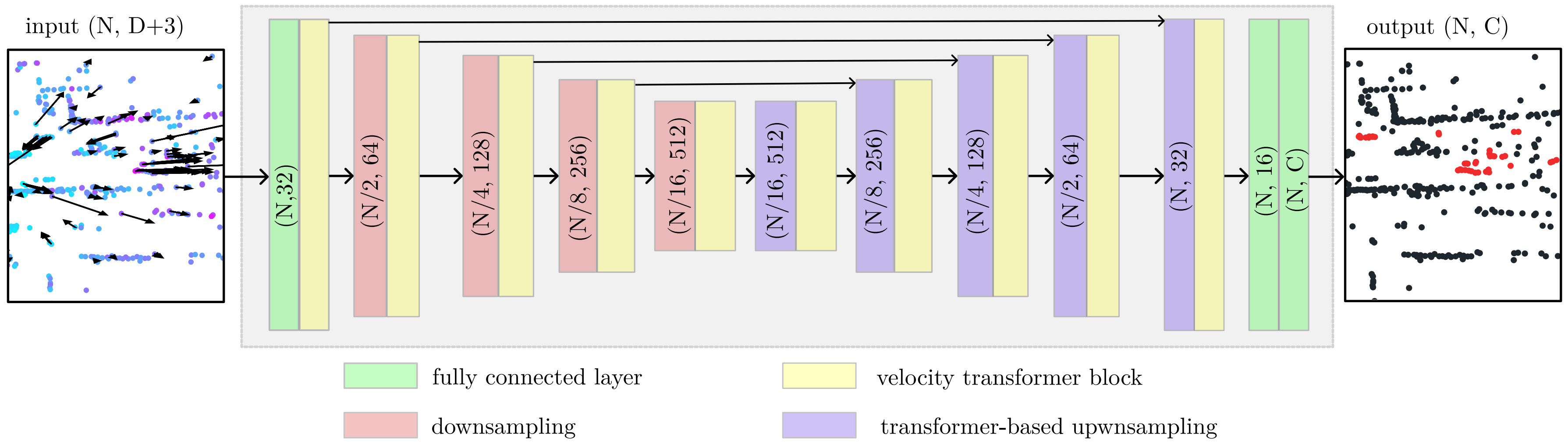


Figure 6.2: Architecture of our Radar Velocity Transformer for moving object segmentation. The fully connected layer first increases the dimension of the per-point features. The velocity transformer block incorporates the velocity transformer layer in a residual block to extract discriminative features from sparse radar data. Our downsampling and optimized transformer-based upsampling further improve the feature extraction. The final fully connected layer predicts the semantic class for the individual points in the single-scan radar point cloud.

6.1 Our Approach to Moving Object Segmentation

The goal of our approach is to achieve precise moving object segmentation in single-scan, sparse radar point clouds to enhance the environmental perception of autonomous vehicles. Figure 6.2 illustrates our Radar Velocity Transformer, which is a transformer-based framework that builds upon the successful self-attention mechanism [212] and directly processes the input point cloud to omit information loss. We incorporate the valuable Doppler velocity information within each module and use the so-called velocity transformer layer as the central building block of each encoder-decoder stage. Furthermore, we introduce transformer-based upsampling modules to adaptively combine local context information to enable fine-grained feature extraction.

6.1.1 Velocity Transformer Layer

In sparse radar point clouds, the information of individual detections can be of great benefit for solving downstream tasks such as moving object segmentation. Therefore, we follow Section 5.1.1 and introduce a velocity transformer layer to enhance feature extraction, as illustrated in Figure 6.3. Furthermore, the Doppler velocity is of central interest in determining moving objects, and the encoding of the velocity within the individual layers can help to perceive this information throughout the whole network.

In addition to the point coordinates $\mathbf{P}$ and the point-wise features $\mathbf{X}$, we include the Doppler velocities $\mathbf{v} = [v_1, \ldots, v_N]^\top \in \mathbb{R}^N$, where $v_i \in \mathbb{R}$, as separate inputs to our network. The idea is to support accurate moving object segmentation based on the relative velocity encoding $\mathbf{R}^v_{i,j} \in \mathbb{R}^{N \times N_{\text{vtl}}}$ since this enables the differentiation of nearby points and the identification of moving objects, where N_{vtl} is the number of points within the local areas. We determine the neighbors within the local area by kNN search with $k = N_{\text{vtl}}$. Hence, we first determine the relative velocities $\mathbf{r}^v_{i,j} = v_i - v_j$ of the neighboring points. We process the resulting relative velocities by two fully connected layers with weight matrices $\mathbf{W}^v_1 \in \mathbb{R}^{1\times 1}$, and $\mathbf{W}^v_2 \in \mathbb{R}^{1\times D}$, and the Gaussian error linear unit as an activation function [77]. We derive the encoded representation of the input features $\mathbf{X}$ following our Gaussian transformer layer, introduced in Section 5.1.1. We adopt the kNN algorithm with $k = N_{\text{vtl}}$ of the relative velocity encoding to derive the sampled queries $\mathbf{Q}^k$, keys $\mathbf{K}^k$, and values $\mathbf{V}^k \in \mathbb{R}^{N \times N_{\text{vtl}} \times D}$.

For the positional encoding, we adopt our approach and process the relative position $\mathbf{r}^p_{i,j} = \mathbf{p}_i - \mathbf{p}_j$ by an MLP, including two fully connected layers, to derive the positional encoding $\mathbf{R} \in \mathbb{R}^{N \times N_{\text{vtl}} \times D}$. In contrast to our Gaussian transformer

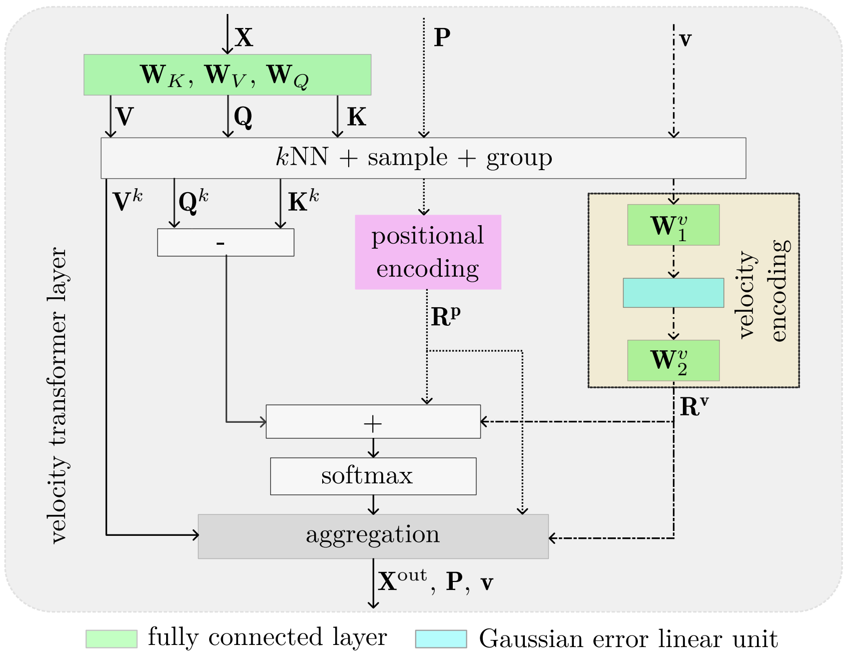


Figure 6.3: Design of the velocity transformer layer incorporates the vector attention mechanism, the velocity encoding, and the positional encoding. The fully connected layers with weight matrix $\mathbf{W}_Q$, $\mathbf{W}_K$, $\mathbf{W}_V$ encode the features as queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$. We use the kNN algorithm to determine local areas and sample and group the encodings accordingly to derive the sampled queries $\mathbf{Q}^k$, keys $\mathbf{K}^k$, and values $\mathbf{V}^k$. The positional encoding $\mathbf{R}^\mathbf{p}$ incorporates precise geometric information. Furthermore, we include the velocity encoding $\mathbf{R}^\mathbf{v}$ and process the Doppler velocities by two fully connected layers and Gaussian error linear unit activation function to provide important motion information for moving object segmentation. The output features $\mathbf{X}^{\text{out}}$ combine the information to derive meaningful representations.

layer, the attention scores $\mathbf{A}_{i,j}$ integrate the relative velocity encoding $\mathbf{R}^v_{i,j}$. We calculate the attention weights based on the vector attention of the queries and the keys, the relative positional encoding $\mathbf{R}^p_{i,j}$, and the relative velocity encoding $\mathbf{R}^v_{i,j}$, enabling fine-grained information aggregation within local areas. The final attention weights are determined by the softmax function:

$$\mathbf{A}^{i,j} = \operatorname{softmax}((\mathbf{Q}^k_i - \mathbf{K}^k_j) + \mathbf{R}^p_{i,j} + \mathbf{R}^v_{i,j}), \qquad (6.1)$$

where we normalize the weights within the local areas for each individual feature. We do not utilize the Gaussian function because the softmax function reduces the runtime due to optimized implementations. Further, the focus is to enhance the segmentation performance by incorporating the velocity information. We add the relative velocity encoding to the values and the relative positional encoding to derive the combined values $\mathbf{V}^c_{i,j} = \mathbf{V}^k_{i,j} + \mathbf{R}^p_{i,j} + \mathbf{R}^v_{i,j}$, which include and update the valuable information throughout the network. To derive the weighted

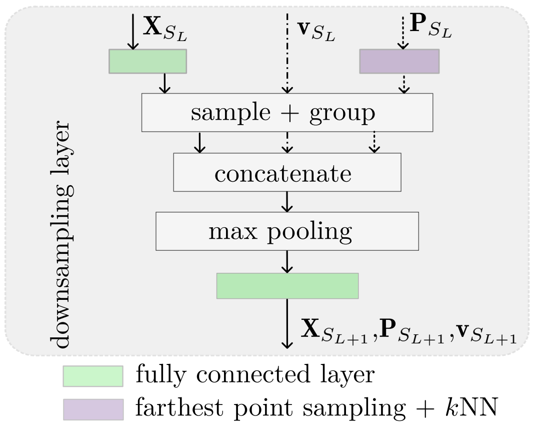


Figure 6.4: Detailed design of the downsampling module of our Radar Velocity Transformer. The input features $\mathbf{X}_{S_L}$ of stage S_L are processed by a fully connected layer before we sample and group the points based on the farthest point sampling and the kNN algorithm. The features, the Doppler velocities, and the point coordinates are concatenated. We process the resulting matrix by max pooling and a fully connected layer to derive the features of the downsampled point cloud.

features $\mathbf{X}^{\text{out}}$, we calculate the sum of the element-wise multiplication:

$$\mathbf{X}_j^{\text{out}} = \sum_{i=1}^{N_{\text{vtl}}} \mathbf{A}_{i,j} \odot \mathbf{V}_i^c, \tag{6.2}$$

within the local areas. The aggregated features, which are enriched by the velocity encoding, are directly processed by the following module to reduce the computational cost within the velocity transformer layer.

6.1.2 Velocity Transformer Block

Our velocity transformer block is a residual block [75], similar to the point transformer block [271] and the Gaussian transformer block, introduced in Section 5.1.2, that embeds the velocity transformer layer in the center of two fully connected layers. We add layer normalization [243] and a Gaussian error linear unit activation function for each fully connected layer. The features $\mathbf{X}_{S_L}$ of the individual stages S_L are processed by the velocity transformer layer and the fully connected layers to enrich the information of individual points within local areas. The velocity $\mathbf{v}_{S_L}$ and position data $\mathbf{P}_{S_L}$ are utilized to determine the relative encodings but are not further transformed to keep unaltered information throughout the network.

6.1.3 Downsampling Layer

The downsampling layer reduces the cardinality of the point cloud $\mathcal{P}_{S_{L+1}} \subset \mathcal{P}_{S_L}$ after each stage S_L and has to keep the most relevant information intact, as introduced in Section 3.3.1. Following Qi et al. [166], we adapt the max pooling operation depicted in Figure 6.4. We first process the feature vector $\mathbf{X}_{S_L}$ by a fully connected layer. To derive the local areas, we sample and group the points using the farthest point sampling and kNN algorithm. The features and the Doppler velocity values are sampled and grouped accordingly. To also induce valuable velocity information in the downsampling, we concatenate the features, the position, and the velocity information. Afterward, we apply max pooling to aggregate the information and process the feature vector by a fully connected layer. We reduce the number of points N_{S_L} by a factor of 2 and keep the position $\mathbf{P}_{S_{L+1}}$ and velocity information $\mathbf{v}_{S_{L+1}}$ of the downsampled point cloud to enrich the information in deeper layers.

6.1.4 Transformer-based Upsampling Layer

The common upsampling method interpolates the $k = 3$ nearest neighbors based on an inverse distance weighted average [166] and combines these with the features of the skip connection. Especially at the boundaries of moving objects, straightforward interpolation can result in a combination of features of different classes, which can harm the extraction of discriminative features, as explained in the previous chapter. Furthermore, the specific properties of the detections, including the Doppler velocity, are not considered during feature aggregation, which also applies to the Gaussian Radar Transformer. Hence, we argue that the upsampling and aggregation of the features in the decoder part of the network are crucial to enhance accuracy, especially for sparse point clouds. We propose a transformer-based upsampling layer to leverage the full potential of transformer-based approaches and utilize the velocity encoding within the full network visualized in Figure 6.5. In contrast to the attentive upsampling layer, introduced in Section 5.1.4, we incorporate an individual weighting of the features to improve the feature extraction for moving object segmentation.

The idea is to enable the network to learn how to concatenate important information by inter-attention to extract valuable features. Building on the attentive sampling, introduced in Section 5.1.4, the inter-attention leverages the full potential of the transformer architecture. The inputs are the output point cloud of the previous velocity transformer block $\mathcal{P}_{\text{up}}$, with the number of points N_{up}, which has to be upsampled, and the point cloud of the skip connection $\mathcal{P}_{\text{skip}}$, where $N_{\text{up}} \leq N_{\text{skip}}$.

Inspired by our velocity transformer layer and Gaussian transformer layer, we first encode the features $\mathbf{X}_{\text{up}}$ as keys $\mathbf{K}^{\text{up}}$, and values $\mathbf{V}^{\text{up}}$ and the features $\mathbf{X}_{\text{skip}}$ as queries $\mathbf{Q}^{\text{skip}}$. To determine the relative position and velocity encoding, we calculate the kNN for the point cloud of the skip connection $\mathcal{P}_{\text{skip}}$ within the point cloud $\mathcal{P}_{\text{up}}$, where $k = N_{\text{tus}}$.

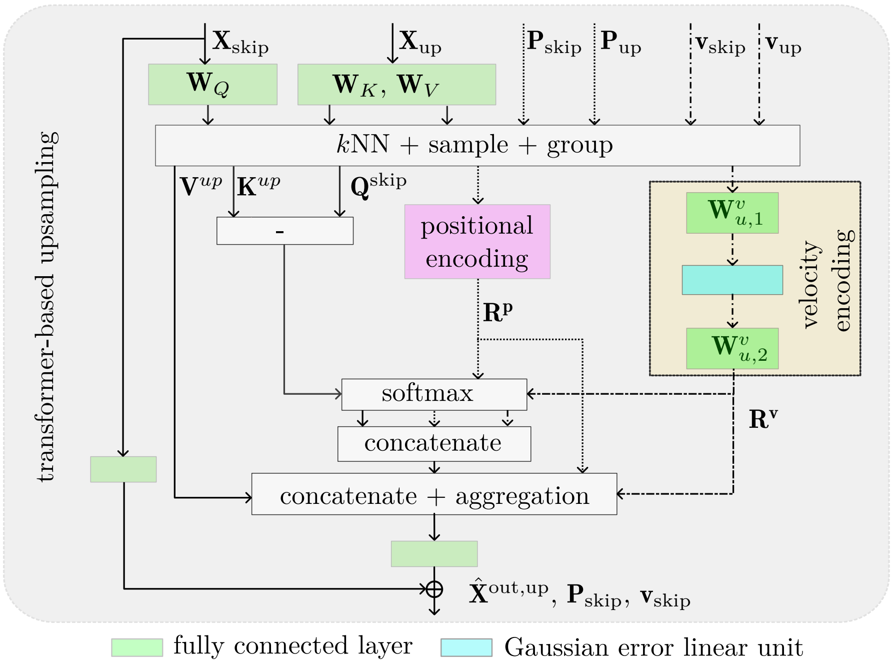


Figure 6.5: Detailed design of the transformer-based upsampling module of our Radar Velocity Transformer. The features $\mathbf{X}_{\text{up}}$ are encoded as keys $\mathbf{K}^{\text{up}}$ and values $\mathbf{V}^{\text{up}}$, and the features $\mathbf{X}_{\text{skip}}$ as queries $\mathbf{Q}^{\text{skip}}$. We determine three individual attention weights utilizing the softmax function for the relation of queries and keys $\mathbf{Q}^{\text{skip}} - \mathbf{K}^{\text{up}}$, the relative positional encoding $\mathbf{R}^{\text{p}}$ and the relative velocity encoding $\mathbf{R}^{\text{v}}$. The attention weights weigh the respective features. The combined and aggregated features are processed by a fully connected layer and added to the features of the point cloud of the skip connection to derive the upsampled features $\hat{\mathbf{X}}^{\text{out,up}}$. The output includes the point coordinates $\mathbf{P}_{\text{skip}}$ and velocity information $\mathbf{v}_{\text{skip}}$.

In the sample and grouping module, we compute the relative position and velocity of the correspondent points of the two point clouds. We determine the encodings by two fully connected layers with the Gaussian error linear unit activation function. In contrast to the velocity transformer layer, we calculate individual attention weights for the relation of queries and keys $\mathbf{A}^{qk}_{i,j} \in \mathbb{R}^{N\times N_{\text{tus}}\times D_{S_L}}$, the relative positional encoding $\mathbf{A}^{p}_{i,j} \in \mathbb{R}^{N\times N_{\text{tus}}\times 9}$, and the relative velocity encoding $\mathbf{A}^{v}_{i,j} \in \mathbb{R}^{N\times N_{\text{tus}}\times 3}$ to enable fine-grained information aggregation and enhance accuracy. The individual attention weights are determined by the softmax function as follows:

$$\mathbf{A}^{qk}_{i,j} = \text{softmax}(\mathbf{Q}^{\text{skip}}_{i,j} - \mathbf{K}^{\text{up}}_{i,j}), \tag{6.3}$$
$$\mathbf{A}^{p}_{i,j} = \text{softmax}(\mathbf{R}^{p}_{i,j}), \tag{6.4}$$
$$\mathbf{A}^{v}_{i,j} = \text{softmax}(\mathbf{R}^{v}_{i,j}). \tag{6.5}$$

We concatenate the individual attention weights to derive the final attention scores $\hat{\mathbf{A}}_{i,j} = (\mathbf{A}^{qk}_{i,j}, \mathbf{A}^{p}_{i,j}, \mathbf{A}^{v}_{i,j})$ where $\hat{\mathbf{A}}_{i,j} \in \mathbb{R}^{N\times N_{\text{tus}}\times D_{S_L}+12}$. To weight the respective information, the values $\mathbf{V}^{\text{up}}$ are concatenated with $\mathbf{R}^{p}_{i,j}$ and the velocity encoding $\mathbf{R}^{v}_{i,j}$ resulting in the combined values $\hat{\mathbf{V}}^{c}_{i,j} = (\mathbf{V}^{\text{up}}_{i,j}, \mathbf{R}^{p}_{i,j}, \mathbf{R}^{v}_{i,j}) \in \mathbb{R}^{N\times N_{\text{tus}}\times D_{S_L}+12}$.

To derive the weighted features $\mathbf{X}^{\text{out,up}}$, we calculate the sum of the element-wise multiplication:

$$\mathbf{X}_j^{\text{out,up}} = \sum_{i=1}^{N_{\text{tus}}} \hat{\mathbf{A}}_{i,j} \odot \hat{\mathbf{V}}_i^c, \tag{6.6}$$

within local areas. We process the aggregated features $\mathbf{X}^{\text{out,up}} \in \mathbb{R}^{N \times D_{S_L}+12}$ by a fully connected layer to compress the features to the input feature dimension D_{S_L} with a learnable weight matrix $\mathbf{W}_{\text{up}} \in \mathbb{R}^{(D_{S_L}+12)\times D_{S_L}}$:

$$\hat{\mathbf{X}}^{\text{out,up}} = \mathbf{X}^{\text{out,up}}\mathbf{W}_{\text{up}}, \tag{6.7}$$

where $\hat{\mathbf{X}}^{\text{out,up}}$ are the updated features for the upsampled point cloud. The fully connected layer enables the information exchange of the individual parts and reduces the complexity of the succeeding modules.

The final output of the transformer-based upsampling layer is the sum of the features $\mathbf{X}_{\text{skip}}$ and $\hat{\mathbf{X}}^{\text{out,up}}$, which incorporates the valuable information of both point clouds to derive discriminative features. The final output includes the point coordinates $\mathbf{P}_{\text{skip}}$ and the velocity information $\mathbf{v}_{\text{skip}}$ to keep the position and motion information intact. Inter-attention enables an adequate exchange of information, leveraging the properties of each point to overcome the limitations of interpolation. Furthermore, the position and velocity information is encoded within the upsampling process to enhance performance.

6.1.5 Network Architecture

We build our network architecture based on the widely-used U-Net [166, 271] with an encoder-decoder architecture including skip connections as illustrated in Figure 6.2. The input to the network are the features $\mathbf{x}_i$, the position information $\mathbf{p}_i$ with two spatial coordinates x_i^C, y_i^C, and the ego-motion compensated Doppler velocity v_i. The features include the position, the velocity, and additionally, the radar cross section σ_i resulting in a 4-dimensional vector $\mathbf{x}_i = \left[x_i^C, y_i^C, \sigma_i, v_i\right]^\top$. The inputs are first processed by an MLP before being passed to the first velocity transformer layer. The per-point features D_{S_L} are gradually increased within each stage from 32 to 64, 128, 256, and 512. The sampling operations change the cardinality by a factor of 2, resulting in $[N, N/2, N/4, N/8, N/16]$ points for the respective stage. The final output is determined by an MLP with two fully connected layers to obtain per-point logit values for the binary classification of moving and static points. The individual stages of our architecture each comprise one single velocity transformer block to build an efficient network.

6.2 Implementation Details

The Radar Velocity Transformer is implemented in PyTorch [161]. We train our model over 50 epochs with the AdamW [132] optimizer with an initial learning rate of 0.0005 and a cosine annealing learning rate scheduler [132]. We combine the Lovász loss [21] and the weighted cross-entropy for which we follow the approach by Schumann et al. [190] and set the weights for moving objects to 8.0 and for static ones to 0.5. The local areas for the velocity transformer layer are set to $N_{\text{vtl}} = 16$ and for the transformer-based upsampling to $N_{\text{tus}} = 12$. We train the network with one Nvidia A100 GPU and a batch size of 128. To reduce overfitting, we further apply data augmentation, including jitter, scaling, rotation, and instance augmentation.

6.3 Experimental Evaluation

The main focus of this chapter is accurate, single-scan moving object segmentation in sparse and noisy radar point clouds. We present our experiments to show the capabilities of our method to segment moving objects reliably. The results of our experiments also support our key claims, which are: Our approach (i) segments moving objects in radar point clouds more precisely compared to state-of-the-art methods and (ii) the velocity encoding and the transformer-based upsampling enhance the accuracy by incorporating valuable information throughout the network.

6.3.1 Experimental Setup

As in Chapter 5, we utilize the RadarScenes [189] dataset to train and evaluate our model. We use the same training, validation, and test split to ensure comparability. We transfer the semantic labels into binary labels, differentiating between moving and static objects. Following Chen et al. [33], we utilize the intersection over union (IoU) [54], introduced in Section 5.3.1, to evaluate the methods.

6.3.2 Moving Object Segmentation Performance

The first experiment evaluates the performance of our approach, and its outcome supports the claim that our approach enhances state-of-the-art moving object detection in sparse and noisy radar point clouds by utilizing only single scans.

To compare the results, we select the recently best-performing point-based segmentation method, our Gaussian Radar Transformer trained on only static and moving detections, introduced in Chapter 5, the Stratified Transformer [242] by Xin et al., which utilizes single scans, the 4DMOS network [140] by Mersch

Method	Input	IoU
Threshold $\lvert v_i\rvert > t$	single-scan	35.1
4DMOS [140]	multiple-scan	73.1
Stratified Transformer [242]	single-scan	74.6
Gaussian Radar Transformer (Chapter 5)	single-scan	79.1
Our Radar Velocity Transformer	single-scan	**81.3**

Table 6.1: Moving object segmentation results on the RadarScenes test set in terms of IoU for the moving class. The threshold $t = 0.92\,\mathrm{m/s}$ is determined on the validation set and afterward applied to the test set [183]. Our approach outperforms state-of-the-art approaches for moving object segmentation. The learning-based methods are superior compared to a simple threshold due to the noise in radar data. The best results are in bold.

et al. for LiDAR moving object segmentation which does not use the range representation because this is incompatible with the 2D coordinates, and a simple threshold for the velocity determined on the validation set [183]. For specific information on the training regime of the two networks, we refer to the original papers [140, 242].

The Radar Velocity Transformer outperforms the existing approaches, and the learning-based methods are superior compared to the threshold-based method, as displayed in Table 6.1. The difference between the learning-based approaches and the threshold-based method illustrates the necessity of advanced models to perform moving object segmentation in noisy radar point clouds. Additionally, the transformer-based methods enhance performance compared to the voxel-based 4DMOS, which suggests that discretization artifacts lead to information loss that cannot be compensated by additional temporal information of consecutive radar scans. The feature input vector of Stratified Transformer and Radar Velocity Transformer both contain valuable velocity information. However, our Radar Velocity Transformer considerably improves the IoU for moving objects by 6.7 absolute percentage points. Furthermore, we are able to outperform the Gaussian Radar Transformer, which illustrates that the velocity encoding and transformer-based upsampling enhance segmentation performance. Additionally, our approach performs well under adverse weather conditions, as illustrated in Figure 6.6. We reliably detect distant pedestrians and occluded vehicles within complex driving scenarios.

The strong performance for moving object segmentation benefits from radar data, which includes Doppler velocity information. Furthermore, the binary classification reduces class imbalance and addresses long-tailed class distributions. As a result, we enhance the performance compared to semantic segmentation in the previous chapter. To illustrate the advancement, we transfer the predictions

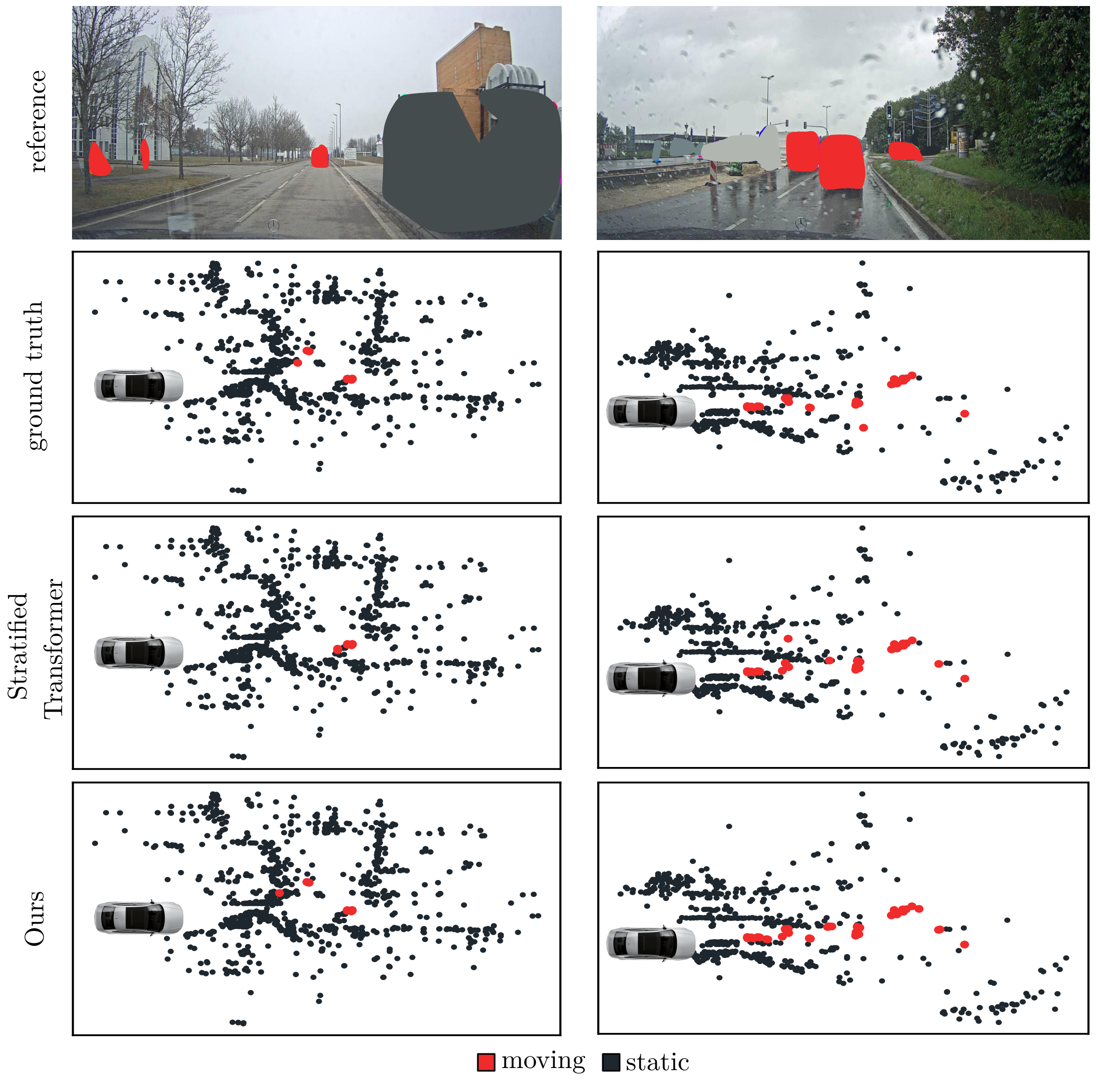


Figure 6.6: Qualitative results of the Stratified Transformer [242] and our Radar Velocity Transformer on the test set of RadarScenes [189]. Red points indicate moving objects and black points belong to static objects.

of our Radar Velocity Transformer into the true semantic class, resulting in a mIoU of over 90 %, which shows the benefits of moving object segmentation.

6.3.3 Ablation Study on Network Components

The second experiment, the ablation study on network components, evaluates the influence of the velocity encoding and transformer-based upsampling on the performance to support our second claim that our proposed modules each contribute to the improvements in terms of IoU. The combined results of the ablation study on the validation set are listed in Table 6.2.

To assess the benefits of transformer-based upsampling, we replace the module with the commonly used trilinear interpolation based on an inverse distance

#	velocity encoding	transformer-based upsampling	IoU
[A]			73.4
[B]		✓	75.2
[C]	✓		75.6
[D]	✓	✓	**77.4**

Table 6.2: Influence of the different components in terms of IoU for moving objects on the RadarScenes validation set. Transformer-based upsampling and velocity encoding are both important in enhancing moving object segmentation. The best results are in bold.

weighted average [166]. Since the velocity encoding is new and the information of the velocity of the individual detection is present in the feature vector $\mathbf{x}$, we remove the velocity encoding to evaluate the influence on the IoU.

Ablation [A], we replace the upsampling and remove the velocity encoding, which leads to a decrease in IoU by 4 absolute percentage points. In ablation [B], we add the transformer-based upsampling, which enables an adaptive feature aggregation of the two point clouds and leads to an improvement of IoU by 1.8 absolute percentage points. In comparison to ablation [A], we add the velocity encoding throughout the network in [C], which enhances performance. We assume that the velocity encoding is highly valuable since the fine-grained Doppler velocity information may be lost in high-level features of deeper layers. Hence, the specific task of moving object segmentation benefits from the velocity encoding. The final model of our Radar Velocity Transformer, represented in [D], further enhances the IoU by using both velocity encoding and transformer-based upsampling. We conclude that the velocity encoding supports the aggregation of the features for the upsampling and hence leads to the best results.

As an additional experiment, we replaced the concatenation of the transformer-based upsampling with an addition in our final Radar Velocity Transformer. The obtained IoU of 75.3 % indicates that the concatenation leads to a more fine-grained weighting of the individual channels and improves performance. Additionally, we exploit transformer-based downsampling. However, this does not improve the overall performance, and hence, we keep the max pooling since it is more efficient and does not mix information, which is suitable for the downsampling for the task of moving object segmentation.

6.3.4 Runtime

Finally, we analyze the runtime of our approach and show that our approach runs fast enough to support online processing in the vehicle, which is important for

model	parameters (M)	mean runtime (ms)
4DMOS [140]	1.8	14.0
Stratified Transformer [242]	8.0	34.4
Gaussian Radar Transformer (Chapter 5)	8.4	24.0
Ours	3.4	12.0

Table 6.3: Evaluation of the number of parameters and mean runtime of the models on an Nvidia RTX A6000 GPU based on 1000 randomly sampled point clouds of the RadarScenes validation set. Our approach has a lower runtime than other state-of-the-art approaches.

deployment. We tested our approach and the baselines on an AMD Threadripper CPU with an Nvidia RTX A6000 GPU, as explained in Section 5.3.5. Our implementations include an optimized farthest point sampling and kNN algorithm in C++ to speed up the inference and preserve consistency within the chapters. We report the results of all approaches in Table 6.3. The mean runtime is 0.012 s, which is equal to 83 Hz, and thus over 4x faster than the frame rate of 17 Hz of the sensor. Furthermore, our Radar Velocity Transformer runs faster than the Stratified transformer, 4DMOS, and our Gaussian Radar Transformer. It is remarkable that 4DMOS achieves comparable runtime but uses denser point clouds. Therefore, the voxelization and optimized 4D convolutions are beneficial. However, the voxelization leads to worse performance in terms of IoU, which is why the point-based approaches are preferable. Furthermore, our Radar Velocity Transformer uses fewer parameters than the Stratified Transformer and Gaussian Radar Transformer, which reduces memory requirements.

In summary, our evaluation supports our statement that our method provides competitive moving object segmentation performance in sparse, single-scan radar processing. At the same time, our method efficiently incorporates the Doppler velocity information within the individual modules of the network, outperforming state-of-the-art approaches. Thus, we support all our claims with this experimental evaluation.

6.4 Conclusion

In this chapter, we presented a novel approach to accurately perform single-scan moving object segmentation in the domain of radar data. The awareness of moving objects in the surroundings of a self-driving vehicle is essential for safe and reliable autonomous navigation. The interpretation of LiDAR and camera data achieves exceptional results but typically requires accumulating and processing temporal sequences of data in order to extract motion information. In contrast, radar sensors, which are already installed in most recent vehicles today, can overcome this limitation as they directly provide the Doppler velocity of the detections and, hence, incorporate instantaneous motion information within a single scan. The key to our Radar Velocity Transformer is to incorporate the valuable velocity information throughout each module of the network, thereby enabling the precise segmentation of moving and non-moving objects. Additionally, we propose a transformer-based upsampling, which enhances performance by adaptively combining information and overcoming the limitation of interpolation of sparse point clouds. Our novel transformer-based approach does not rely on exploiting temporal dependencies to identify moving objects and enables us to perform single-scan moving object segmentation in sparse radar scans accurately. Finally, we create a new radar moving object segmentation benchmark based on the RadarScenes dataset and compare our approach to other state-of-the-art methods.

The experiments and the comparisons to other approaches support all claims made in this chapter and suggest that our architecture achieves superior performance on moving object segmentation in noisy, single-scan point clouds obtained from automotive radars. The sensors used for recording the RadarScenes dataset are series sensors already implemented in vehicles, which makes our approach applicable without additional cost. Overall, our approach outperforms the state-of-the-art methods and proposes advanced modules for radar data processing, taking a step forward towards reliable single-scan moving object segmentation and sensor redundancy for autonomous vehicles.

Chapter 7

Moving Instance Segmentation

In the previous chapter, we introduced our approach to segment moving objects within sparse and noisy radar data. Moving object segmentation focuses on the classification of the individual detections as moving and static but lacks valuable information about the number of agents present within the surroundings of a self-driving vehicle. To navigate safely, information about the instances plays a crucial role in handling interactions with other traffic participants and influences the decision-making for specific maneuvers in path planning, such as lane changes or overtaking. Therefore, knowledge about the agents is mandatory for safety-relevant applications.

In this chapter, we focus on the problem of moving instance segmentation in sparse and noisy radar point clouds, as depicted in Figure 7.1. The task combines instance segmentation and moving object segmentation into a panoptic problem, including the differentiation of object instances, such as parked and moving cars. The moving instances represent the "things" and the static points the "stuff". For many tasks, the information, if an object is moving in combination with the velocities, is often sufficient, substantially simplifying the labeling efforts as explained in Chapter 6.

Furthermore, semantic information can be exploited to reason about the mobility of objects, but it also entails the aforementioned "long-tail" problem; i.e., no matter how much data we collect, we will always have semantic classes that are underrepresented or not even covered by the training data. Therefore, we restrict the instance segmentation within the moving or non-moving instances that are essential for autonomous driving and allow it to potentially cover long-tail classes.

State-of-the-art methods often address the task of moving instance segmentation separately as moving object segmentation [33] and instance segmentation [125]. Chen et al. [33] and Kim et al. [95] focus on moving object segmentation and pass aggregated scans through the whole network, which induces

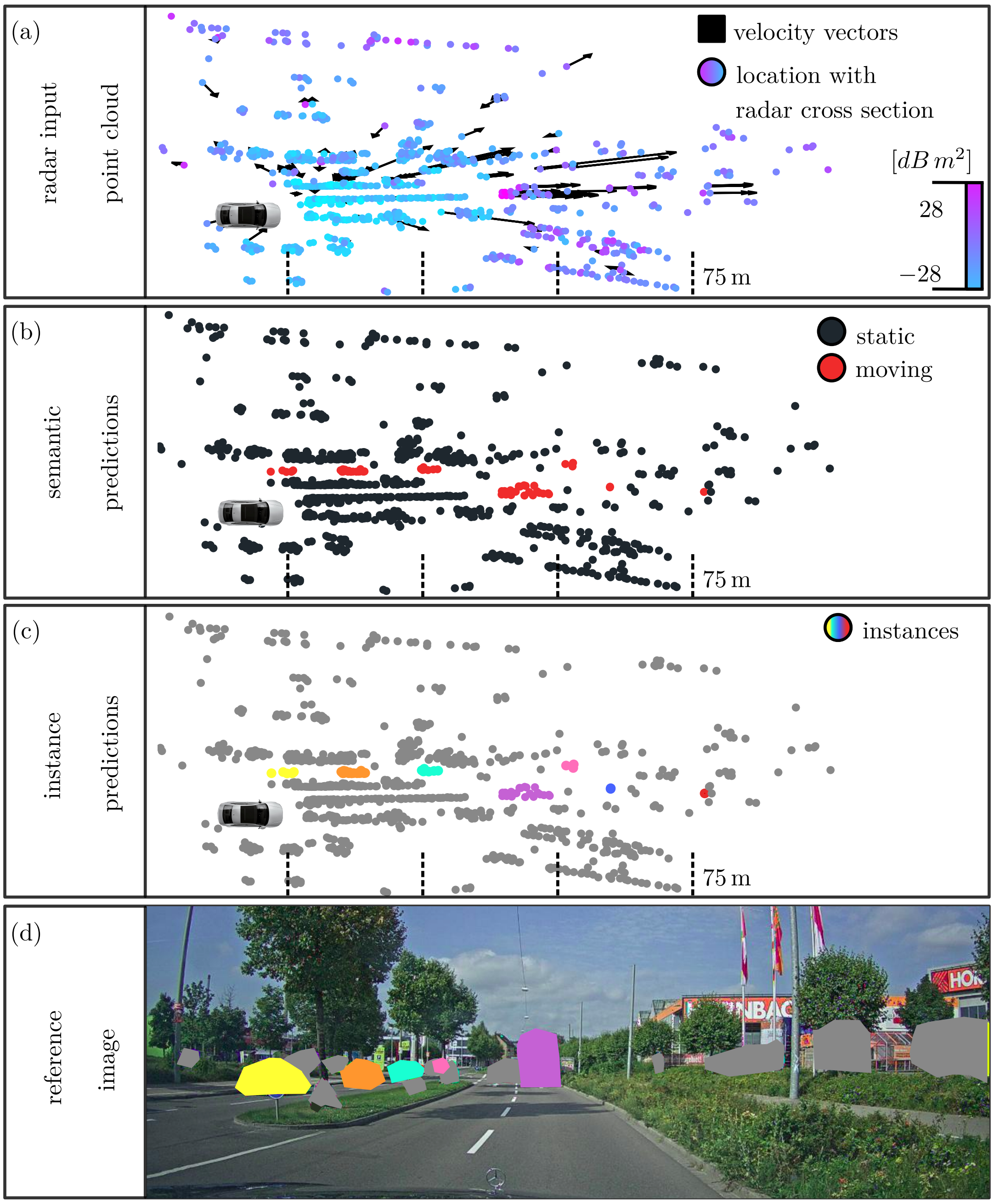


Figure 7.1: Our approach combines moving object segmentation (b) and instance segmentation (c) to solve the panoptic task of moving instance segmentation from sparse radar point clouds (a). The reference image in (d) illustrates the scene and includes privacy-preserving colored masks. In the image of the point cloud (c), each color represents a different instance of moving objects (static is grey). The colors in (c) and (d) correspond if the object is visible.

latency and is disadvantageous for a task requiring immediate feedback, such as collision avoidance. Recent instance segmentation approaches, as proposed by Schult et al. [187], Vu et al. [214], and Xie et al. [240], work on single inputs, neglecting the temporal information, and do not differentiate between moving and static objects. The temporal information is especially helpful for radar point clouds since per-scan data is sparse. Furthermore, most noise points are directly identifiable within the temporal domain because these points fluctuate over time. In addition, Gasperini et al. [59] and Li et al. [118] rely on specific semantic labels to perform instance segmentation, which are absent in the context of moving instance segmentation since all objects belong to the moving class. Our approach aims to overcome these limitations and exploits the valuable temporal information in an effective manner.

The main contribution of this chapter is a novel approach that combines moving object segmentation and instance segmentation within a single network and accurately predicts moving instances in sparse and noisy radar point clouds. Our approach, called Radar Instance Transformer, predicts for each point in the input radar scan if it is moving or static and assigns an instance ID to each moving detection. Thus, we unify moving object segmentation of Chapter 6 and instance segmentation. To reliably identify moving instances, we efficiently incorporate the temporal information within the single current scan by our sequential attentive feature encoding module without passing aggregated scans through the whole network. We optimize the network architecture in terms of efficiency and accuracy by processing the point cloud with the original resolution throughout the network to keep as much information as possible and enrich the points with high-level features. We utilize local and global attention to include instance information and propose a graph-based instance assignment to improve performance. In contrast to typical LiDAR approaches, our individual modules tackle the challenges of sparsity and noise of radar point clouds that make radar data interpretation comparably challenging. Besides, we extended the RadarScenes dataset and transferred it into the first moving instance segmentation benchmark for point clouds and published it publicly at `https://doi.org/10.5281/zenodo.10203864`.

In sum, we make five claims in this chapter: First, our approach shows state-of-the-art performance for moving instance segmentation in sparse and noisy radar point clouds without passing multiple scans through the whole network, reducing the memory requirements compared to aggregation approaches. Second, our sequential attentive feature encoding extracts valuable temporal information by enriching the features of individual points to enhance accuracy. Third, our attentive instance transformer head is able to incorporate essential instance information, which improves the overall performance. Fourth, our attention-based graph partitioning enhances instance assignments without requiring class-dependent semantic information. Fifth, our backbone enables valuable feature extraction for sparse radar point clouds by processing the full-resolution point cloud, i.e., the original number of points throughout the network, to enhance the accuracy and reduce the number of parameters.

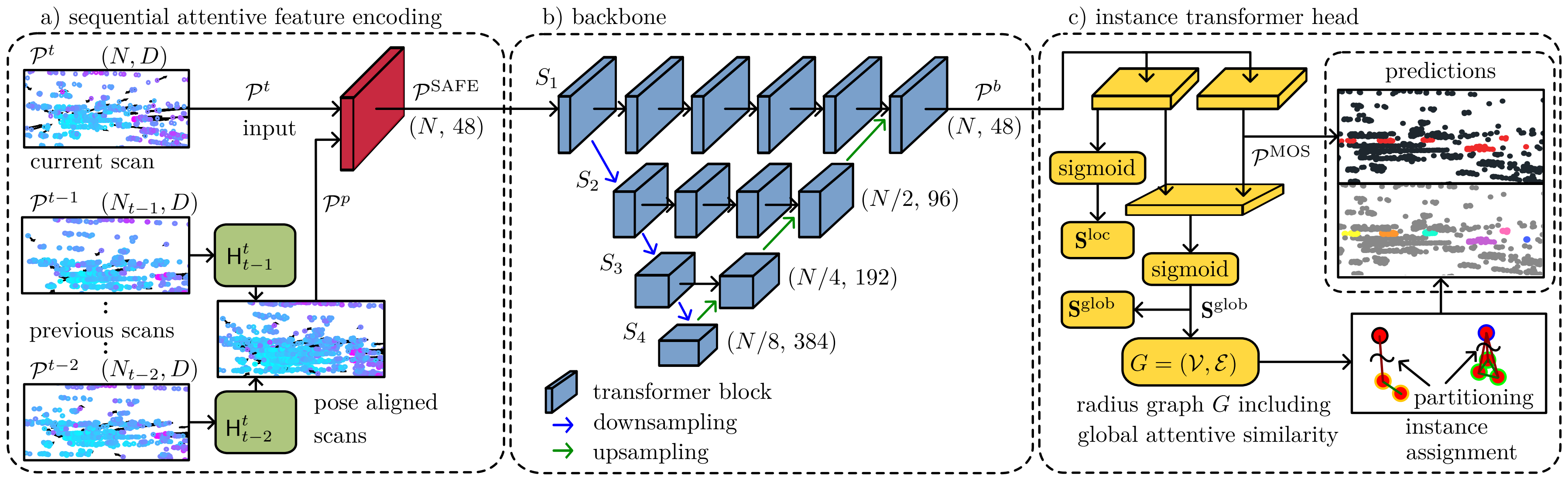

a) sequential attentive feature encoding
$\mathcal{P}^t$ (N, D)
current scan
$\mathcal{P}^t$
input
$\mathcal{P}^{\text{SAFE}}$
$(N,\ 48)$
$\mathcal{P}^{t-1}$ (N_{t-1}, D)
previous scans
$\mathcal{P}^{t-2}$ (N_{t-2}, D)
H^t_{t-1}
H^t_{t-2}
$\mathcal{P}^p$
pose aligned scans
b) backbone
S_1
S_2
S_3
S_4
$\mathcal{P}^b$
$(N,\ 48)$
$(N/2,\ 96)$
$(N/4,\ 192)$
$(N/8,\ 384)$
transformer block
downsampling
upsampling
c) instance transformer head
predictions
sigmoid
$\mathbf{S}^{\text{loc}}$
$\mathcal{P}^{\text{MOS}}$
sigmoid
$\mathbf{S}^{\text{glob}}$
$\mathbf{S}^{\text{glob}}$
$G = (\mathcal{V}, \mathcal{E})$
radius graph G including global attentive similarity
partitioning
instance assignment

Figure 7.2: Detailed design of each module of our Radar Instance Transformer. (a) The sequential attentive feature encoding module combines the pose-aligned point clouds and their corresponding features using the current scan into a featurized point cloud, where the features encode temporal information. (b) This aggregated feature volume is then processed in a multi-pathway transformer backbone, where the transformer layers process the information at different resolutions and aggregate per-point features. (c) The per-point features are then used to estimate the semantics $\mathcal{P}^{\text{MOS}}$ and perform graph partitioning for instance alignment based on the global similarity quantity $\mathbf{S}^{\text{glob}}$. The local similarity $\mathbf{S}^{\text{loc}}$ includes the differentiation of instances and the static environment. The tuples denote the number of points and feature channels.

7.1 Our Approach to Moving Instance Segmentation

Our goal is to achieve precise moving instance segmentation in a sequence of noisy, sparse radar point clouds to enhance scene interpretation of autonomous vehicles. Figure 7.2 illustrates the overall architecture of our Radar Instance Transformer, which is a point-based framework and builds upon the self-attention mechanism [212]. We incorporate temporal relationships in a computationally efficient way to enrich the features of the current point cloud, as detailed in Section 7.1.1. In Section 7.1.2, we propose a new backbone with four stages S, processing the enriched full-resolution point cloud throughout the network to extract valuable features and omit information loss. We utilize local and global information derived from self-attention to improve the panoptic task of instance association and semantic segmentation, which we explain in Section 7.1.3. Finally, our graph-based instance assignment, detailed in Section 7.1.4, incorporates global instance information to enhance accuracy.

7.1.1 Sequential Attentive Feature Encoding

Temporal information is essential to enhance scene interpretation in sparse and noisy radar point clouds. Especially for moving instance segmentation, the temporal relationship helps to identify agents reliably. Additionally, temporal dependencies support the differentiation of moving and static detections, including noise directly identifiable in the temporal domain due to the often changing appearance in sequences of scans. In contrast to other approaches [95, 140] that pass multiple point clouds through the whole network, we propose the sequential attentive feature encoding module to efficiently enrich the features of a single point cloud with temporal information. Therefore, the previous scans are only processed within the sequential attentive feature encoding module, and we only pass the current scan through the whole network. Consequently, we do not increase the number of points that need to be processed by the network. The goal is to adaptively combine the temporal information of the previous scans with the features of the current point cloud, hence reducing the computational burden but still keeping important information.

The input to our sequential attentive feature encoding module is the current scan $\mathcal{P}^t$ at time t and T previous scans $\mathcal{P}^{t-T}, \dots, \mathcal{P}^{t-1}$ of a sequence of radar scans, as depicted in Figure 7.2. As explained in Section 5.1.1, the current scan $\mathcal{P}^t$ includes the point coordinates $\mathbf{P}$ and point-wise features $\mathbf{X}$ with input feature dimension D. Additionally, we aggregate the previous scans into a single scan, which comprises the $N_{t-T} + \cdots + N_{t-1}$ points of all the combined T scans

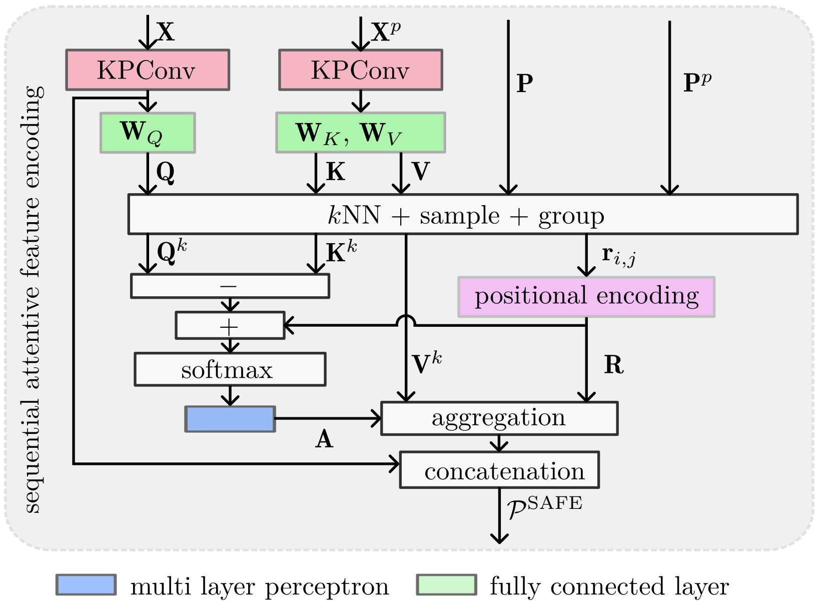


Figure 7.3: The sequential attentive feature encoding module enriches the current point cloud $\mathcal{P}^t$, which includes the point coordinates $\mathbf{P}$ and point-wise features $\mathbf{X}$, with the information of the previous point clouds $\mathcal{P}^p$, which include the point coordinates $\mathbf{P}^p$ and point-wise features $\mathbf{X}^p$. After the KPConv layer [210], we extract the queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$. The final attention weights $\mathbf{A}$ combine vector attention with relative positional encoding $\mathbf{R}$ to derive valuable temporal information from the previous point clouds.

with point coordinates $\mathbf{P}^p$ and point-wise features $\mathbf{X}^p$. We keep the original number of points for each individual point cloud. To align the previous scans with the current scan locally, we assume that the homogeneous relative pose transformation $\mathsf{H}^t_{t-1} \in \mathbb{R}^{4\times4}$ between point cloud $\mathcal{P}^t$ and $\mathcal{P}^{t-1}$ is given. The transformation can be derived from global navigation satellite systems, online simultaneous localization and mapping algorithms [2, 15, 28, 266], or using wheel encoders and inertial measurement units, as explained in Section 2.3.7. In our work, we use the provided vehicle odometry and differential global positioning system information. The alignment helps to combine the information of the related instances over time and supports the identification of noise within a sequence of scans. The idea is to align the static points within the sequence of scans and learn to identify noise points due to their changing appearance over time. Additionally, moving instances change their location over time, and small displacements of the respective detections can help identify moving instances. Since the data association of individual radar points within unprocessed point clouds is difficult [238, 242], we first process the point-wise features of the current scan $\mathbf{X}$ and the features of the previous scans $\mathbf{X}^p$ with a KPConv layer [210] to extract higher dimensional features $\mathbf{x}_i \in \mathbb{R}^{D_1}$, as depicted in Figure 7.3. The aggregated features of the previous scans and the features of the current scan are processed by a separate KPConv layer. The resulting features are still point-wise features with dimension D_1. To adaptively combine the features and extract temporal information, we introduce an intra-attention module that is inspired by our Gaussian transformer layer, introduced in Section 5.1.1, and the transformer-based upsampling module, introduced in Section 6.1.4. We first encode the features of the previous scans $\mathbf{X}^p$ as values $\mathbf{V}$ and keys $\mathbf{K}$ and the features of the current scan $\mathbf{X}$ as queries $\mathbf{Q}$. In contrast to our previous approaches, we increase the feature dimension of the encoding to D_2 to enable fine-grained information aggregation resulting in the weight matrices $\mathbf{W}_Q \in \mathbb{R}^{D_1 \times D_2}$, $\mathbf{W}_K \in \mathbb{R}^{D_1 \times D_2}$, and $\mathbf{W}_V \in \mathbb{R}^{D_1 \times D_2}$.

Since temporal information is present within the local neighborhood, we restrict the intra-attention of the current scan and the aligned previous scans to local areas, similar to our transformer-based upsampling. To determine the local areas, we calculate the k nearest neighbors with $k = N_{\text{rit}}$ for the points in the current point cloud $\mathcal{P}^t$ within the aligned past point clouds $\mathcal{P}^p$. This results in the combination of the respective detections over time. To extract the related information of the queries, values, and keys within the local areas, we utilize the sample and grouping algorithm [166] resulting in $\mathbf{Q}^k, \mathbf{K}^k$, and $\mathbf{V}^k \in \mathbb{R}^{N \times N_{\text{rit}} \times D_2}$. The queries comprise the repeated entries of the current point cloud, and the keys and values include the encoded features of the past point clouds, which belong to the corresponding local neighbors of the N points of the current point cloud.

As described in Section 6.1.4, we include the relative positional encoding. However, we adjust the feature dimensions to include more fine-grained position information. We compute the positional encoding $\mathbf{R} \in \mathbb{R}^{N \times N_{\text{rit}} \times D_2}$ by an MLP, including two fully connected layers with weight matrices $\mathbf{W}_{R_1} \in \mathbb{R}^{3 \times 3}$ and $\mathbf{W}_{R_2} \in \mathbb{R}^{3 \times D_2}$, batch normalization [89], and rectified linear unit activation function [150]. To calculate the attention weights $\mathbf{A} \in \mathbb{R}^{N \times N_{\text{rit}} \times D_2}$ for the individual points, we follow our previous approaches and combine the vector attention and positional encoding to derive a fine-grained weighting of the individual feature channels. The attention weights connect the detections in the current scan with the detections in the aligned previous scans.

In contrast to the previous chapters, we process the attention weights by an MLP with two fully connected layers, where each layer is followed by batch normalization [89] to derive the final attention weights $\hat{\mathbf{A}}_{i,j}$. We apply the softmax function to each feature of the points within the local area. To aggregate the weighted features $\mathbf{X}^{\text{temp}}$, which comprise the temporal information of previous scans encoded as values, we calculate the sum of the element-wise multiplication, indicated by $\odot$, as:

$$\mathbf{X}_i^{\text{temp}} = \sum_{j=1}^{N_{\text{rit}}} \hat{\mathbf{A}}_{i,j} \odot (\mathbf{V}_{i,j}^k + \mathbf{R}_{i,j}). \tag{7.1}$$

We include the positional encoding as before because it provides valuable information about the local areas and their appearance over time. The idea is that static objects remain in the same location, whereas moving objects change position within a sequence of scans, which both can be represented within the positional encoding. To combine the information within the current scan, we concatenate the temporal features $\mathbf{X}^{\text{temp}} \in \mathbb{R}^{N \times D_2}$ and the features of the current scan $\mathbf{X} \in \mathbb{R}^{N \times D_1}$ after the KPConv layer resulting in the temporal enriched features $\mathbf{X}^{\text{SAFE}} \in \mathbb{R}^{N \times (D_1 + D_2)}$ of our sequential attentive feature encoding module. Besides the features, the output point cloud $\mathcal{P}^{\text{SAFE}}$ includes the point coordinates $\mathbf{P}^{\text{SAFE}}$, where $\mathbf{P}^{\text{SAFE}} = \mathbf{P}$. The number of points N equals the number of points in the current point cloud. In contrast to aggregation, the number of points processed by the backbone remains the same. The point coordinates are not processed to include fine-grained position information within the consecutive network. The sequential attentive feature encoding module is model-agnostic and can be applied to different backbones.

7.1.2 Backbone

With our backbone design, we aim to keep the full resolution, i.e., the original number of points of the current scan throughout the network, and thus avoid information loss and extract valuable features from sparse radar scans, as illustrated in Figure 7.2. Contrary to the widely-used U-Net [166, 271] with an encoder-decoder architecture including skip connection, introduced in Section 3.3.1, we process the full-resolution point cloud in parallel to the downsampled feature maps and enrich the information in the original point cloud with high-level features. In contrast to 3D LiDAR scans, the sparsity of radar data enables processing full-resolution point clouds but also requires keeping as much information as possible to enhance performance. To handle the changing number of points in radar scans, we adopt the advanced batching algorithm [271], which handles the batch of point clouds as a concatenated vector to always keep the original number of points. Compared to the first parallel network for image analysis, the HRNet [216], we reduce the processing of the high-level features to a minimum and do not keep the parallel tracks throughout the network. Considering that the processing of these features is computationally expensive, HRNet reduces the input size to keep the parallel feature extraction, which is in contrast to our approach that preserves the information of the point cloud. Consequently, our design reduces the number of parameters and computational complexity.

The input point cloud $\mathcal{P}^{\text{SAFE}}$ of our backbone includes the temporally enriched features $\mathbf{X}^{\text{SAFE}}$ and the position information $\mathbf{P}$ of the current scan. The dimension of the features for the stages S_L with $L = 1, \dots, 4$ are D_{S_L}, where $D_{S_1} = (D_1 + D_2)$, $D_{S_2} = 2D_{S_1}$, $D_{S_3} = 2D_{S_2}$, and $D_{S_4} = 2D_{S_3}$. The individual building blocks of our backbone are the commonly used transformer blocks as well as up- and downsampling layers. In contrast to the Gaussian Radar Transformer and Radar Velocity Transformer, introduced in Chapter 5 and in Chapter 6, we focus on the design of the backbone structure instead of the specific layers. The idea is to achieve better performance by optimizing the arrangement of the layers and reducing the computation within the transformer blocks.

The stages S_1, S_2, S_3, and S_4 comprise 6, 4, 2, and 1 transformer blocks, respectively. The transformer block is a residual block that follows the design of the Stratified Transformer block [242], where we replace the window attention with the Point Transformer layer [271] since it is computationally more efficient. Within the block, we first process the input features by layer normalization [8] before feeding the features and position information into the transformer layer. The resulting output is processed by two fully connected layers, including the Gaussian error linear unit [77] activation function and layer normalization. We add the features of the residual path to the updated features to determine the output features.

The transformer layer adopts the basic vector attention, introduced in Section 3.3. Therefore, we encode the features $\mathbf{X}^{\text{SAFE}}$ as values, keys, and queries. In contrast to our sequential attentive feature encoding module, we do not increase the dimension of the encodings by the learned linear projections. Hence, we set the input and output dimensions of the fully connected layers to D_{S_L}. As a result, we need to adjust the output dimension of $\mathbf{W}_{R_2}$ to D_{S_L}. We calculate the final output features of the transformer layer based on Equation (7.1). The point coordinates are not further processed to keep the information intact and improve the features with fine-grained position information within consecutive blocks.

For the downsampling, we follow Section 3.3.1 and utilize farthest point sampling to derive a subset of points based on the distance to cover the geometric space evenly. Our idea is to minimize computational effort during sampling, thereby reducing runtime and improving accuracy through our optimized network architecture. We utilize kNN search with $k = N_d$ and max pooling [166] to combine the local features. The top level does not include downsampling since we want to keep the original number of points. From stage S_L to stage S_{L+1}, we sample $N_{S_{L+1}}$ points where $N_{S_{L+1}} = N_{S_L}/2$. We reduce the cardinality of the point cloud by a factor of 2 instead of 4 [166, 271] to keep more information since radar point clouds are sparse, resulting in $[N, N/2, N/4, N/8]$ points for the four stages, where N is the number of points in the current scan. Based on the local areas, we aggregate the features and process them by a fully connected layer with weight matrix $\mathbf{W}_{S_L}^{\text{down}} \in \mathbb{R}^{D_{S_L} \times D_{S_{L+1}}}$ where $D_{S_{L+1}} = 2\,D_{S_L}$ and layer normalization. Afterward, we apply max pooling to derive the high-level features of the downsampled point cloud and pass the features and the point coordinates of the sampled points to the consecutive stage.

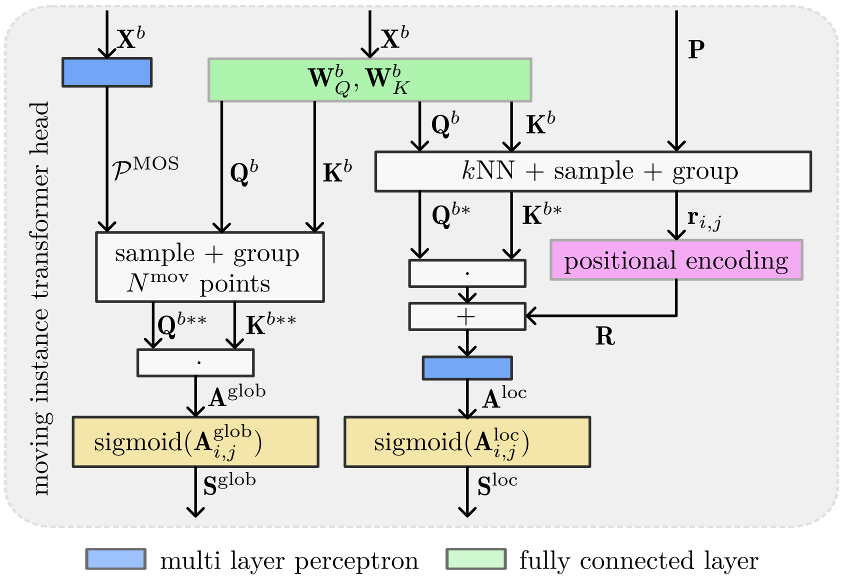

moving instance transformer head
$\mathbf{X}^b$
$\mathbf{X}^b$
$\mathbf{P}$
$\mathbf{W}^b_Q, \mathbf{W}^b_K$
$\mathcal{P}^{\mathrm{MOS}}$
$\mathbf{Q}^b$
$\mathbf{K}^b$
$\mathbf{Q}^b$
$\mathbf{K}^b$
kNN + sample + group
$\mathbf{Q}^{b*}$
$\mathbf{K}^{b*}$
$\mathbf{r}_{i,j}$
positional encoding
sample + group N^{mov} points
$\mathbf{Q}^{b**}$
$\mathbf{K}^{b**}$
$\cdot$
$+$
$\mathbf{R}$
$\mathbf{A}^{\mathrm{glob}}$
$\mathbf{A}^{\mathrm{loc}}$
sigmoid($\mathbf{A}^{\mathrm{glob}}_{i,j}$)
sigmoid($\mathbf{A}^{\mathrm{loc}}_{i,j}$)
$\mathbf{S}^{\mathrm{glob}}$
$\mathbf{S}^{\mathrm{loc}}$
multi layer perceptron
fully connected layer

Figure 7.4: Detailed design of the computation of the attentive similarity matrices $\mathbf{S}^{\mathrm{loc}}$ and $\mathbf{S}^{\mathrm{glob}}$. The module incorporates essential instance information derived from the point coordinates $\mathbf{P}$ of the current point cloud and point-wise features $\mathbf{X}^b$ extracted in the backbone. The calculation is based on the dot-product attention, including the queries $\mathbf{Q}^b$, keys $\mathbf{K}^b$, and the relative positional encoding $\mathbf{R}$.

For the upsampling, we adopt the trilinear interpolation based on an inverse distance weighted average [166]. We determine $k = 3$ neighbors of the points of stage S_{L-1} within the downsampled points of stage S_L based on the kNN algorithm. We process the features of stage S_L by a fully connected layer with weight matrix $\mathbf{W}^{\mathrm{up}}_{S_L} \in \mathbb{R}^{D_{S_L} \times D_{S_{L-1}}}$ where $D_{S_{L-1}} = D_{S_L}/2$ and layer normalization to adjust the dimensions of the features of the two stages. We interpolate the resulting features and add them to the features of the current stage S_{L-1}, which we additionally process by a fully connected layer with $\mathbf{W}^{\mathrm{up}}_{S_{L-1}} \in \mathbb{R}^{D_{S_{L-1}} \times D_{S_{L-1}}}$ and layer normalization. The upsampled features are processed within the succeeding stages to integrate the information iteratively. Subsequently, we add high-level features to the full-resolution point cloud of stage S_1 to derive the final output features of our backbone $\mathbf{X}^b \in \mathbb{R}^{N \times D_{S_1}}$. The advanced network architecture enables the extraction of valuable features for sparse radar point clouds to enhance performance.

7.1.3 Moving Instance Transformer Head

Our moving instance transformer head combines moving object segmentation with instance segmentation to derive the final panoptic output, as detailed in Figure 7.2. For moving instance segmentation, currently moving agents, such as cars, bikes, and trucks, belong to the same moving class without further differentiation.

For many tasks, the information of being a moving object in combination with the velocities is often sufficient, simplifying the labeling efforts. However, state-of-the-art approaches for panoptic segmentation and instance segmentation rely on specific semantic labels to enhance accuracy [32, 59, 118, 214]. To overcome current limitations and further incorporate instance information, we propose a new attention-based module to reliably identify moving instances in sparse and noisy radar point clouds.

First, we determine the prediction for per-point moving object segmentation $\mathcal{P}^{\text{MOS}} = \{p_1^{\text{MOS}}, \dots, p_N^{\text{MOS}}\}$, where $p_i^{\text{MOS}} \in \{\text{static}, \text{moving}\}$ using an MLP, as visualized in Figure 7.4. To identify points belonging to the same instances, our idea is to deduce an attentive similarity quantity $\mathbf{S}_{i,j} \in \mathbb{R}$ based on the self-attention mechanism. Therefore, we process the output features of the backbone $\mathbf{X}^b$ by two fully connected layers to encode the features as keys $\mathbf{K}^b$ and queries $\mathbf{Q}^b$, with $\mathbf{W}_Q^b \in \mathbb{R}^{D_{S_1} \times D_{S_1}}$ and $\mathbf{W}_K^b \in \mathbb{R}^{D_{S_1} \times D_{S_1}}$, as introduced in Chapter 5. Following our previous approaches, we determine the local neighborhoods by kNN with $k = N_a$ and aggregate the information resulting in queries $\mathbf{Q}^{b*}$ and keys $\mathbf{K}^{b*} \in \mathbb{R}^{N \times N_a \times S_1}$. Additionally, we calculate the relative positions within the resulting local areas. However, we reduce the dimensionality of the relative positional encoding $\mathbf{R}^b$ by a fully connected layer with weight matrix $\mathbf{W}_R^b \in \mathbb{R}^{3 \times 1}$ and rectified linear unit activation function [150]. To condense the important information within a scalar value for the local similarity quantity, we calculate the dot-product attention $\mathbf{A}^{\text{loc}} \in \mathbb{R}^{N \times N_a}$ for the keys and queries, resulting in:

$$\mathbf{A}_{i,j}^{\text{loc}} = \mathbf{Q}_{i,j}^{b*} \mathbf{K}_{i,j}^{b* \top} + \mathbf{R}_{i,j}^{b}. \tag{7.2}$$

The goal is to have local similarity quantities close to 1 for all points within the local area that belong to the same instance and values close to 0 for different instances. Therefore, we replace the softmax function with the sigmoid function to determine the final local similarity quantity $\mathbf{S}^{\text{loc}}$ as:

$$\mathbf{S}_{i,j}^{\text{loc}} = \text{sigmoid}(\mathbf{A}_{i,j}^{\text{loc}}) = \frac{1}{1 + \exp(-\mathbf{A}_{i,j}^{\text{loc}})}. \tag{7.3}$$

The resulting local similarity matrix $\mathbf{S}^{\text{loc}} \in \mathbb{R}^{N \times N_a}$ includes a similarity quantity for all N points of the current point cloud. The ground truth of the local similarity matrix can be directly derived from the indices of the kNN algorithm and the ground truth instance IDs. Since the focus is to extract valuable information for moving instances, we set the similarity quantity for two points within the local area that both belong to the static class to 0, which we elaborate in more detail in Section 7.3.5.

The restriction to local areas induces the problem that often not all points that belong to one instance have an attentive similarity quantity. To overcome this limitation, we introduce the global attentive similarity quantity $\mathbf{S}^{\mathrm{glob}}$ for moving detections. Hence, the second part of our instance transformer head combines the prediction of the moving object segmentation and our local similarity quantity. Since computing the global similarity, including all points, comes with a large computational burden, we first select the N^{mov} points that we predict as moving. In contrast to Cheng et al. [36], we select all moving predictions, not just those predicted within the same mask, which corresponds to the foreground region for each query. We already encoded the features as keys $\mathbf{K}^b$ and queries $\mathbf{Q}^b$ for the local attentive similarity such that no additional encoding is required. We sample and group the global moving predictions for the queries and keys, resulting in $\mathbf{Q}^{b**}, \mathbf{K}^{b**} \in \mathbb{R}^{N^{\mathrm{mov}} \times N^{\mathrm{mov}} \times S_1}$. We use the dot-product to derive the global attention weights $\mathbf{A}^{\mathrm{glob}} \in \mathbb{R}^{N^{\mathrm{mov}} \times N^{\mathrm{mov}}}$ as:

$$\mathbf{A}^{\mathrm{glob}}_{i,j} = \mathbf{Q}^{b**}_{i,j} \mathbf{K}^{b**\top}_{i,j}. \tag{7.4}$$

We remove the relative positional encoding because the values largely differ in magnitude compared to the local similarity and, at least in our experience detailed in Section 7.3.5, do not provide important information to differentiate instances in the global context. We follow Equation (7.3) to calculate the final global similarity quantity $\mathbf{S}^{\mathrm{glob}} \in \mathbb{R}^{N^{\mathrm{mov}} \times N^{\mathrm{mov}}}$. We construct the ground truth matrix for the global similarity matrix based on the moving object predictions.

The attentive similarity modules are differentiable and can be learned in an end-to-end manner. We emphasize that our attention-based similarity quantity encodes essential information for moving instance association and moving object segmentation in noisy radar point clouds. Incorporating global information supports the extraction of discriminative features, especially for sparse point cloud processing. The computation of the local and global similarity quantities is model-agnostic and can be combined with different backbones.

7.1.4 Graph-based Instance Assignment

The final part of our instance transformer head is the graph-based instance assignment module. It utilizes the predictions $\mathcal{P}^{\mathrm{MOS}}$ and the global similarity matrix $\mathbf{S}^{\mathrm{glob}}$, which includes a similarity quantity for each pair of moving predictions to derive the final instance IDs. The key idea is to overcome the limitations of the direct assignment since the global similarity matrix may include multiple instance assignments and different similarity quantities for different instances, which are difficult to solve. Furthermore, we want to remove the semantic class dependencies mentioned in Section 7.1.3 to derive the final instances.

We first construct a radius graph $G = (\mathcal{V}, \mathcal{E})$ based on the prediction of moving detections and the position information to take into account that sparse and noisy radar point clouds differ in density. The moving detections within the individual scans represent the vertices $\mathcal{V} = \{1, \dots, N^{\text{mov}}\}$ of the graph. We derive the set of m edges $\mathcal{E}$ by connecting each moving point with the neighboring points, which lie within the spatial radius r_g. The graph can be represented by the square adjacency matrix $\mathbf{A}^{\text{adj}} \in \mathbb{R}^{N^{\text{mov}} \times N^{\text{mov}}}$, where $\mathbf{A}^{\text{adj}}_{i,j} = 1$, if $||p_i - p_j|| \leq r_g$ and otherwise $\mathbf{A}^{\text{adj}}_{i,j} = 0$. To derive the instance IDs, we partition the graph into $G_1, \dots, G_{k'}$ disjoint subgraphs, where k' is not given in advance because the number of instances within a scan is unknown. Based on the constructed graph, our idea is that individual instances have highly interconnected nodes and points belonging to different instances are sparsely connected. To exploit this property, we partition the graph by maximizing the modularity via a spectral approach [151].

The modularity depends on the number of edges falling within clusters minus the approximated expected number of random edges in an equivalent network, including the same number of nodes. Each node has the same degree, but the edges are randomly attached. For the case of two clusters, $l_i = 1$ if vertex i belongs to the first cluster and $l_i = -1$ if it belongs to the second one. The modularity is:

$$Q = \frac{1}{4m} \sum_{i,j} \left(\mathbf{A}^{\text{adj}}_{i,j} - \frac{k_i k_j}{2m} \right) l_i l_j, \tag{7.5}$$

where k_i and k_j are the degrees, the number of edges incident to a vertex, of the corresponding vertices. The adjacency matrix $\mathbf{A}^{\text{adj}}_{i,j}$ contains information about the actual number of edges, and the factor $\frac{k_i k_j}{2m}$ is the approximated expected number of edges between vertices i and j if the edges are placed randomly. The modularity can be further expressed by the modularity matrix $B_{i,j} = \mathbf{A}^{\text{adj}}_{i,j} - \frac{k_i k_j}{2m}$ as:

$$Q = \frac{1}{4m} \mathbf{l}^\top \mathbf{B} \mathbf{l}, \tag{7.6}$$

where $\mathbf{l}$ is the vector of the elements l_i. To correctly divide the network into more than two clusters, we construct a modularity matrix $\mathbf{B}^{\text{sub}}$ for each subgraph resulting in:

$$\mathbf{B}^{\text{sub}}_{i,j} = \mathbf{A}^{\text{adj}}_{i,j} - \frac{k_i k_j}{2m} - \delta_{i,j} \left[k^{\text{sub}}_i - k_i \frac{d^{\text{sub}}}{2m} \right], \tag{7.7}$$

where k^{sub}_i is the degree of the node i within the subgraph, d^{sub} is the sum of all degrees k_i of the nodes in the specific subgraph, and $\delta_{i,j}$ is the Kronecker delta, which is 1 if both nodes belong to the same subgraph and zero otherwise. We calculate the subgraph modularity Q^{sub} following Equation (7.6) to determine the additional contribution to the total modularity Q.

The modularity matrix $\mathbf{B}$ always depends on the adjacency matrix $\mathbf{A}^{\text{adj}}$. Since the plain radius graph leads to interconnections of nearby instances, represented in $\mathbf{A}^{\text{adj}}$, we optimize the cluster assignment by including additional information about the vertices. Our global attentive similarity matrix $\mathbf{S}^{\text{glob}}$ directly provides this valuable information since the similarity quantities differentiate between instances. Hence, we calculate the element-wise product of both matrices as:

$$\mathbf{A}^{\text{adj}}_{\text{attn}} = \mathbf{S}^{\text{glob}} \odot \mathbf{A}^{\text{adj}}, \tag{7.8}$$

to resolve the issue and derive the final attention-based adjacency matrix $\mathbf{A}^{\text{adj}}_{\text{attn}}$ for the instance assignment. The modularity, which now incorporates the additional instance information in Equation (7.7), is optimized by spectral modularity maximization. The optimization stops if any proposed split for all the subgraphs has a negative or no effect on the modularity. The final partitions represent our instance IDs. The partitioning is class-independent and incorporates attention-based instance information.

7.2 Implementation Details

We implemented our Radar Instance Transformer in PyTorch [161] and train our network with one Nvidia A100 GPU and a batch size b of 64 over 100 epochs. We utilize the AdamW [132] optimizer with an initial learning rate of 0.001 and drop the learning rate by a factor of 10 after 60 and 80 epochs to train our model. We use the binary cross-entropy loss to learn the local $\mathbf{S}^{\text{loc}}$ and global $\mathbf{S}^{\text{glob}}$ similarity matrix for instance association. For the moving object segmentation output, we utilize the focal Tversky loss [1].

The individual scans, which are the input to our network, are sparse radar point clouds with N points and feature dimension D. The RadarScenes [189] dataset comprises two spatial coordinates and the Doppler velocity, whereas the View-of-Delft dataset [158] also includes the elevation information. Each sequence consists of $T + 1$ scans, and the batch size b is the number of input sequences. For RadarScenes, we extend the coordinates x_i^C, y_i^C with the coordinate $z_i^C = 0$ to apply the pose transformation mentioned in Section 7.1.1. In addition to the position information and the Doppler velocity v_i, the radar sensors provide the radar cross section σ_i, resulting in input vector $\mathbf{x}_i = \left[x_i^C, y_i^C, z_i^C, \sigma_i, v_i\right]^\top$.

We first increase the per-point features to $D_1 = 16$ before we apply the intra-attention to enrich the current point cloud with temporal information. We increase the dimension in our sequential attentive feature encoding module and set $D_2 = 32$ resulting in an output feature dimension of 48, which is kept for the full-resolution stage $D_{S_1} = 48$ and gradually increased to $D_{S_2} = 96$, $D_{S_3} = 192$, and $D_{S_4} = 384$ during downsampling. We set $N_{\text{rit}} = N_d = N_a = 12$, where N_{rit} represents the local area in the transformer layers, N_d the local neighborhood for downsampling, and N_a the local area for the attentive local similarity quantity.

The input sequence consists of the current point cloud and $T = 2$ previous scans, which we further elaborate in Section 7.3.4. We pad missing previous scans as zeros to always include $T+1$ scans as input. The padded scans comprise a set of $1,024$ points where we set the features to zero. The padding is adopted until $T+1$ scans are available. To reduce overfitting during training, we apply data augmentation [245], including Gaussian jittering (variance of 0.01), scaling ($\times 0.95$, $\times 1.05$), shifting ($\pm 0.1\,\mathrm{m}$), and random flipping of the point cloud.

7.3 Experimental Evaluation

The main focus of this chapter is to achieve reliable moving instance segmentation in sparse and noisy radar point clouds. We present our experiments to show the capabilities of our method and to support our key claims, including that our approach achieves state-of-the-art performance in moving instance segmentation without processing multiple scans throughout the whole network. Our sequential attentive feature encoding module improves prediction performance by efficiently incorporating temporal information. In addition, the attentive instance transformer head enhances the overall performance of moving instance segmentation by including local and global instance knowledge. Our attention-based graph clustering enhances semantic class-independent instance assignments. The optimized backbone architecture extracts discriminable features from sparse radar point clouds by enriching individual detection with high-level features to improve performance.

7.3.1 Experimental Setup

As in Chapter 5, we perform our evaluation on the RadarScenes dataset. We use the same training, validation, and test split to ensure comparability. Additionally, we evaluate our method on the medium-sized View-of-Delft dataset [158] to illustrate the generalization capabilities of our method. Furthermore, we assemble the annotations of the RadarScenes dataset to establish the new task of moving instance segmentation, which we introduce in Section 7.3.2. For the View-of-Delft dataset, we keep the original training and validation split. We transfer the bounding box labels into point-wise annotations, including the differentiation between moving and static instances. We utilize the panoptic quality (PQ) [14] to evaluate the combined task of semantic segmentation and instance segmentation. Additionally, we report the segmentation quality (SQ), the recognition quality (RQ), and the intersection over union (IoU). To derive a detailed evaluation, we further differentiate between the static and moving classes.

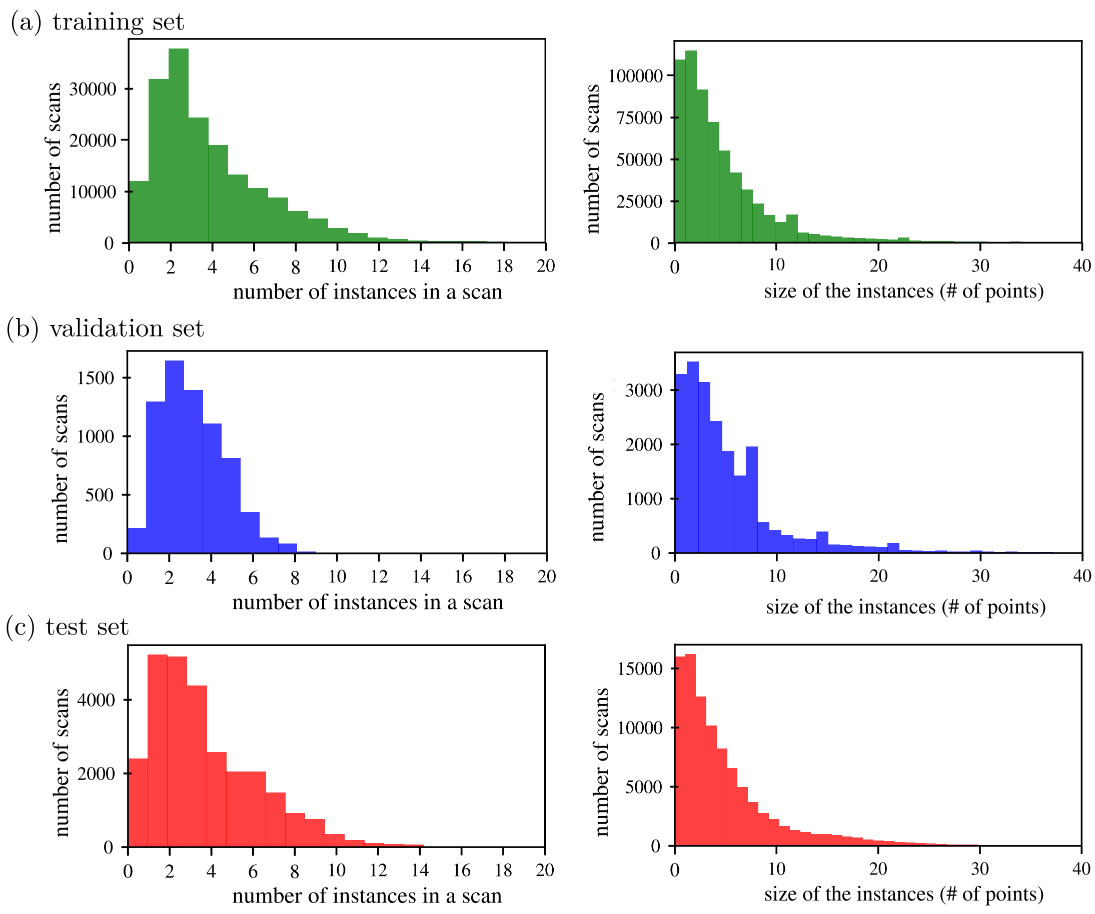


Figure 7.5: Statistics of the moving instance segmentation benchmark based on the RadarScenes dataset. The statistics are conducted individually for the training (a), the validation (b), and the test set (c). The left column illustrates the number of moving instances within a scan, and the right column shows the size of the moving instances, which correspond to the number of points.

7.3.2 Moving Instance Segmentation Benchmark

We utilize the dataset split introduced in Section 5.3.5 to establish the moving instance segmentation benchmark. The semantic annotations for the moving agents comprise the car, large vehicle, two-wheeler, pedestrian, and pedestrian group class in the original dataset. We transfer the semantic classes into moving and static annotations. Additionally, we transfer the track IDs into instance labels for the individual scans. The resulting annotations include an instance ID and a semantic class for each point. To illustrate the diversity of the dataset, we show in Figure 7.5 histograms of the number of instances within a scan and the size of the instances, which is equal to the number of points per instance. The number of moving instances within a scan displays the diversity of scenes with changing numbers of agents.

model	PQ	mIoU	SQ	RQ	PQ^{mov}	IoU^{mov}	SQ^{mov}	RQ^{mov}	PQ^{stat}	IoU^{stat}	SQ^{stat}	RQ^{stat}
Threshold $\lvert v_i \rvert > v_t$ + HDBSCAN [27]	59.7	65.2	88.3	64.6	23.7	35.1	80.9	29.3	95.7	95.2	95.7	100.0
Threshold $\lvert v_i \rvert > v_t$ + mean shift [42]	57.2	65.2	90.4	61.0	18.7	35.1	85.1	22.0	95.7	95.2	95.7	100.0
4DMOS [140] + HDBSCAN [27]	71.3	86.0	93.1	75.0	43.7	73.1	87.3	50.1	98.9	99.0	98.9	100.0
4DMOS [140] + mean shift [42]	79.0	86.0	95.9	81.9	59.1	73.1	92.7	63.7	98.9	99.0	98.9	100.0
Point Voxel Transformer [262] + HDBSCAN [27]	70.3	84.7	91.8	74.6	41.9	70.7	84.9	49.3	98.7	98.8	98.7	100.0
Point Voxel Transformer [262] + mean shift [42]	76.3	84.7	94.6	79.4	53.9	70.7	90.7	59.9	98.7	98.8	98.7	100.0
Stratified Transformer [242] + HDBSCAN [27]	71.2	87.3	92.1	75.5	43.5	75.2	85.2	51.1	98.9	99.0	98.9	100.0
Stratified Transformer [242] + mean shift [42]	78.4	87.3	94.8	81.7	57.9	75.2	90.8	63.7	98.9	99.0	98.9	100.0
Gaussian Radar Transformer (Chapter 5) + HDBSCAN [27]	73.2	89.1	92.9	77.3	47.3	79.1	86.8	54.5	99.0	99.1	99.0	100.0
Gaussian Radar Transformer (Chapter 5) + mean shift [42]	81.1	89.1	95.9	84.1	63.3	79.1	92.7	68.2	99.0	99.1	99.0	100.0
DS-Net [82]	62.0	82.2	94.2	64.4	25.9	66.1	90.3	28.7	98.1	98.2	98.1	100.0
Mask3D [187]	80.4	84.3	96.4	83.0	62.0	69.8	94.0	66.0	98.8	98.9	98.8	100.0
Ours	**86.5**	**92.6**	**96.9**	**89.0**	**73.6**	**85.7**	**94.4**	**78.0**	**99.4**	**99.4**	**99.4**	**100.0**

Table 7.1: Moving instance segmentation results on the RadarScenes test set in terms of PQ, SQ, RQ, and IoU scores. Our Radar Instance Transformer outperforms the existing methods, especially in terms of PQ and IoU. The best results are in bold.

7.3.3 Moving Instance Segmentation

The first experiment evaluates the performance of our approach, and its outcome supports the claim that we achieve state-of-the-art results for moving instance segmentation in sparse and noisy radar point clouds. We compare our Radar Instance Transformer to recent and high-performing networks with a strong performance on different benchmarks, including 4DMOS [140] by Mersch et al. for moving object, Stratified Transformer [242] by Xin et al., our Gaussian Radar Transformer introduced in Chapter 5, and Point Voxel Transformer [262] by Zhang et al. for semantic, Mask3D [187] by Schult et al. for instance, and DS-Net [82] by Hong et al. for panoptic segmentation. We select the Gaussian Radar Transformer introduced in Chapter 5 as a baseline because the model includes more parameters to learn additional features for instance segmentation. This provides an advantage over the Radar Velocity Transformer discussed in the previous chapter. We added the two approaches [82, 187] based on the fact that the clustering has to be class-agnostic because all instances belong to the moving class. Additionally, the methods do not include range image representations since RadarScenes does only provide x and y coordinates. To derive panoptic labels, we extend Mask3D [187] with state-of-the-art post-processing for mask predictions [36], which applies a threshold on the mask predictions to derive the instance IDs. We extend the semantic and moving object segmentation approaches [140, 242, 262] with commonly used clustering algorithms, namely HDBSCAN [27] and mean shift [42], to group points into instances.

Furthermore, we add a threshold-based baseline that first identifies moving detections based on the ego-motion compensated Doppler velocities where threshold $v_t = 0.92\,\mathrm{m/s}$ and utilizes the clustering algorithms to group the detected point into moving instances. To reliably detect the most vulnerable road users, the pedestrians, we have to identify instances that comprise just a single detection. Therefore, we optimize the hyperparameters and set the minimum cluster size for HDBSCAN to 1. For mean shift, we optimize the bandwidth b^{ms} based on the overall performance on the validation set, resulting in 3.5, 4.5, 5.0, and 4.5 for 4DMOS, Stratified Transformer, Point Voxel Transformer, and Gaussian Radar Transformer, respectively. To reliably detect larger objects, we additionally implement an offset prediction head [83] to support the clustering and enhance accuracy for point-based methods.

The offset prediction head combines two fully connected layers, batch normalization [89], and a rectified linear unit [150] to regress offset to the instance centers. We concatenate the features of the respective backbone $\mathbf{X}^b$ and the coordinates $\mathbf{P}$ to include fine-grained position information. The offsets $\mathbf{O} \in \mathbb{R}^{N\times 3}$ point from the point coordinates $\mathbf{P} \in \mathbb{R}^{N\times 3}$ to the instance centers $\mathbf{C} \in \mathbb{R}^{N\times 3}$. We derive the center based on the ground truth instance IDs and point coordinates.

model	PQ$^{\text{mov}}$	SQ$^{\text{mov}}$	RQ$^{\text{mov}}$	IoU$^{\text{mov}}$
Stratified Transformer [242]	22.8	78.1	29.3	55.6
Gaussian Radar Transformer (Chapter 5)	26.0	**79.4**	32.7	62.3
Ours	**26.9**	79.1	**34.0**	**63.3**

Table 7.2: Moving instance segmentation results on the View-of-Delft [158] validation set in terms of PQ$^{\text{mov}}$, SQ$^{\text{mov}}$, RQ$^{\text{mov}}$, and IoU$^{\text{mov}}$ scores. Our Radar Instance Transformer outperforms the existing methods in terms of PQ and IoU. The best results are in bold.

The loss function for offset regression can be expressed as follows:

$$\mathcal{L}^{\text{off}} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{o}_i - (\mathbf{c}_i - \mathbf{p}_i)\|_1, \tag{7.9}$$

where N is the number of points in the current point cloud.

Our Radar Instance Transformer outperforms the existing methods, especially in terms of PQ and IoU for the moving instances, as displayed in Table 7.1. The velocity threshold-based baseline further illustrates the difficulties of the task and underlines the necessity of learning-based approaches. The major problem is the noise, which leads to false positive predictions of the moving class, as mentioned in the previous section. The learning-based approaches improve IoU$^{\text{mov}}$ by more than 30 absolute percentage points. DS-Net utilizes the optimized dynamic shifting module for instance clustering to handle complex point distributions. However, point-based approaches [242, 262] clearly exceed the voxel-based method, which underlines that minimizing discretization artifacts enhances performance. Point Voxel Transformer includes a point-based branch, which helps to reduce the negative effect. On the contrary, 4DMOS [140] enhances performance by aggregating scans and prediction smoothing over time. However, the instance assignment is difficult. We assume that the problems result from the fact that all instances belong to the same class, which makes offset prediction and clustering more complex due to the missing differentiation of the semantic class. Nevertheless, algorithms that do not depend on semantic labels are more generally applicable and potentially cover the long-tail classes.

The dedicated mask predictions of Mask3D [187] overcome this limitation and improve PQ. However, our Radar Instance Transformer enhances PQ$^{\text{mov}}$ by 10 absolute percentage points compared to Mask3D, which is a strong improvement. The Stratified Transformer performs well without temporal information. Particularly, the mean shift algorithm performs well since the restriction of a cluster size of one for HDBSCAN makes it difficult to cluster larger objects reliably. However, even in the semantic segmentation, our Radar Instance Transformer enhances the IoU$^{\text{mov}}$ by more than 9 absolute percentage points, which again

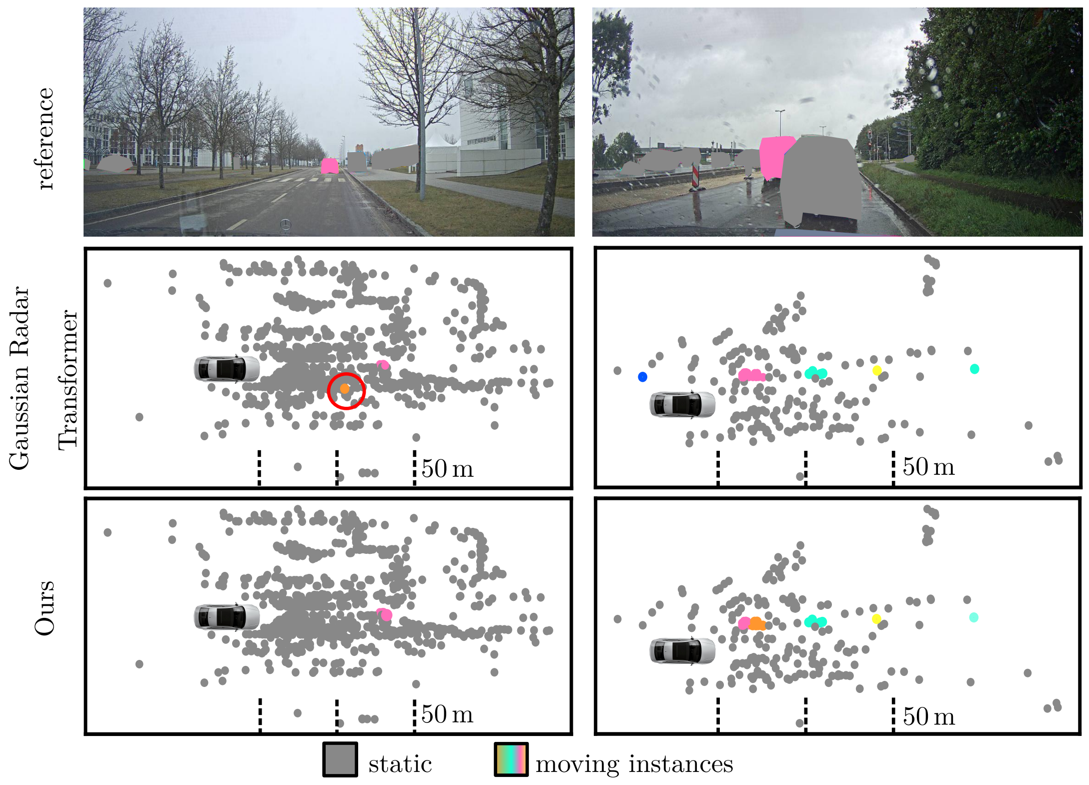
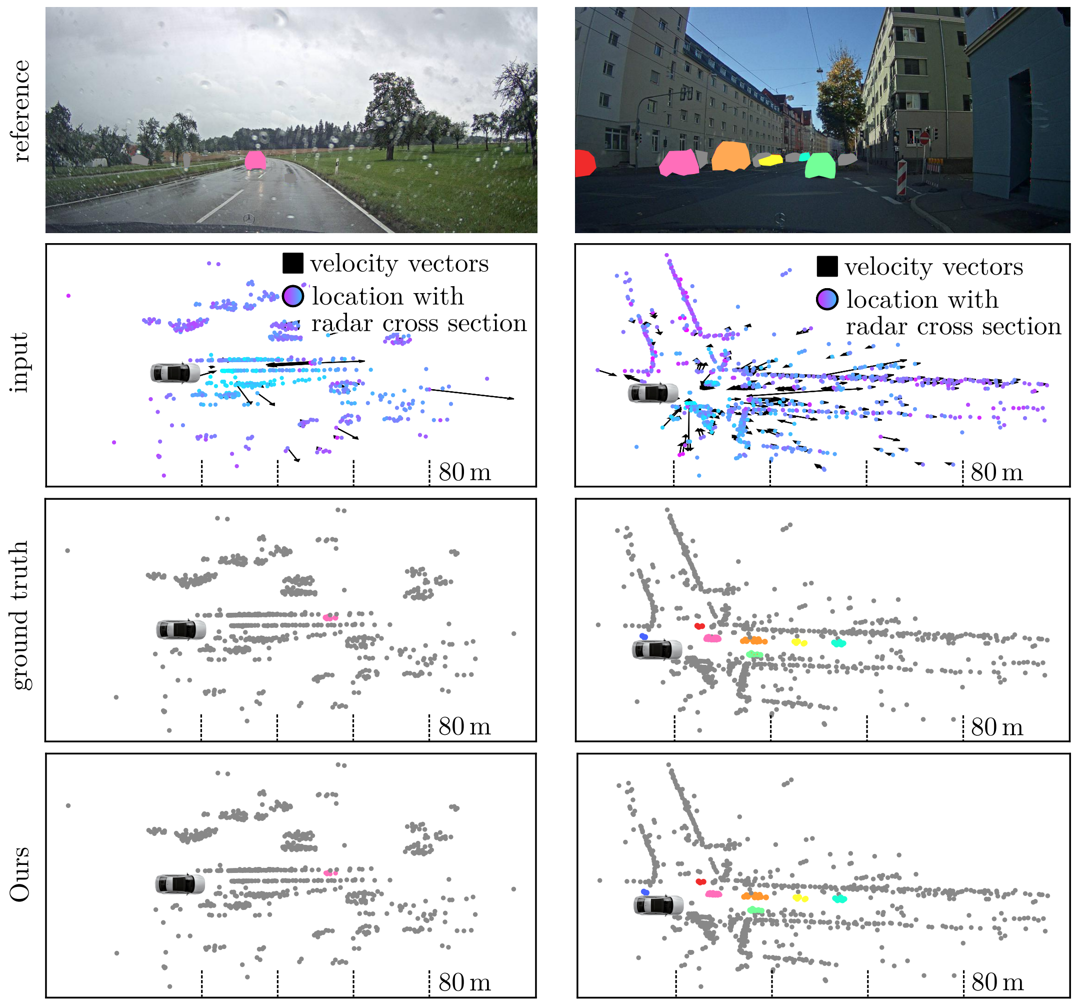


Figure 7.6: Qualitative results of our Radar Instance Transformer and our Gaussian Radar Transformer on the test set of RadarScenes. The camera images are anonymized and shown for reference. The left column is from sequence 14 (fog), and the right is from sequence 93 (rain). In the images of the predictions, each color represents a different instance of moving objects. The colors in the images correspond if the object is visible.

illustrates the superiority of our method. Moreover, we even surpass our other radar-specific method, the Gaussian Radar Transformer, by more than 5 absolute percentage points in terms of $\mathrm{IoU}^{\mathrm{mov}}$.

To further illustrate the generalization capability of our model, we evaluate the three best-performing methods in terms of $\mathrm{IoU}^{\mathrm{mov}}$ on the View-of-Delft validation set. Table 7.2 shows the superior performance of our method compared to our Gaussian Radar Transformer and Stratified Transformer. To enable a fair comparison, we utilize the IoU since both baselines are developed for semantic segmentation. The Stratified Transformer achieves an $\mathrm{IoU}^{\mathrm{stat}}$ of 98.1, whereas our methods achieve an $\mathrm{IoU}^{\mathrm{stat}}$ of 98.4. Nevertheless, our approach enhances the $\mathrm{IoU}^{\mathrm{mov}}$ by 1 absolute percentage point compared to the best-performing radar-specific method and by more than 7 absolute percentage points compared to Stratified Transformer. We observe that the relative transformations to align the point clouds are more precise in the RadarScenes dataset. We assume that this might be one reason why the improvements are more limited, as detailed in Section 7.3.4. Overall, the approaches perform well in both moving instance



(a) rural road (scene:79) (b) urban environement (scene:138)

Figure 7.7: Qualitative results of our Radar Instance Transformer on the RadarScenes test set. The reference camera images are anonymized. The illustration of the instance predictions and the ground truth instances have the same color. The colors in the image correspond if the object is visible. The results illustrate the performance within different scenes, including a rural road on a rainy day (a) and an urban environment with multiple agents (b).

segmentation benchmarks, which is a good starting point for future research.

Figure 7.6 shows some qualitative results on the RadarScenes test set of our approach and the Gaussian Radar Transformer. Our approach correctly identifies the distant moving instance in scene 14 and does not include a false positive prediction. Furthermore, our Radar Instance Transformer works reliably under versatile scenes, as illustrated in Figure 7.7. Our approach is able to identify moving instances under different and changing scenarios. The urban environment illustrates the difficulties of reliably identifying occluded distant moving instances. Additionally, the Doppler velocity values of instances that pass the ego vehicle are very small due to the fact that the Doppler velocity is the radial velocity of the detections, and the tangential movement is not covered. However, the

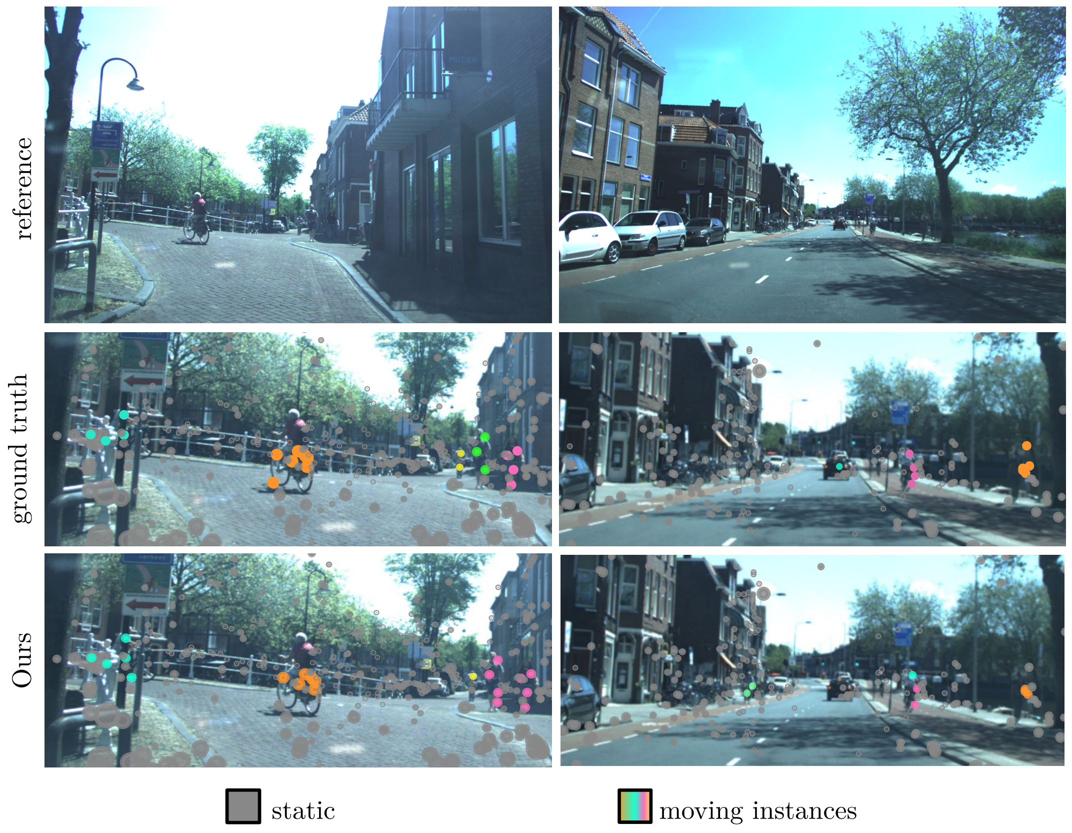


Figure 7.8: Qualitative results of our Radar Instance Transformer on the View-of-Delft validation set. The camera images are shown for reference. The ground truth instance labels and the corresponding instance predictions are projected into the reference image. The ground truth and the predictions are cropped for better visibility.

network accurately differentiates between moving and static agents. Furthermore, our approach identifies the oncoming car in rainy scene 79 precisely. The scene illustrates the handling of noise, which is best seen for the Doppler velocity vectors for off-road detections belonging to the static class, as depicted in the ground truth. The different scenarios further illustrate the sparsity and the changing density of the radar scan. Figure 7.8 illustrates further qualitative results on the View-of-Delft validation set of our approach. Overall, our approach performs well despite the various difficulties of sparse and noisy point clouds in real-world environments.

7.3.4 Ablation Studies on the Sequential Attentive Feature Encoding Module

The second experiment evaluates our sequential attentive feature encoding and illustrates that our module improves performance by efficiently including valuable temporal information within the features of the current point cloud. For this ex-

	pose compensation	PQ^{mov}	SQ^{mov}	RQ^{mov}	IoU^{mov}
$T = 0$	✓	73.8	94.7	77.9	82.2
$T = 1$	✓	74.0	94.8	78.0	82.5
$T = 3$	✓	76.4	95.0	80.4	83.6
$T = 4$	✓	77.3	95.2	81.2	**84.8**
$T = 5$	✓	**77.7**	95.2	**81.6**	84.5
Ours without pose compensation		76.1	**95.3**	79.8	83.3
Ours ($T = 2$)	✓	76.8	95.1	80.8	84.4

Table 7.3: Influence of the number of previous input point clouds and pose compensation on the RadarScenes validation set. The best trade-off for accuracy and runtime is achieved for two previous scans. The best results are in bold.

periment, we vary the number of input scans for our Radar Instance Transformer and evaluate the pose compensation, as depicted in Table 7.3. The pose compensation aligns the current and previous point clouds based on the ego-motion, as explained in Section 7.1.1. The temporal information improves the performance with the best result for five additional previous scans, i.e., $T = 5$. However, the training time, memory consumption, and runtime increase with adding more additional scans. Therefore, we select $T = 2$ as the best trade-off since our method performs well in terms of PQ^{mov} and IoU^{mov} and reduces the inference time. Furthermore, to save resources, the training time decreases by more than a factor of two compared to $T = 4$. The enhancement by temporally enriched features is directly visible by the increase of more than 2 absolute percentage points by including $T = 2$ previous scans instead of none. Furthermore, we argue that recent scans include the most important information to identify moving instances, leading to a slight decrease in accuracy for $T = 3$ and minor improvements for $T = 4$ and $T = 5$. Additionally, the pose compensation does not compensate for the relative movement of instances, leading to false associations over time. Thus, the recent scans contain the most valuable information, and the IoU^{mov} shows just small improvements. To verify that the temporal information supports the identification of static points, including noise, we determine the IoU^{stat} for $T = 0$ and $T = 2$ resulting in 99.4 and 99.5, respectively. We assume that the changing appearance of noise can be identified within the temporal domain, and hence, the IoU^{stat} improves. To elaborate on the improvement in detail, separate labels for static and noise are required, which are not available.

In the second step, we remove the pose compensation, which helps to align the previous and the current scan. Here, the decrease in PQ^{mov} is small. One advantage of radar sensors is the often higher frame rate compared to LiDAR sensors,

#	local similarity ($\mathbf{S}^{\text{loc}}$)	global similarity ($\mathbf{S}^{\text{glob}}$)	IoU$^{\text{mov}}$	PQ$^{\text{mov}}$
[A]			81.6	-
[B]	✓		83.7	-
[C]		✓	83.2	75.6
[D]	✓	✓	**84.4**	**76.8**
global: with positional encoding			82.6	74.2
local: static included			82.7	74.0

Table 7.4: Influence of the different components of our instance transformer head in terms of IoU$^{\text{mov}}$ and PQ$^{\text{mov}}$ on the RadarScenes validation set. The local $\mathbf{S}^{\text{loc}}$ and global similarity improve the predictions of moving instances. The best results are in bold.

which leads to smaller transformations between scans. In addition, we expect that the self-attention mechanism adaptively controls the information exchange within the current and previous scans and helps to extract valuable features, which underlines the advantages of our sequential attentive feature encoding module.

7.3.5 Ablation Studies on the Instance Transformer Head

The third experiment is presented to illustrate that our instance transformer head incorporates important instance information and enhances the performance of moving instance segmentation. Notably, we do not report the PQ$^{\text{mov}}$ for the models without the global attentive similarity quantity because we cannot perform the partitioning based on the learned weights, and thus we compare the results for this experiment based on the IoU$^{\text{mov}}$. To assess the benefits of the proposed attentive similarity quantities, we remove the respective modules for the global and local computation, as depicted in Table 7.4. We first remove the local and global attentive similarity in configuration [A], which reduces the IoU$^{\text{mov}}$ by 2.8 absolute percentage points. In ablation [B], we add the local attention, which already improves the IoU$^{\text{mov}}$ compared to [A]. We assume that the knowledge of which point within the local neighborhood belongs to the same instance helps to differentiate between instances and also to identify moving detections reliably. Compared to [A], the added global attentive similarity in [C] improves the performance. The global context is beneficial for identifying moving instances. Furthermore, false positive detections of moving detection are penalized, which enhances moving object segmentation. The combined local and global attentive similarity [D] combines both properties and further improves PQ$^{\text{mov}}$.

To analyze our design decisions, we add the positional encoding for the global attentive similarity. Since the relative position grows large in magnitude in the

	PQ^{mov}	SQ^{mov}	RQ^{mov}
r_g=3.0 m	70.4	94.5	74.5
r_g=4.0 m	73.4	95.3	77.0
r_g=5.0 m	75.1	**95.4**	78.7
r_g=6.5 m	76.6	95.1	80.6
r_g=7.5 m	**76.8**	95.1	**80.8**
r_g=8.0 m	76.7	95.0	**80.8**
Ours (r_g=7.0 m)	**76.8**	95.1	**80.8**

Table 7.5: Ablation study on the graph-based instance assignment on the RadarScenes validation set. The best performance is achieved for r_g=7.0 m. The best results are in bold.

global context, as mentioned in Section 7.1.3, we assume that the local context is lost, which is essential to differentiate between nearby points and hence does not improve accuracy. Additionally, we exclude the static point from the local similarity in our method, which means that for all static points, the predicted attentive similarity quantity has to be 0. We presume that the focus of the instance assignment is clearly on moving instances, which is not the case if we include the static points and thus decrease PQ^{mov}. In conclusion, the instance transformer head improves performance by exploiting local and global context to extract valuable features from noisy radar point clouds.

7.3.6 Ablation Studies on the Instance Assignment

The ablation study presented in this section supports our claim that attention-based graph partitioning enhances instance assignments without class-dependent information. We evaluate the hyperparameter r_g for the radius graph, as depicted in Table 7.5. Our method achieves good performance for different radii with the best PQ^{mov} for $r_g = 7\,\text{m}$. Compared with the bandwidths for mean shift algorithms in Section 7.3.3, our approach is able to utilize a larger radius. One reason for a larger radius is that we do not use an offset prediction and still reliably detect large moving instances. This could lead to a combination of instances, especially small instances with nearby larger instances. However, the major advantage of our attention-based graph partitioning is that the global attentive similarity quantity effectively removes the interconnection between these instances. We additionally evaluated the performance of the most vulnerable road users, pedestrians, which often comprise single detections to show that the interconnection can be reliably solved. However, potential future work might address the changing resolution of radar point clouds based on the distance to the sensor to refine the instance assignment. Within the evaluated range of radii,

model	IoU^{mov}	PQ^{mov}
Ours + Point Voxel Transformer [262]	80.0	69.0
Ours + Stratified Transformer [242]	82.2	70.1
Ours w/o top level	81.9	73.7
Ours	**84.4**	**76.8**

Table 7.6: Ablation study on the backbone design and comparison to recently best-performing backbones on the RadarScenes validation set. Our dedicated backbone design improves performance by full-resolution processing. The best results are in bold.

the PQ^{mov} for pedestrians remains stable at 84.7, which illustrates the robustness of our approach. In comparison, clustering often relies on accurate offset predictions and class-specific properties [157] to reliably identify instances of varying sizes. Hence, our attention-based graph partitioning helps to identify moving instances and overcomes the limitations of state-of-the-art approaches.

7.3.7 Ablation Studies on the Backbone Architecture

In our fourth experiment, we analyze our method with respect to the ability to extract valuable features by keeping the full resolution of the point cloud throughout the network. The results in Table 7.6 verify that removing the intermediate transformer blocks on the top levels of our model harms the accuracy. These building blocks are the additional modules compared to the widely-used U-Net [166, 271] architecture. Furthermore, we extend the two best-performing non-radar-specific backbones, which follow the U-Net architecture, with our model-agnostic modules to illustrate the advancements of our approach. The full-resolution processing and the extraction of additional high-level features outperform other methods, such as the Stratified Transformer [242]. Nevertheless, our model-agnostic sequential attentive feature encoding and instance transformer head enhance the performance of the Stratified Transformer [242] and the Point Voxel Transformer [262], which illustrates that these modules help to improve scene interpretation in radar scans.

7.3.8 Runtime

Finally, we analyze the runtime and the number of parameters of the different methods. We consistently follow our setup, introduced in Section 5.3.5, including the Nvidia RTX A6000 GPU and optimized farthest point sampling and kNN algorithm. For our approach, we utilize the best-performing configuration and include $T = 2$ previous scans.

model	parameters (M)	mean runtime (ms)
4DMOS [140]	1.8	14.0
Point Voxel Transformer [262]	6.2	47.1
Stratified Transformer [242]	8.0	34.4
DS-Net [82]	50.2	52.3
Mask3D [187]	39.6	85.2
Gaussian Radar Transformer (Chapter 5)	8.4	24.0
Ours	3.8	31.7

Table 7.7: The evaluation of the number of parameters and the mean runtime of the models on an Nvidia RTX A6000 GPU based on 1000 randomly sampled point clouds of the RadarScenes validation set. Our approach reduces the number of parameters compared to other approaches and runs faster than the frame rate of 17 Hz of the sensor.

Stages of our model	parameters (M)	mean runtime (ms)
Pre-processing	-	5.8
sequential attentive feature encoding	0.006	2.7
Backbone	3.8	15.7
Instance Transformer Head	0.02	1.2
Graph Partitioning	-	6.3

Table 7.8: Detailed evaluation of the number of parameters and the mean runtime of the individual building blocks of our model on an Nvidia RTX A6000 GPU based on 1000 randomly sampled point clouds of the RadarScenes validation set.

All results are detailed in Table 7.7. The mean runtime of our approach is 31.7 ms, which is equal to 31 Hz, and thus faster than the frame rate of 17 Hz of the sensor. Our Gaussian Radar Transformer has the lowest runtime of the transformer-based models in our experiments with 24 ms but utilizes twice as many parameters. We argue that the runtime depends on the different specific implementations of the respective attention mechanism. The Stratified Transformer [242] uses a more complex mechanism that induces further latencies compared to the simple vector attention of the Gaussian Radar Transformer. Additionally, we observe that the runtime scales with the number of points. To illustrate that aspect, we calculated the minimum and maximum runtime of our model, which are 23 ms and 221 ms, respectively. For all models, scans without any moving object prediction do not need clustering, which reduces runtime. However, with the increasing number of moving predictions, our global attentive association matrix gets more complex, and therefore, the runtime scales with the number of points and moving predictions. Furthermore, the runtime scales with the number of T previous scans, which increases to 35.3 ms for $T = 5$. Future research can address these issues to decrease runtime further. Another option to reduce the runtime is the replacement of the commonly used kNN algorithm. Since the resolution of radar point clouds is dependent on the distance to the sensor, the adjustment of the sampling might address this limitation and additionally improve performance.

To get an in-depth analysis, we additionally report the runtime and parameters of the individual blocks in Table 7.8. We observe that the processing of the intermediate building blocks of our backbone is time-consuming, and explicit parallelization of the stages can enhance the inference time. Nevertheless, we achieve state-of-the-art performance with an adequate runtime, outperforming all other methods except Gaussian Radar Transformer and DS-Net, while we include temporal information and use fewer parameters, reducing memory requirements.

In summary, our evaluation suggests that our method provides competitive moving instance segmentation in sparse radar point clouds by efficiently including temporal information. The attentive similarity quantities encode valuable information and enable class-agnostic, graph-based instance assignment, outperforming state-of-the-art approaches. Thus, we support all our claims through this experimental evaluation.

7.4 Conclusion

In this chapter, we addressed moving instance segmentation in sparse and noisy radar point clouds and presented a novel approach that shows strong performance and outperforms a large set of baseline methods. Our method efficiently exploits temporal information and overcomes the runtime limitations of passing aggregated scans through the whole network. We propose a full-resolution backbone to prevent information loss in sparse point cloud processing. We utilize the self-attention mechanism throughout the network to extract valuable features and introduce attentive graph-based instance partitioning. This allows us to successfully identify moving agents and enhance the feature extraction by incorporating local and global instance knowledge. We further established a new benchmark based on the RadarScenes dataset, which allows further comparisons with future work and provides comparisons to other methods. The experiments suggest that the different building blocks of our approach are essential to achieving good performance on moving instance segmentation. Furthermore, we propose model-agnostic modules to incorporate temporal information and perform instance clustering, which are applicable to different backbones to enhance scene understanding.

The strong performance within versatile driving scenarios and under challenging weather conditions illustrates the advantages of radar sensors, underlining the importance of these sensors. The provided Doppler velocity supports the identification of moving instances within single or temporally enriched scans, which is a fundamental advantage compared to other modalities. Additionally, the task of moving instance segmentation potentially solves the long-tailed class distribution problem by including rare and underrepresented classes within the moving class. The strong performance for moving object segmentation further shows that the approaches perform better for binary classification compared to semantic segmentation, introduced in Chapter 5, by reducing the class imbalance. Therefore, it makes sense to explore the advanced segmentation capabilities of moving instance segmentation and utilize the predictions to address additional tasks. Consequently, we introduce two approaches that leverage the moving instance predictions of the Radar Instance Transformer to enhance scene understanding in the following chapters.

Part II

Exploiting Moving Instance Segmentation to Enhance Scene Understanding

Chapter 8

Moving Instance Tracking

The identification of moving objects in dynamic real-world environments, as presented in the previous chapters, is the basis for safe autonomous navigation. Especially the segmentation into moving and static targets and the instance assignment to derive information about the agents in the scene are essential for collision-free path planning. The advantages of the radar sensor compared to other popular sensors are that it provides Doppler velocity information and radar cross section values that support accurate segmentation and lead to exceptional results. However, the prediction performed based on the current scan does not cover the temporal information, which, for example, humans are aware of, to predict future movement and anticipate possible maneuvers. Knowledge about the trajectory of traffic participants and their future state prediction enables advanced driving functionalities and is particularly useful in preventing collisions. Consequently, the temporally consistent tracking information about moving agents is crucial to enhancing scene understanding and avoiding dangerous situations.

In this chapter, we aim to leverage the predictions of the moving instance segmentation and extend the approach to track instances in sparse and noisy radar point clouds, as illustrated in Figure 8.1. This requires differentiating between moving and static parts of the surroundings and consistently distinguishing individual agents in the environment over time. The 4D moving instance segmentation task falls within 4D panoptic segmentation [7]. However, all moving instances belong to the moving object class without further differentiation into a more detailed separation. Consequently, there exists no additional semantic information about the instances that supports tracking of closely related instances, resulting in a challenging prediction task.

Current state-of-the-art methods [7, 35, 136] often address moving instance tracking within aggregated scans, and Weng et al. [223] associate instances and existing tracks based on the intersection over union score. However, the aggrega-

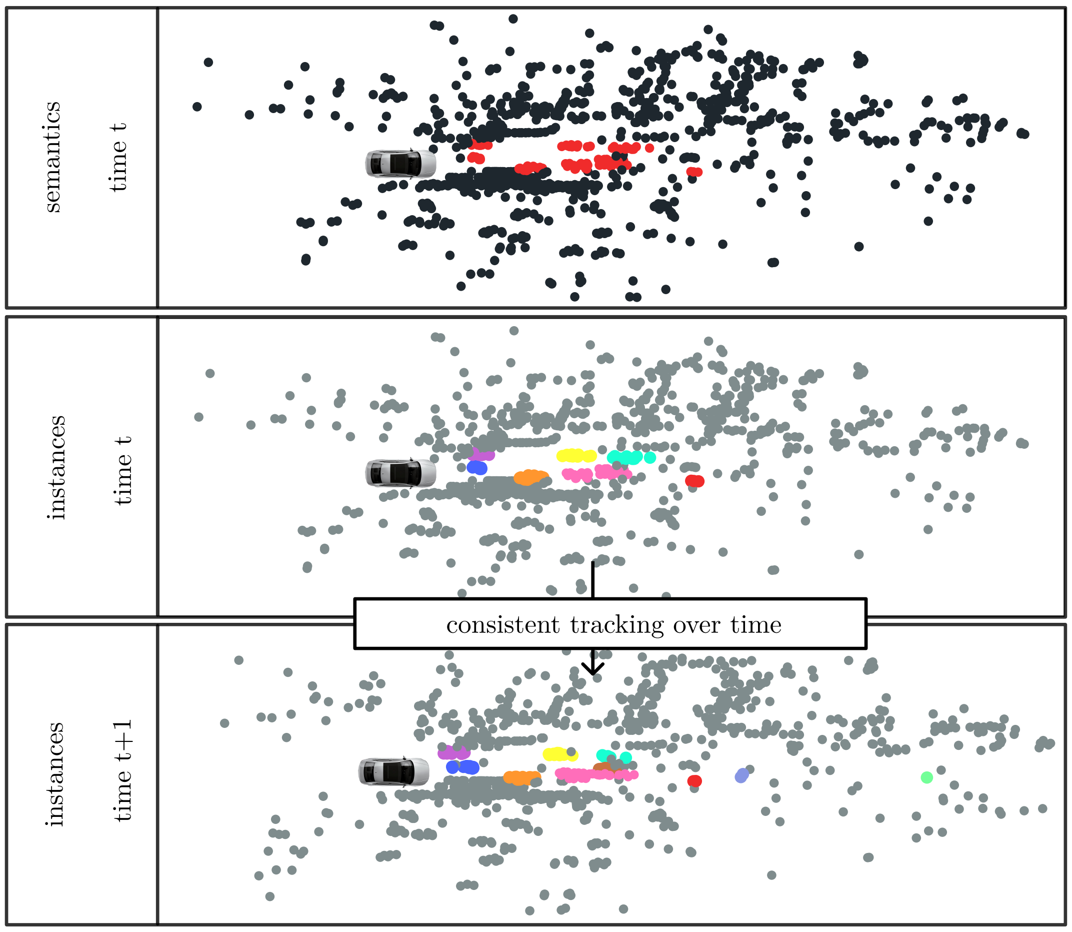


Figure 8.1: Our method combines moving object segmentation (top), instance segmentation (middle), and tracking (bottom) to solve the 4D panoptic task of moving instance tracking from sparse radar point clouds. The corresponding colors in the middle and bottom images represent the respective tracked instances (static is grey).

tion of scans induces latency and is disadvantageous for tasks requiring immediate feedback, such as collision avoidance. Additionally, instances within sparse radar point clouds often comprise single points for which an IoU-based association is inappropriate because no overlap of the bounding boxes exists. Other methods [16, 38] rely on dedicated trackers based on Kalman filters [91], which often neglect valuable appearance features [39, 223]. To extract appearance features, other approaches voxelize point clouds, which is particularly harmful to sparse radar data processing due to discretization artifacts. Furthermore, common approaches by Chen et al. [35] and Marcuzzi et al. [136] focus on the tracking of instances, including the semantic classes, which incorporate additional helpful information to derive correct associations. We go beyond that by tracking moving objects without the knowledge of the semantic class, which makes association more difficult. Furthermore, our approach overcomes the limitations of other methods by considering the tracking of instances that comprise single detections and reducing latency by performing association within consecutive scans.

The main contribution of this chapter is a novel point-based approach that enables moving instance tracking by incorporating geometric and appearance features to accurately associate moving instances in sparse and noisy radar point clouds over time. Our approach, called Radar Tracker, utilizes a neural network and extends the prediction of our moving instance segmentation approach to derive temporal consistent tracking IDs. We efficiently incorporate temporal information for each point by a temporal offset prediction to enhance the segmentation and enable direct center-based tracking of moving instances. We propose an attention-based instance feature extraction network to reduce information loss and keep the appearance features of the individual instances. Furthermore, we derive attention-based association scores to extend tracking by attention. The final geometric and appearance features are combined within our instance association to improve the performance of the overall estimation task.

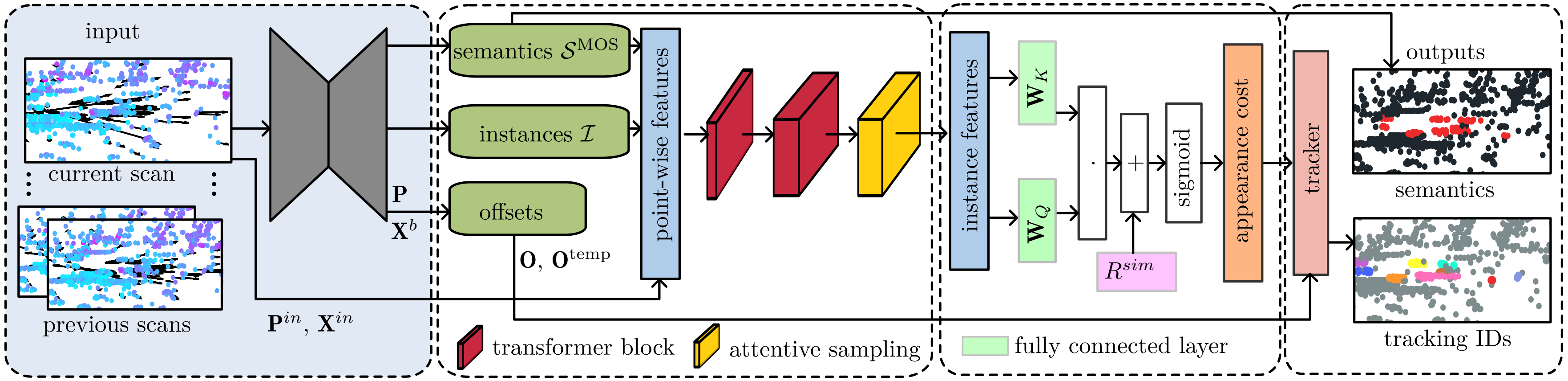
In sum, we make three claims in this chapter: First, our approach shows state-of-the-art performance for moving instance tracking in sparse and noisy radar point clouds. Second, our temporal offset prediction incorporates valuable information for segmentation, improving tracking performance. Third, our attention-based track association overcomes the shortcomings of state-of-the-art center-based tracking by incorporating appearance feature information.

8.1 Our Approach to Track Moving Instances

Our approach aims to achieve reliable moving instance tracking in sparse radar point clouds. We follow the tracking-by-detection paradigm and extend the Radar Instance Transformer with dedicated tracking modules, as illustrated in Figure 8.2. We directly predict the temporal offset for each detection within single radar scans to enhance segmentation and enable direct center-based association. We utilize the self-attention mechanism [212] to regress an additional cost function and include appearance features to improve tracking performance. The final association combines geometric and learned appearance features based on radar measurements to enhance scene understanding.

8.1.1 Moving Instance Segmentation Backbone

The performance of the instance segmentation backbone limits the tracking of objects. Therefore, we utilize the state-of-the-art Radar Instance Transformer introduced in Chapter 7 as the backbone to extract moving instances reliably. We follow the same procedure and utilize the current radar scan $\mathcal{P}^t$ at time t, which comprises the point coordinates and the radar features such as the Doppler



(a) backbone (b) attentive instance network (c) instance similarity module (d) data association

Figure 8.2: Detailed design of the individual modules of our Radar Tracker. (a) The backbone of the Radar Instance Transformer is extended with the offset predictions and provides the semantic classes and instance predictions. (b) The attentive instance network extracts features to represent instances. (c) Our instance similarity module determines the appearance-based association matrix to enhance instance tracking. (d) The data association utilizes the appearance and geometric features to predict the tracking IDs of moving instances in sparse radar point clouds.

velocity and radar cross section and add temporal information from T previous scans $\mathcal{P}^{t-T}$ by the sequential attentive feature encoding module. Since the temporal information improves moving instance segmentation, the enhanced performance directly influences the tracking accuracy. The processed single scan, including temporally enriched point-wise features, is then passed through the network to derive the outputs of the backbone, including the moving object segmentation labels $\mathcal{S}^{\text{MOS}}$, the instance IDs $\mathcal{I} = \{I_1, \dots, I_N\}$ with $I_i \in \mathbb{N}$, and the point-wise features $\mathbf{X}^b$. We utilize the moving instance predictions as input to our attentive instance network to incorporate the information on how many agents are present in the current scan and which points belong to the moving class. Since we do not rely on additional information, we can potentially substitute the backbone for other moving instance segmentation networks. Consequently, our Radar Tracker is model-agnostic and does not rely on the feature extraction capabilities of the backbone network to derive appearance features, which can be a limiting factor for the applicability of the approach.

8.1.2 Offset Prediction Module

Geometric features of instances are essential to track moving instances reliably. However, in sparse radar point clouds, the appearance of objects changes, making tracking based on bounding boxes difficult [223]. The case of single-point instances is especially not covered adequately. Therefore, our Radar Tracker focuses on center-based associations to exploit important geometric relationships.

Furthermore, future state prediction is crucial to associate tracks and objects over time. In contrast to other approaches [35, 252], which predict velocity vectors for bounding boxes or voxels, we directly process the point cloud on a per-point basis to include per-point motion cues without discretization artifacts. Our resulting approach first utilizes the commonly used offset prediction head [83] to regress offsets $\mathbf{O} \in \mathbb{R}^{N\times 2}$ to the instance center $\mathbf{C}^t \in \mathbb{R}^{N\times 2}$ of the current scan. Secondly, we predict the temporal offset $\mathbf{O}^{\text{temp}} \in \mathbb{R}^{N\times 2}$ for each point, which is a vector that points to the center of the instance $\mathbf{C}^{t+1} \in \mathbb{R}^{N\times 2}$ in the next scan. We calculate the center as the average of the coordinates of the points belonging to the instance.

The input to our offset prediction module is the concatenation of the features of the backbone $\mathbf{X}^b$, and the point coordinates $\mathbf{P}$ of the current scan to include fine-grained position information. Specifically, the offset prediction module within the current scan is often included in the backbone architecture [83] to cluster instances. However, the Radar Instance Transformer does not include offset predictions due to the advanced attention-based instance clustering, which is why offset predictions are additionally added as output to the backbone. For the individual offset prediction heads, we combine two fully connected layers, batch normalization [88] and a rectified linear unit [150]. The offset modules are model-agnostic and can be applied to different methods to include additional motion cues and enhance segmentation performance. The resulting offsets directly incorporate the information for center-based moving instance tracking and add motion cues about moving instances within each scan. Additionally, the temporal offset includes a regression target for single-point moving instances, which does not account for the spatial offset prediction because the single point already corresponds to the center. The effect is negligible for commonly used LiDAR data because instances normally comprise more than a single point. However, if we have a closer look at the distribution of radar points within an instance, introduced in Section 7.3.2, the problem of having a single detection for a complete instance is much more severe in radar data. Therefore, the additional motion cue for moving instances potentially improves segmentation performance in a multi-task manner, which we verify in Section 8.3.3.

8.1.3 Attentive Instance Network

Center-based instance association works remarkably well. However, the geometric association often neglects appearance features extracted by the network dependent on the radar features, which are essential if the geometric features are inaccurate or multiple agents interact. This makes it difficult for a purely geometrically based approach to perform well, especially for the targeted task of moving instance tracking where additional information on the semantic class of the instance is not present. In contrast to other methods, such as proposed by Chen et al. [35] and Marcuzzi et al. [136], we propose to extract discriminative instance features by a transformer-based network to reduce information loss by discretization artifacts in sparse radar data.

Our attentive instance network comprises two transformer blocks and an attentive aggregation module as depicted in Figure 8.2 (b). In contrast to the previous chapters, we only consider moving predictions as input to our network. Therefore, the input consists of the point coordinates $\mathbf{P}^{\text{in}} = [\mathbf{p}_1, \dots, \mathbf{p}_{N^{\text{mov}}}]^\top \in \mathbb{R}^{N^{\text{mov}} \times 2}$ and the features $\mathbf{X}^{\text{in}} = [\mathbf{x}_1, \dots, \mathbf{x}_{N^{\text{mov}}}]^\top \in \mathbb{R}^{N^{\text{mov}} \times D}$, where $\mathbf{p}_i \in \mathbb{R}^2$ and $\mathbf{x}_i \in \mathbb{R}^D$ for N^{mov} moving points. Hence, $\mathbf{X}^{\text{in}}$ only includes a subset of the features that are utilized by the backbone, which includes moving and static points. During training, we select the instances based on the ground truth labels and for the inference based on the semantic and instance predictions of the backbone. Therefore, our Radar Instance Transformer and the tracking modules are trained separately. We adopt the transformer design proposed in the previous chapter to extract point-wise information for the moving detection in the current scan. The transformer block is a residual block, including a feature dimension expansion that embeds a transformer layer. We first process the input features $\mathbf{X}^{\text{in}} \in \mathbb{R}^{N^{\text{mov}} \times D}$ by a fully connected layer with weight matrix $\mathbf{W}_l \in \mathbb{R}^{D \times D_1}$ to increase the feature dimension. The resulting features $\mathbf{X}_l^{\text{in}}$ and corresponding point coordinates $\mathbf{P}^{\text{in}}$ are fed into a transformer layer. The output features are processed by another fully connected layer and added to the skip connection features.

For the transformer layer, we follow Section 3.3 to calculate the attention weights. The local areas are determined with $k = N^k$ resulting in $\mathbf{Q}^k, \mathbf{K}^k$, and $\mathbf{V}^k \in \mathbb{R}^{N^{\text{mov}} \times N^k \times D_1}$ and $\mathbf{R} \in \mathbb{R}^{N^{\text{mov}} \times N^k \times D_1}$. We adopt vector attention and calculate the final attention weights $\mathbf{A}_{i,j}$ by the softmax function. To calculate the output features $\mathbf{X}^{\text{out}} \in \mathbb{R}^{N^{\text{mov}} \times D_1}$ of the transformer layer, we calculate the sum of the element-wise multiplication and add the relative positional encoding. We utilize two consecutive transformer blocks following the same processing steps, but we increase the feature dimension within the second transformer block to D_2. The resulting output features are $\hat{\mathbf{X}}^{\text{out}} \in \mathbb{R}^{N^{\text{mov}} \times D_2}$.

The second step of the network focuses on extracting one representation of the instances to achieve correct data association. The attentive aggregation module is inspired by the attention mechanism and follows the attentive sampling operation introduced in Chapter 5. We utilize the attentive aggregation module to keep and combine the information of instances within sparse radar point clouds. The idea is that the information of the individual detections is combined within one feature vector without information loss, which is in contrast to max pooling. We process the resulting output features $\hat{\mathbf{X}}^{\text{out}} \in \mathbb{R}^{N^{\text{mov}} \times D_2}$ of the second transformer block by a fully connected layer with weight matrix $\mathbf{W}^{\text{agg}} \in \mathbb{R}^{D_2 \times D_2}$ and softmax activation function. The resulting outputs are our aggregation weights $\mathbf{A}^{\text{agg}}$, which we utilize to weight the N^I points within the individual instance. The final instance feature is derived by the summation of the weighted point features, resulting in:

$$\mathbf{X}_i^{\text{inst}} = \sum_{j=1}^{N^I} \mathbf{A}_{i,j}^{\text{agg}} \odot \hat{\mathbf{X}}_{i,j}^{\text{out}}. \tag{8.1}$$

The weighting of the features of the detections combines the information about the instance and, therefore, omits information loss. We leverage the instance-wise feature vectors for the appearance-based data association. Additionally, we extract the coordinates $\mathbf{P}^{\text{inst}}$ of each point belonging to an instance to include position information in the association step. We argue that the spatial relationship is essential to prevent the association of distant objects that may have similar feature representations. The output of the network encodes the important information representing the individual instances. In the following, we want to exploit the instance features and coordinates to derive detailed associations based on similarities.

8.1.4 Instance Similarity Module

The essential part of improving the temporal data association of tracks and newly detected objects is the cost function or similarity measure for the tracking. We utilize the features and center coordinates of our attentive instance network to determine the similarities and incorporate appearance features. Additionally, we follow the previous chapters and utilize the attention mechanism throughout the network.

We encode the features $\mathbf{X}^{\text{inst}} \in \mathbb{R}^{N^{\text{inst}} \times D_2}$ as queries $\mathbf{Q}^{\text{sim}} \in \mathbb{R}^{N^{\text{inst}} \times N^{\text{inst}} \times D_2}$ and keys $\mathbf{K}^{\text{sim}} \in \mathbb{R}^{N^{\text{inst}} \times N^{\text{inst}} \times D_2}$ and perform dot product attention to derive an attention-based similarity value as visualized in Figure 8.2 (c). Within the attention matrix, we include the self- and cross-attention of the instances. Additionally, we calculate the relative center positions $\mathbf{r}_{\text{center}} = \mathbf{p}_i - \mathbf{p}_j$ where $\mathbf{p}_i$ and $\mathbf{p}_j \in \mathbf{P}_{\text{center}} = [\mathbf{p}_1, \ldots, \mathbf{p}_{N^{\text{inst}}}]^\top \in \mathbb{R}^{N^{\text{inst}} \times 2}$ for the number of moving instances N^{inst} and encode the relative positions by an MLP. We reduce the dimensionality of the encoding to one resulting in $\mathbf{R}^{\text{sim}} \in \mathbb{R}^{N^{\text{inst}} \times N^{\text{inst}}}$.

To calculate the resulting attention-based instance similarity scores, we replace the softmax function with an element-wise sigmoid function to derive values in the range of zero and one. The procedure is similar to that described in Section 7.1.3, but the attention weights now operate at the level of entire instances rather than distinguishing between individual points. Additionally, we add the positional center encoding as follows:

$$\mathbf{A}^{\text{sim}}_{i,j} = \text{sigmoid}\left(\mathbf{Q}^{\text{sim}}_{i,j} \mathbf{K}^{\text{sim}\top}_{i,j} + \mathbf{R}^{\text{sim}}_{i,j}\right). \tag{8.2}$$

The similarity scores incorporate the feature and position information and directly indicate how likely two instances belong together. The combination of both features combines the appearance of the instance within the scene to enhance the tracking performance. To utilize the scores as an additional cost function, we calculate the similarity cost as:

$$\mathbf{C}^{\text{sim}}_{i,j} = \frac{1}{(\mathbf{A}^{\text{sim}}_{i,j} + \epsilon)}, \tag{8.3}$$

where ϵ is an arbitrarily small constant for numerical stability. The similarity cost includes the appearance features and thus incorporates essential information for moving instance tracking.

8.1.5 Data Association

The central part of our data association is the offset predictions $\mathbf{O}^{\text{temp}}$ and $\mathbf{O}$ for the coordinates of the instance $\mathbf{P}^{\text{inst}}$ because the center-based association within a small distance is reliable for tracking moving instances. We add the respective offset to the corresponding point coordinates of the instance and derive the mean value of the predictions to calculate the center of the instance to directly associate instances. However, to improve data association when the geometric association is imprecise, we utilize the attention-based instance similarity scores $\mathbf{A}^{\text{sim}}$ to enhance tracking performance. Hence, the overall process includes appearance and geometric features to improve tracking quality.

We first calculate the Euclidean distance d based on the center predictions of our method. For the existing tracks, the center is defined by the temporal offset predictions $\mathbf{O}^{\text{temp}}$, whereas for the newly identified instances, the offset $\mathbf{O}$ is utilized. Since a global association within the complete field of view of the sensors includes multiple misleading connections, which would be considered in the optimization step, we directly restrict the optimization to local areas. Therefore, we cluster the instances based on the distances d into local areas with DBSCAN [53]. After the clustering, we utilize the local cost matrices, i.e. only instances in each cluster, and perform Hungarian matching [104].

Additionally, we process the input features $\mathbf{X}^{\text{in}}$ by our model to extract instance features $\mathbf{X}^{\text{inst}}$ and determine the similarity scores $\mathbf{A}^{\text{sim}}$ of the tracks and the objects. Since the geometric information is valuable within the short-range displacement, we determine a threshold t_{d1} where the data association is purely based on the geometric assignment for instances that move slowly and are not occluded during the tracking process. Since the measurement frequency of the radar sensors is high, the relative displacement for consecutive scans is small, which facilitates the geometric assignment. Above that threshold, we include the similarity cost function $\mathbf{C}^{\text{sim}}$ to perform the association. Therefore, we determine a similarity cost threshold t_c^{sim}, which resolves whether the instance is assigned to the corresponding track. The goal is to utilize the appearance features for larger displacements, where the geometric assignment is difficult since multiple possible instance candidates exist. The similarity cost threshold also handles occlusion and initializes new tracks. Furthermore, we define a second distance-based threshold t_{d2} where the appearance-based association is difficult, and the association is omitted due to false positive assignments. We update the existing tracks with the information of the assigned instances after the processing of the individual scans. Occluded tracks are propagated according to the temporal offset predictions $\mathbf{O}^{\text{temp}}$, and we initialize new tracks with the corresponding information of the instance, including the instance feature representation and the offset information. The final data association incorporates geometric and appearance features to enhance tracking performance.

8.2 Implementation Details

We implemented our approach in PyTorch [161] and trained the instance segmentation backbone and our Radar Tracker with one Nvidia A100 GPU. We adopt the training parameter of the Radar Instance Transformer, see Section 7.2. To learn the spatial and temporal offsets of the radar detections, we add for both offsets the following loss function:

$$\mathcal{L}^{\text{offset}} = \frac{1}{N} \sum_{i=1}^{N} \lVert \mathbf{o}_i - (\mathbf{c}_i - \mathbf{p}_i) \rVert_1, \tag{8.4}$$

where N is the number of points in the point cloud, and $\mathbf{c}_i$ is the respective ground truth center of the instance that $\mathbf{p}_i$ belongs to. We extract the ground truth center for the temporal offset by transferring the points of the instances of the future scan into the current scan. The transformations are given in the dataset.

We utilize the AdamW [132] optimizer with an initial learning rate of 0.001 to train our Radar Tracker. We process the original features with the input dimension $D = 4$, comprising the point coordinates $\left[x_i^C, y_i^C\right]^\top$, the radar cross section σ_i, and the ego-motion compensated Doppler velocity v_i, by the transformer blocks where $D_1 = 64$ and $D_2 = 256$. We define the local areas for sample and grouping by $N_l = 6$ points. We reduce the number of points because we only consider the moving predictions, in contrast to Chapter 7. The attentive instance aggregation module keeps the feature dimension and combines the information within one instance feature vector. One batch includes 64 scan pairs, where only the first scan is considered during the loss calculation. The network is able to predict the data association and if the objects within the first scan are present in the second scan. We supervise the attentive similarity output by a binary cross-entropy loss.

We set the bandwidth $b = 10$ for the clustering using DBSCAN to determine local areas for the association. We set the distance based thresholds $t_{d1} = 5\,\text{m}$ and $t_{d2} = 10\,\text{m}$. We keep the tracks for 12 consecutive scans. The cost threshold for the attentive similarity is set to $t_c^{\text{sim}} = 1.5$. We add points belonging to the static class as additional instances for data augmentation to include the differentiation between static and moving points in the attentive similarity. This way, we account for false predictions during inference and enable the network to reliably estimate the associations.

8.3 Experimental Evaluation

The main focus of this chapter is to enable reliable moving instance tracking in sparse and noisy radar point clouds. We present our experiments to show the capabilities of our method and to support our key claims, which include that our method outperforms existing state-of-the-art methods in moving instance tracking. Secondly, our temporal offset prediction enhances the classification and tracking score by adding additional motion cues. Thirdly, our attention-based association scores incorporate valuable appearance features enhancing performance.

8.3.1 Experimental Setup

We train and evaluate our model on the RadarScenes dataset, as explained in detail in Chapter 5. We utilize the LiDAR segmentation and tracking quality (LSTQ) score [7] to evaluate the moving instance tracking performance. The LiDAR segmentation and tracking quality is designed to evaluate point-based segmentation and tracking methods and does not depend on LiDAR-specific properties. Additionally, the metric ensures comparability for follow-up research. The LiDAR segmentation and tracking quality combines the classification score $\mathrm{S}_{\mathrm{cls}}$ for the semantic evaluation and the association score $\mathrm{S}_{\mathrm{assoc}}$ for the temporal evaluation, resulting in $\mathrm{LSTQ} = \sqrt{\mathrm{S}_{\mathrm{cls}} \times \mathrm{S}_{\mathrm{assoc}}}$.

8.3.2 Moving Instance Tracking

The first experiment evaluates the performance of our approach and shows that our method achieves state-of-the-art performance in moving instance tracking in sparse and noisy radar scans. We compare our Radar Tracker to the high-performing networks with strong performance in LiDAR point-based tracking benchmarks. However, we do not consider the best performing Eq-4D-StOP [281] since it incorporates large rotations of the input point clouds, which is detrimental to radar data [158].

Consequently, we utilize CA-Net [136] proposed by Marcuzzi et al., MOT [223] by Weng et al., and the center tracking approach proposed by Yin et al. [252] as baselines. We extend the center tracking with Hungarian matching [104] and directly use the measured ego-motion-compensated Doppler velocities to perform tracking instead of predicting velocities of the individual bounding boxes. For the MOT [223] approach, we utilize the IoU as the cost to illustrate the limitations in instance association. We adopt the Radar Instance Transformer, introduced in Chapter 7, as the backbone for all methods since it is the best-performing approach for moving instance segmentation and thus ensures a fair comparison.

Our Radar Tracker outperforms the existing methods, especially in terms of LiDAR segmentation and tracking quality and association scores, as displayed in Table 8.1. As mentioned, the MOT [223] approach struggles to associate small instances due to the IoU-based association. In particular, instances that consist of single detections do not result in an intersection and, therefore, limit

Approach	LSTQ	S_{assoc}	S_{cls}
MOT [223]	42.4	19.4	**92.7**
Center tracking [252] + Hungarian [104]	59.3	38.0	**92.7**
CA-Net [136]	34.8	13.0	**92.7**
Ours	**66.8**	**48.2**	**92.7**

Table 8.1: Moving instance tracking results on the RadarScenes test set in terms of LiDAR segmentation and tracking quality (LSTQ), classification scores (S_{cls}) and association scores (S_{assoc}). Our approach outperforms state-of-the-art methods and enhances instance association for tracking. The best results are in bold.

the performance. The center-based tracking overcomes these limitations and enhances performance. Nevertheless, MOT [223] and center tracking with Hungarian matching [104] neglect the appearance features of the instances and thus can not compensate for the shortcomings of geometric tracking. CA-Net combines both features within one cost function. However, the method struggles to extract discriminative instance information and to associate the instances based on appearance features. Relying on the geometric features within small displacements can help to omit false predictions.

To illustrate the capabilities of our method, we show in Figure 8.3 two qualitative results of the Radar Tracker. Our approach shows good performance within diverse scenarios, including adverse weather. For the rainy scene, we are able to track approaching objects, which change the number of points, which is difficult for IoU-based approaches since, for really small instances, no overlap might exist. Furthermore, the urban street scene includes several instances that belong to the same semantic class, which makes tracking challenging. However, our approach handles the complex scene well and provides consistent tracking IDs. We argue that extracting appropriate features is challenging in sparse radar data, and the design of our network and association is crucial to enhance accuracy. Our method reliably tracks moving instances by combining geometric and appearance features.

8.3.3 Ablation Study on Offset Predictions

The second experiment evaluates our offset predictions, especially the temporal offset, and illustrates that our approach is capable of including valuable motion cues to enhance segmentation and tracking quality. To evaluate the segmentation performance, we utilize the $\mathrm{IoU}^{\mathrm{mov}}$ since segmentation of moving detection is essential for tracking. We extend the backbone, the Radar Instance Transformer, with the spatial offset prediction and the temporal offset prediction as additional regression targets, as depicted in Table 8.2. The spatial offset, which points to

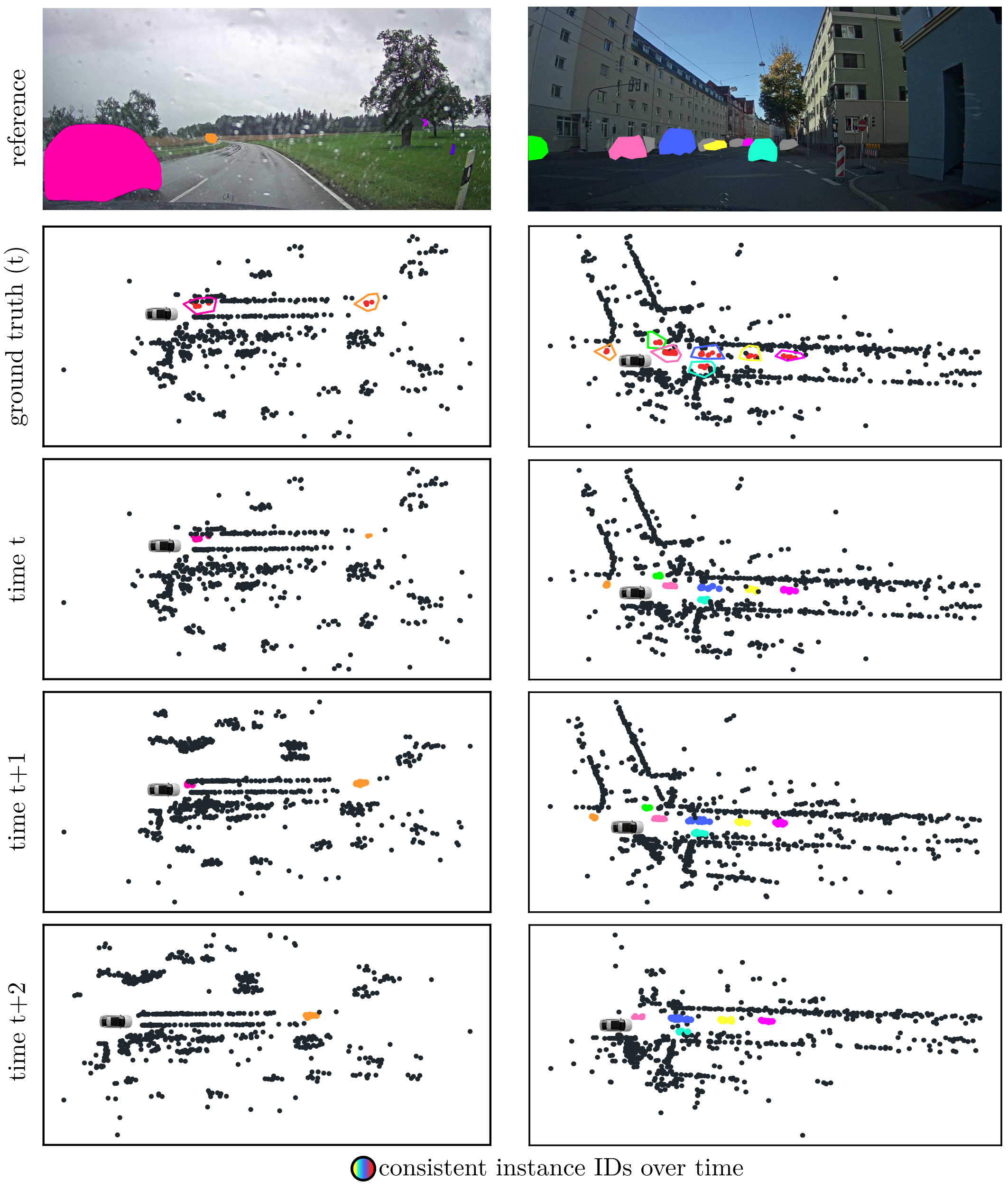


Figure 8.3: Qualitative results of our Radar Tracker on the test set of RadarScenes over time t. The camera images are anonymized and shown for reference. The colors in the image correspond to the predicted instances over time. The left column is from sequence 79 (rain), and the right is from sequence 138 (urban). In the images of the predictions, each color represents a different instance of moving objects (static is black). The colors in the images correspond if the object is visible.

the center of the instance within the current scan, already improves the IoU^{mov} by 0.8 absolute percentage points. Despite this improvement, the temporal offset prediction enhances the performance by an additional 0.1 absolute percentage point. We presume that the temporal offset prediction includes well-defined motion cues, especially for instances comprising single detection that do not have a

Approach using:	spatial offset	temporal offset	IoU^{mov}
Radar Instance Transformer (Chapter 7)			84.4
Ours	✓		85.2
Ours		✓	85.3
Ours	✓	✓	**85.4**

Table 8.2: Influence of the temporal and spatial offset predictions in terms of IoU^{mov} on the RadarScenes validation set. The temporal and spatial offset improves segmentation performance. The best results are in bold.

Approach	S_{assoc}
Ours w/o positional encoding	54.0
Ours with no object class	54.1
Ours w/o sigmoid	54.0
Geometric association $t_{d2} = 10\,\text{m}$	52.2
Ours	**54.3**

Table 8.3: Influence of the design decision for the attentive association on the RadarScenes validation set. The comparison with the geometric association shows that our approach leads to superior results by considering appearance features. The best results are in bold.

regression target for the spatial offset. Therefore, we observe a better moving object segmentation performance if we only add the temporal offset. If we combine both offset predictions to enable direct center-based tracking, we achieve the best IoU^{mov} of 85.4%.

To verify that both offset predictions improve the tracking performance, we evaluate a simple center-based association with and without the center predictions in an additional experiment. We remove the appearance features to strictly assess the performance of the geometric approach. The offset prediction improves the S_{assoc} from 49.5% to 50.2%, which underlines the advantage. As a result, the jointly learned instance and offset predictions combine the advantages and improve segmentation and tracking quality.

8.3.4 Ablation Study on Attentive Association

In our third experiment, we analyze our method concerning the ability to extract reliable attentive similarity scores to associate instances. Therefore, we evaluate the different components of our method as detailed in Table 8.3. First, we remove the positional encoding within our attentive instance association, which results in a decrease of 0.3 absolute percentage points. We argue that positional encoding is important to differentiate between similar instances within the scan. Furthermore,

Model	parameters (M)	mean runtime (ms)
Radar Instance Transformer (Chapter 7)	3.8	31.7
Center tracking [252] + RIT	3.8	32.0
CA-Net [136] + RIT	7.2	46.7
Ours + RIT	4.6	43.7

Table 8.4: The evaluation of the mean runtime and the number of parameters of the models on an Nvidia RTX A6000 GPU based on 1000 randomly sampled point clouds of the RadarScenes dataset.

we are able to combine appearance and geometric features to enhance tracking quality.

In the second step, we add an additional no-object regression target [261] to the attention score to address the occlusion within the appearance features. However, small and distant instances are often detected in one scan but not covered in the next one, leading to several no-object assignments as ground truth. We assume that this forces the network to assign more instances to the no object class, and the information to track the instances is not covered adequately, which results in a 0.2 absolute percentage points decrease of S_{assoc}. Additionally, we tried to remove the sigmoid function [261] to directly learn the attention scores. However, this also results in a decrease in performance. To verify that the association based on the attention scores enhances accuracy, we evaluate our method, including only geometric information for the threshold $t_{d2} = 10\,\text{m}$. The geometric association performs worse compared to our approach. Hence, the appearance features are essential to improve tracking and resolve ambiguities between instances at larger distances.

8.3.5 Runtime

We analyze the runtime of our approach and show that our approach runs fast enough to support online processing in autonomous vehicles. We evaluated the runtime of our approach and the best-performing learning-based approach following Section 5.3.5.

Our Radar Instance Transformer has a runtime of 31.7 ms and comprises 3.8 million parameters, as introduced in Chapter 7. Our Radar Tracker adds a runtime of 12 ms, including 800 k parameters. The CA-Net utilizes a larger network and hence increases the runtime compared to our approach. The geometric association of the center tracking results in the lowest overall runtime but neglects the valuable appearance features, reducing the tracking quality. The resulting runtime of 43.7 ms equals a frequency of 22 Hz, which is faster than the frequency of 17 Hz of the sensors. As a result, we are able to incorporate the appearance features with a minor overhead and solve the task of moving instance tracking.

In summary, our evaluation suggests that our method provides competitive moving instance tracking results in sparse and noisy radar point clouds by incorporating geometric and appearance features. Thus, we supported all our claims with this experimental evaluation.

8.4 Conclusion

In this chapter, we addressed the task of segmentation and tracking of moving objects, which is essential for reliable path planning and collision avoidance. To enhance scene understanding, we presented a novel method for moving instance tracking in sparse radar point clouds. We follow the tracking-by-detection paradigm since moving instance segmentation produces exceptional results by exploiting radar-specific properties such as the Doppler velocity.

Our approach exploits temporal offset predictions to encode geometric information to enhance segmentation and tracking. We combine temporal offset predictions and spatial offset predictions of the current scan to maximize segmentation performance and include geometric features in the instance association. We further infer instance appearance features using a transformer-based network and introduce an attention-based association cost function to improve the tracking quality. This allows us to successfully associate individual instances based on valuable geometric and appearance features over time. We experimentally evaluated our approach on the radar moving instance tracking benchmark based on the RadarScenes dataset. The results suggest that combining geometric and appearance features is essential to achieve good performance on moving instance tracking in sparse radar data. Furthermore, our approach successfully associates instances that comprise only a single point, resolving the limitations of IoU-based association methods. Overall, our approach outperforms state-of-the-art methods, paving the way for reliable moving instance tracking in sparse radar data.

The tracking of moving instances supports the scene understanding and is essential for future state prediction. Consequently, temporal understanding is crucial for advanced driving systems. However, humans do not only rely on the perception of moving and static objects but predict the maneuvers based on semantic understanding. This knowledge is important because the possible movement of cars and pedestrians differs. Therefore, semantic segmentation helps to improve scene understanding, which we address in the next chapter.

Chapter 9

Panoptic Segmentation

The moving instance segmentation and tracking provide valuable information to improve scene understanding, as presented in Chapter 8. To navigate safely in complex real-world driving scenarios, additional information, such as the semantic class of dynamic traffic participants, is useful. It must be correctly identified to ensure safe autonomous mobility because interaction with different traffic participants plays a crucial role in collision avoidance and path planning. Furthermore, the semantic classes provide information about the possible movements of the agents. For example, the movement of cars or trucks is more restricted due to the rigid structure and the larger inertia, as well as road boundaries and traffic rules. Hence, knowledge about the semantic class is important for anticipating individual behaviors and providing context-aware decisions.

In this chapter, we tackle the problem of panoptic segmentation of moving agents in sparse and noisy radar point clouds. We need to distinguish between moving and static parts of the environment and separate individual instances within our surroundings. We assign a corresponding semantic class and consider the moving instances as “things” and the static points as “stuff”. This combines instance segmentation with semantic segmentation to panoptic segmentation. We focus on dynamic instances and hence include the differentiation between moving and non-moving objects, such as parked and moving vehicles. Radar sensors are particularly suitable to identify moving agents because they provide the Doppler velocity of the detection, resulting in exceptional moving instance segmentation, as shown in Chapter 7. We aim to leverage the reliable moving instance predictions and propose a semantic instance refinement, as illustrated in Figure 9.1, to enhance scene understanding. Compared to state-of-the-art approaches [190, 269], we are able to work on single-scan radar data and do not rely on scan aggregation, which can increase memory consumption and induce latencies, limiting the applicability in real-world scenarios.

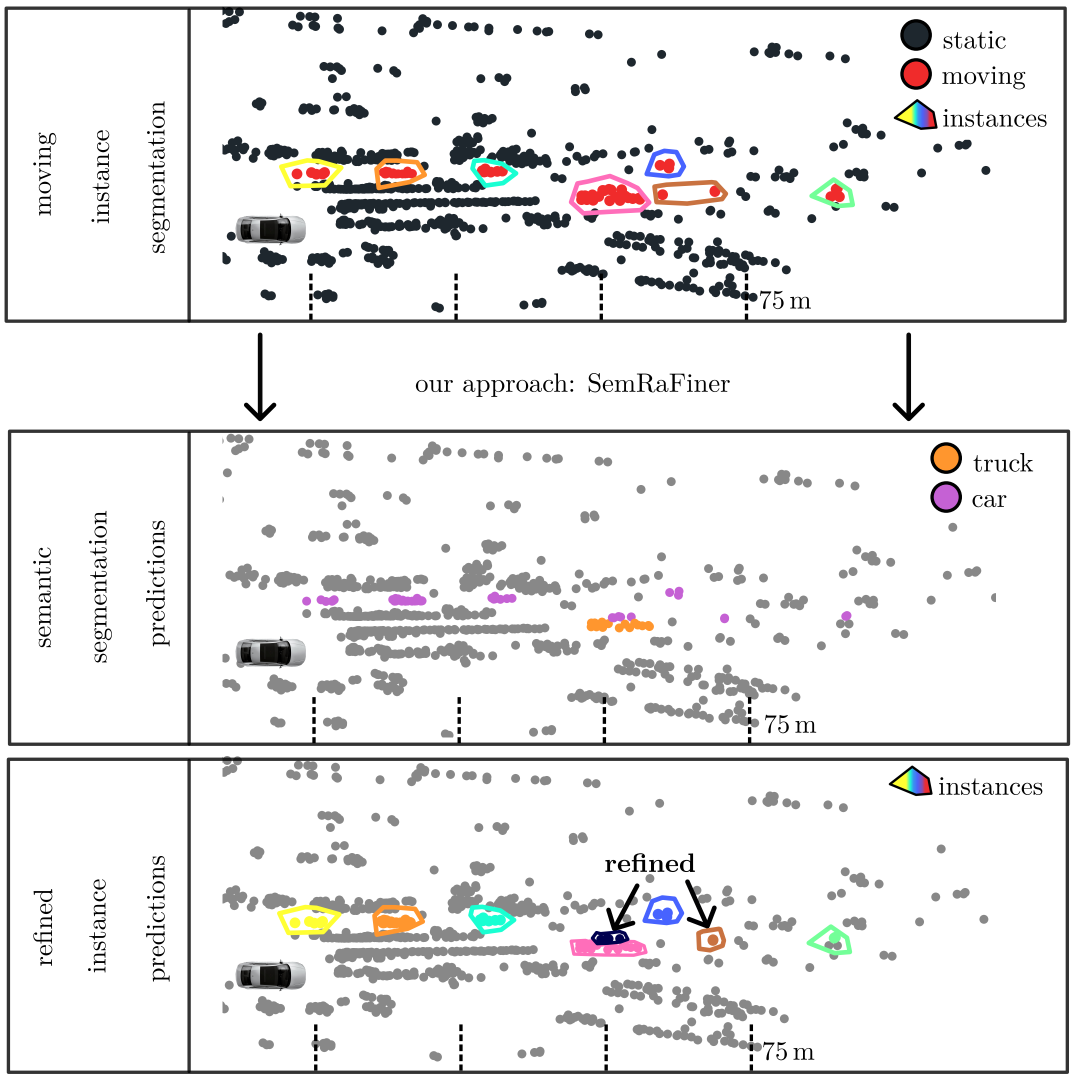

moving instance segmentation
static
moving
instances
75 m
our approach: SemRaFiner
semantic segmentation predictions
truck
car
75 m
refined instance predictions
instances
refined
75 m

Figure 9.1: Our method processes moving instance segmentation predictions (a) to predict point-wise semantic classes (b) and refine the instance assignment (c) to solve the panoptic segmentation task from sparse radar point clouds. In the image of the point cloud (a), each polygon represents a different instance of moving objects.

Additionally, Zhou et al. [274] and Xiang et al. [237] utilize feature refinement within the backbone to enhance performance by combining multiple representations, which increases the computational burden. Therefore, we focus on leveraging the predictions to enhance accuracy efficiently and only add a small overhead to the backbone. Furthermore, instead of directly solving the task of panoptic segmentation, such as Li et al. [118], Xiao et al. [239], and Kolodiazhnyi et al. [99], our approach leverages the advantages of solving two separate tasks with very good performance enhancing overall accuracy. We first perform moving instance segmentation to leverage the advantages, such as incorporating underrepresented classes and reducing the class imbalance. Then, we refine the instance segmentation and predict the semantic classes in a two-step approach.

We overcome the limitations of scan aggregation and increased computational burden with a lightweight network focusing on the essential information to enhance performance.

Our main contribution is a novel method for accurate panoptic segmentation in sparse and noisy radar point clouds. The approach, called SemRaFiner, leverages the advantages of moving instance segmentation, which does not require differentiation between semantic classes but allows us to enhance the overall performance. We optimize the layers of our network and propose a new self-attention module, our radius transformer layer, to account for the changing density of radar point clouds, especially when processing moving instance predictions. We utilize dedicated data augmentation to further refine the instance prediction and enhance panoptic segmentation. We construct an efficient network incorporating the individual modules and an optimized training approach to improve the performance of the overall perception task.

In sum, we make four key claims in this chapter: First, our SemRaFiner shows state-of-the-art performance for panoptic segmentation in sparse and noisy radar point clouds. Second, our radius transformer layer optimizes feature extraction, especially for processing moving object predictions. Third, our optimized training process, including dedicated data augmentation to refine instances and improve semantic predictions, results in better performance. Fourth, our proposed approach runs faster than the sensor frame rate.

9.1 Our Approach to Panoptic Segmentation

The goal of our approach is to achieve precise panoptic segmentation in sparse and noisy radar point clouds to enhance scene understanding of autonomous vehicles. Our SemRaFiner architecture, illustrated in Figure 9.2, utilizes the prediction of our Radar Instance Transformer, introduced in Chapter 7, and extends the backbone to refine instance assignments and include semantic classes. Our network is a point-based transformer network that builds upon the self-attention mechanism [212]. We directly process the individual points using our radius transformer layer to enhance feature extraction by addressing the specific spatial distribution of radar points. We further enhance the instance prediction by enabling the network to account for false predictions through our training procedure. The final point-based prediction includes the corresponding semantic class and refined instance assignments.

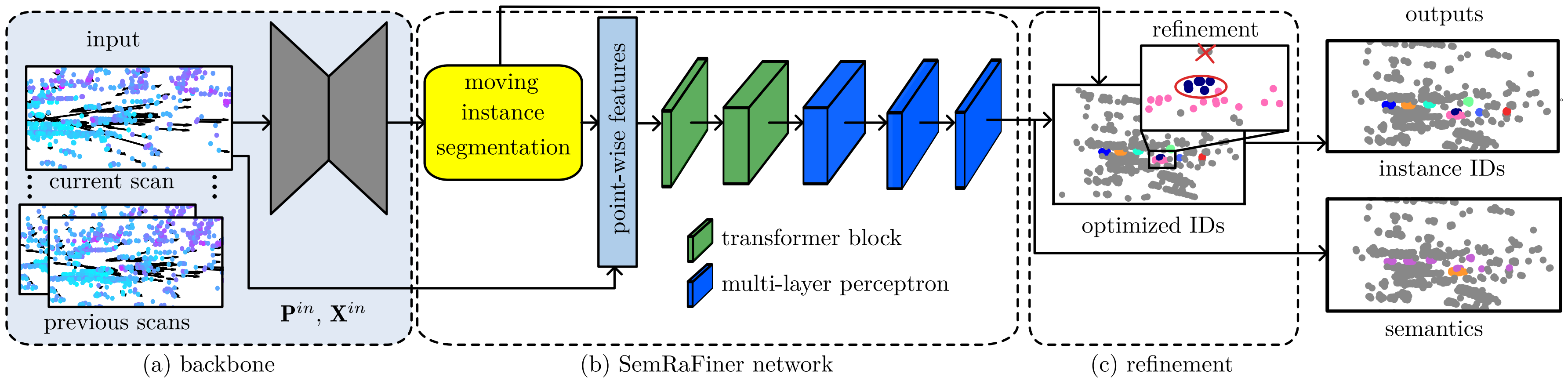


Figure 9.2: Detailed design of the individual modules of our SemRaFiner. (a) The backbone provides the moving instance predictions. (b) Our network extracts valuable features from filtered moving instances. (c) The semantic prediction of our approach refines the instance assignment. The final output includes the semantic predictions and corrected instance IDs. The width of the transformer blocks and multi-layer perceptrons illustrate the number of feature channels.

9.1.1 Moving Instance Segmentation Backbone

Radar data is typically sparse and noisy, but it provides additional Doppler information, which is useful for moving object detection and simplifies moving instance segmentation. Due to the enhanced performance of moving instance segmentation (Chapter 7) compared to standard semantic segmentation (Chapter 5), our idea is to exploit this prediction performance advantage to improve the accuracy of panoptic segmentation. Therefore, we utilize the state-of-the-art moving instance segmentation method, our Radar Instance Transformer, presented in Chapter 7, to reliably determine moving instances and refine these predictions to improve panoptic segmentation.

We follow the design of our Radar Instance Transformer and enrich the current radar point cloud $\mathcal{P}^t$ at time t, consisting of the coordinates of the points and the radar features, including the Doppler velocity and radar cross section values, with temporal information from T previous scans $\mathcal{P}^{t-T}, \ldots, \mathcal{P}^{t-1}$. The enriched point cloud is processed by transformer blocks to extract valuable features for attention-based graph clustering to determine the instance IDs $\mathcal{I} = \{I_1, \ldots, I_N\}$ with $I_i \in \mathbb{N}$. Additionally, the outputs of the backbone include the binary moving object segmentation labels $\mathcal{S}^{\text{MOS}}$. For processing the current point cloud, we utilize the predicted labels to filter the point clouds for moving prediction, which is the input to our method. The features of the backbone remain untouched after the training, and only the moving instance segmentation predictions are required to select the corresponding data points, as introduced in Chapter 8. Therefore, we do not rely on extracted features of the backbone, and we can potentially substitute the network for other moving instance segmentation approaches. Furthermore, the filtering reduces the computational complexity, and the majority of the point cloud does not depend on an additional processing step. However, the good performance of the moving instance segmentation backbone is crucial for our method to improve panoptic segmentation.

9.1.2 Radius Transformer Layer

In sparse radar point clouds, individual points can represent whole instances, such as distant vehicles or nearby pedestrians. Therefore, they contain important information for safe autonomous mobility. Feature extraction of sparse radar data is key to enhancing performance for downstream tasks. Since we process the filtered point cloud, often including only moving instances, the point clouds are sparse and unevenly distributed compared to the unfiltered point cloud, including static points. To address this issue, especially for sparse moving instance prediction, we propose a radius transformer, which combines a ball query sampling strategy with a vector attention mechanism. We argue that the most important information for correctly classifying individual points is within the local area of the object. Most state-of-the-art approaches also focus on the local area. However, due to the filtering of the point clouds explained in Section 9.1.1, individual instances are distributed over the whole field of view. Performing attention between the k nearest neighbors, explained in the previous chapters, or determining the corresponding points with serialization algorithms [235] can result in interconnections of distant points, which is likely to harm the feature extraction capability for our type of problem. In contrast to our other approaches in previous chapters, our radius-based approach limits the considered related local area for sparse point cloud processing.

Other approaches by Lai et al. [108] and Xin et al. [242] restrict the local area by performing attention on predefined areas. However, the grid representation is fixed, which does not necessarily align with the instances. Therefore, we propose a radius instance transformer layer, defining the local area for each point individually to overcome this shortcoming. Consequently, the local area covers the important information to segment individual detections reliably.

The input to our radius transformer layer contains the point coordinates $\mathbf{P}^{\text{in}}$ and point-wise features $\mathbf{X}^{\text{in}}$ of the moving instances within the current scan, as explained in Chapter 8. We do not process the static predictions. In contrast to the previous chapter, we process the input features $\mathbf{X}^{\text{in}} \in \mathbb{R}^{N^{\text{mov}} \times D}$ by two fully connected layers with weight matrices $\mathbf{W}_1^b \in \mathbb{R}^{D \times D_1}$ and $\mathbf{W}_2^b \in \mathbb{R}^{D_1 \times D_1}$ to increase the feature dimension, which we elaborate in Section 9.1.3. In our transformer layer, we follow our general approach, introduced in Chapter 5, and encode the features of the moving points by fully connected layers.

To adaptively combine the features within the local areas and extract valuable features for individual points, we need to group the points. As mentioned before, to account for the changing density of points and replace the grid representation with a flexible solution, we sample the points within a circle defined by the radius r, around the individual points. We determine the neighborhood of the corresponding points with point coordinates $\mathbf{p}$ by the relative position $\mathbf{r}_{i,j} = \mathbf{p}_i - \mathbf{p}_j$ where $\mathbf{p}_i$ and $\mathbf{p}_j \in \mathbf{P}^{\text{in}}$. The domain of definition for our self-attention is then the circle $\mathcal{B}_r(\mathbf{p}_i) = \{\mathbf{p}_i \in \mathbf{P}^{\text{in}} \mid \|\mathbf{r}_{i,j}\| \leq r\}$ with radius r, similar to the ball query of PointNet++ [166]. However, we replace the processing within the local areas of PointNet++ with a powerful attention mechanism. Furthermore, we observe that having many interconnections between points can make it difficult to extract valuable features within local regions, consisting of different semantic classes, due to the information exchange within the attention mechanism. Hence, we restrict the maximum number of points N^{max} within the radius, resulting in an optimized neighborhood representation to enhance feature extraction.

We sample the points using a ball query [210] to extract the related queries,

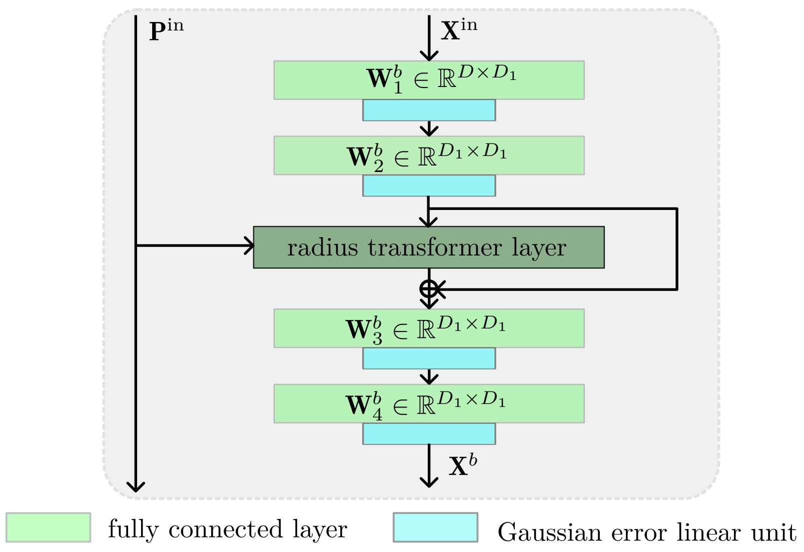

$\mathbf{P}^{\text{in}}$
$\mathbf{X}^{\text{in}}$
$\mathbf{W}_1^b \in \mathbb{R}^{D \times D_1}$
$\mathbf{W}_2^b \in \mathbb{R}^{D_1 \times D_1}$
radius transformer layer
$\mathbf{W}_3^b \in \mathbb{R}^{D_1 \times D_1}$
$\mathbf{W}_4^b \in \mathbb{R}^{D_1 \times D_1}$
$\mathbf{X}^b$
fully connected layer
Gaussian error linear unit

Figure 9.3: Our transformer block is a residual block that includes our radius transformer layer and two multi-layer perceptrons with two fully connected layers and Gaussian error linear unit activation functions. The input includes the point coordinates $\mathbf{P}^{\text{in}}$ and the point-wise features $\mathbf{X}^{\text{in}}$. The output of the transformer block consists of the enriched features $\mathbf{X}^b$, and the point coordinates $\mathbf{P}^{\text{in}}$, which are kept intact to include fine-grained position information within consecutive processing steps.

keys, and values, resulting in $\mathbf{Q}^{\text{r}}, \mathbf{K}^{\text{r}}$, and $\mathbf{V}^{\text{r}} \in \mathbb{R}^{N^{\text{mov}} \times N^{\text{max}} \times D_1}$ within the local neighborhoods. If the local area around the points contains less than N^{max} points, the remaining keys, queries, and values are zero-padded. Furthermore, we utilize the relative position $\mathbf{r}_{i,j}$, which we calculate to determine the local areas to determine the relative positional encoding $\mathbf{R} \in \mathbb{R}^{N^{\text{mov}} \times N^{\text{max}} \times D_1}$. We follow our previous approaches explained in Chapter 5 and adopt vector attention [270] to calculate the attention weights $\mathbf{A} \in \mathbb{R}^{N^{\text{mov}} \times N^{\text{max}} \times D_1}$. In contrast to our tracking approach, we process the resulting attention weights by a multi-layer perceptron (MLP), including two fully connected layers with weight matrices $\mathbf{W}_1^a, \mathbf{W}_2^a \in \mathbb{R}^{D_1 \times D_1}$, two batch normalization [88] layers, and rectified linear unit activation functions [150]. We then apply the softmax function to derive the final attention weights $\hat{\mathbf{A}}_{i,j}$. The output features $\mathbf{X}^{\text{out}} \in \mathbb{R}^{N^{\text{mov}} \times D_1}$ of our radius transformer layer are the sum of the element-wise multiplication and the relative positional encoding. Besides the features, which comprise the information within the local area, the output of the transformer layer includes the point coordinates $\mathbf{P}^{\text{in}}$ of the moving points. The point coordinates are kept to include fine-grained position information within the consecutive layers.

9.1.3 SemRaFiner Network

We aim to efficiently process the radar data to reduce latencies. Consequently, our SemRaFiner includes only two transformer blocks and three MLPs to derive the refined instance IDs $\mathcal{I}^r = \{I_1^r, \dots, I_N^r\}$ with $I_i^r \in \mathbb{N}$ and semantic labels $\mathcal{S}^{\text{sem}}$, see Figure 9.2.

We follow our approaches described in previous chapters and embed our radius transformer layer into a residual block to extract valuable features, see Figure 9.3. We first increase the dimension of the features $\mathbf{X}^{\text{in}}$, which we then process within the radius transformer layer. The output features of the radius transformer layer are added to the features of the skip connection and processed by the second MLP. The point coordinates are directly passed into the transformer layer to calculate the positional encoding and include fine-grained position information within the local areas. The resulting features after the second transformer block $\mathbf{X}_2^b \in \mathbb{R}^{N^{\text{mov}} \times D_2}$ are processed by the three consecutive MLPs, as illustrated by the blue blocks in Figure 9.2, to reduce the dimensionality and derive the final predictions. The first two MLPs consist of two fully connected layers, where the first fully connected layer keeps the dimension of the features, and the second reduces the dimension by a factor of two. To derive the final semantic predictions, the last MLP first reduces the dimensionality by a factor of two before predicting the semantic labels. The final semantic predictions $\mathcal{S}^{\text{sem}} = \{s_1^{\text{sem}}, \dots, s_N^{\text{sem}}\}$ where $s_i^{\text{sem}} \in \{1, \dots, C\}$ combine the static predictions of the backbone and the refined semantic class predictions. The predicted semantic classes of our network for the moving predictions of the backbone include the static class to enable instance refinement.

9.1.4 Data Augmentation

We utilize ground truth annotations to train our model and to classify instances within sparse radar point clouds reliably. During inference, we leverage the moving instance prediction to perform point-wise semantic segmentation and refine the instance prediction. The major problem is that the ground truth does not include failure cases such as static points predicted as moving. To account for the false prediction of static points, we augment our training data. Our idea is to enable our network to correct false predictions, which cannot be learned by only considering the ground truth annotations. Furthermore, we want to enable the method to leverage the semantic information for panoptic segmentation to correct false predictions. Therefore, we incorporate static points into the training process and add static points close to an instance by the probability of p_I to learn to correct false predictions. This approach also aligns with the false predictions, which often occur at the boundary of the instance. Additionally, we observe that the moving instance predictions include false predictions for small instances within the individual scans. Therefore, we add instances comprising only static ground truth annotations, including one to five points by the probability of p_S. The correction of these predictions can eliminate incorrectly detected instances.

9.1.5 Instance Refinement

Our data augmentation enables us to correct false predictions and remove static points from instance predictions. Furthermore, we utilize the semantic predictions to correct instance assignments. Since the classification of moving and static points does not include further differentiation, we observe that nearby instances belonging to different semantic classes, such as trucks and cars, may get clustered together. However, our panoptic segmentation includes that knowledge, and hence, we can resolve the false assignments. We combine the instance assignments and semantic predictions within our instance refinement module. Due to the point-wise processing, we derive a semantic class for each individual radar point. We combine these predictions and the instance assignment per point and assign different instance IDs if the semantic classes differ. Additionally, we remove the instance ID if our approach predicts that the corresponding point belongs to the static class. The overall refinement thus accounts for false semantic predictions and wrong instance assignments.

9.2 Implementation Details

We train the backbone separately and adopt the original parameters introduced in Section 7.2. We implemented our SemRaFiner in PyTorch [161] and trained our network with one Nvidia A100 GPU and a batch size of 64 over 80 epochs. As optimizer, we utilize AdamW [132], set the initial learning rate to 0.001, and drop the learning rate by a factor of 10 after 60 epochs. We combine the Lovász loss [21] and cross-entropy loss to learn the point-wise classification using the same weighting for all classes. Compared to the straightforward semantic segmentation in Chapter 5, the processing of the moving instance predictions does not include the strong class imbalance. Therefore, the loss function does not include a weighting of the classes.

Additionally, we introduce a consistency loss to enforce the same class prediction for the individual instances. We construct a loss based on the semantic predictions $\mathcal{I}_h^{\text{sem}}$ for the individual instances h. The ideal case is that each instance belongs to the same class. However, this limits the performance since we cannot correct false predictions during instance refinement. Therefore, it is important to maintain point-wise classification while preserving consistency within the true instances. We utilize the number $|\mathcal{I}_h^{\text{sem}}|$ of distinct classes within an instance $\mathcal{I}_h^{\text{sem}}$ to calculate the loss over the number of all instances N_h as follows:

$$\mathcal{L}_c = \frac{1}{N_h} \sum_{h=1}^{N_h} 1 - \frac{1}{|\mathcal{I}_h^{\text{sem}}|}. \tag{9.1}$$

We combine all three losses without weighting to derive the final training loss for our network. The input features comprise the point coordinates, the radar cross section σ_i, and the ego-motion-compensated Doppler velocity v_i, forming the input vector $\mathbf{x}_i = \left[x_i^C, y_i^C, z_i^C, \sigma_i, v_i\right]^\top$. The coordinate $z_i^C = 0$ is added to apply the pose compensation, explained in Section 7.1.1. We increase the per-point features to $D_1 = 64$, $D_2 = 256$ within our transformer blocks. We set the parameters for our radius transformer layer to $r = 5.0\,\mathrm{m}$ and the maximum considered neighbors within the ball query to $N^{\max} = 24$. The first two MLPs reduce the dimensionality to 128 and 64, respectively. In the third MLP, the first fully connected layer has an output dimension of 32 before predicting C classes by the second fully connected layer. We set the probabilities of augmenting the instances p_I and the scan p_S to $40\,\%$.

9.3 Experimental Evaluation

The main focus of this chapter is to achieve accurate panoptic segmentation in sparse radar point clouds. We present our experiments to show the capabilities of our method. The results of our experiments support our key claims, including that our approach achieves state-of-the-art performance in panoptic segmentation without aggregating scans. Moreover, our radius instance transformer layer accounts for the changing density in sparse radar data to enhance panoptic segmentation. Our dedicated data augmentation and point-wise processing enables instance refinement to improve overall performance. Our SemRaFiner is efficient and only adds a small overhead to the backbone to incorporate semantic classes.

9.3.1 Experimental Setup

As detailed in Chapter 5, we train and evaluate our model using the RadarScenes dataset. Consistent with Chapter 7, we utilize the panoptic quality to evaluate the panoptic segmentation. Additionally, we report the intersection over union (IoU), introduced in Chapter 5, to evaluate the semantic segmentation performance in detail. To enable a comprehensive evaluation, we further differentiate between the six classes, including static, pedestrian, pedestrian group, car, truck, and bike.

9.3.2 Panoptic Segmentation

The first experiment evaluates the performance of our approach and its outcomes support the claim that our approach achieves state-of-the-art performance in panoptic segmentation using sparse and noisy radar scans. We compare our

Method	Input	PQ	mIoU	PQ						IoU					
				static	car	pedestrian	pedestrian group	bike	truck	static	car	pedestrian	pedestrian group	bike	truck
RadarPNv1 [188]	multi-scan	-	61.0	-	-	-	-	-	-	98.7	58.2	36.0	58.7	58.4	56.1
RadarPNv2 [190]	multi-scan	-	61.9	-	-	-	-	-	-	98.7	63.8	38.8	58.5	51.0	61.0
STA-Net [269]	multi-scan	-	**70.4**	-	-	-	-	-	-	**99.8**	**75.9**	**43.1**	**82.0**	**67.1**	54.6
Mask3D [187]	single-scan	56.9	56.1	98.4	70.6	18.4	31.9	57.7	64.3	98.6	68.7	11.8	35.9	57.7	63.9
Chapter 5 + DBSCAN [53]	single-scan	56.3	56.9	98.7	59.0	29.7	38.3	54.6	57.3	98.7	58.5	21.7	46.0	54.6	61.8
Ours	single-scan	**81.4**	**70.4**	**99.7**	**85.7**	**59.2**	**83.1**	**78.9**	**82.1**	99.4	74.9	42.1	71.0	65.2	**69.6**

Table 9.1: Panoptic segmentation results on the RadarScenes test set in terms of PQ and IoU scores. The results of RadarPNv1 [188], RadarPNv2 [190] and STA-Net [269] are adapted from the original papers. The gray rows correspond to multi-scan segmentation methods. Our approach outperforms state-of-the-art approaches and performs on par with aggregation-based methods. The best results are in bold.

method to high-performing networks with strong performance in point-based instance segmentation and radar segmentation. Furthermore, we utilize the approach Mask3D [187] proposed by Schult et al. as a baseline and extend our Gaussian Radar Transformer with DBSCAN [53] to reliably cluster instances. We extend Mask3D [187] with state-of-the-art post-processing for mask predictions [36] and set the score threshold to 0.5 to enhance performance and derive the panoptic labels.

Additionally, we report the results of the works by Schumann et al., namely RadarPNv1 [188] and RadarPNv2 [190], and the STA-Net [269] by Zhang et al., which rely on scan aggregation and do not consider single-scan processing. Our SemRaFiner outperforms the existing single-scan approaches, especially in terms of PQ, by a solid margin, as displayed in Table 9.1. The performance improves, especially for small instances comprising only a few points, such as pedestrians and distant objects. We observe that our dedicated network, which utilizes the moving instance segmentation as input, is able to extract valuable features, particularly for smaller classes, due to the reduced class imbalance and optimized architecture. We emphasize that our performance depends on the prediction of our Radar Instance Transformer, and therefore, precise moving instance segmentation is important to enhance the accuracy. Furthermore, this supports the claim that processing the moving instance prediction leads to a better performance than directly performing panoptic segmentation.

Another way to enhance segmentation performance is by aggregating radar point clouds. However, the aggregation of point clouds can induce disadvantageous latencies, which is relevant when performing our approach on a vehicle in real-time. Additionally, our method performs on par in terms of mIoU with the best-performing semantic segmentation model STA-Net, which aggregates scans over 500 ms with input point clouds of 3072 points. In contrast, our single scans include an average number of points of 539 in the test set. The Radar Instance Transformer utilizes two previous scans to enrich the current scan, but only the single current scan is processed within the backbone. Therefore, we achieve similar results using five times fewer points and also include instance segmentation.

Gaussian Radar Transformer achieves a higher mIoU compared to Mask3D for the single-scan approaches. We assume that the optimized radar-specific architecture performs better on radar data. However, the powerful mask predictions lead to better performance in terms of PQ but come with a higher computational cost. Overall, both methods struggle to reliably segment instances in sparse and noisy radar point clouds. We enhance the PQ for all classes, leading to an improvement of more than 20 percentage points on the overall PQ.

#	Model	$r\,[m]$	$N^{\max}$	PQ	mIoU
[A]	Stratified Transformer [108]			55.0	42.6
[B]	Point Transformer [270]			75.8	63.1
[C]	KPConv [210]			81.0	69.6
[D]	Ours	5.0	24	**83.0**	**72.2**
[E]	Ours with varying parameters	6.0	24	82.3	71.3
[F]	Ours with varying parameters	6.0	34	82.4	71.5
[G]	Ours with varying parameters	5.0	34	82.7	72.0
[H]	Ours with varying parameters	3.0	24	81.7	70.4
[I]	Ours with varying parameters	5.0	6	78.7	66.4

Table 9.2: Ablation study on the radius transformer layer on the RadarScenes validation set in terms of PQ and mIoU scores. The radius transformer layer improves the feature extraction for the target processing of moving object predictions compared to other approaches.

9.3.3 Ablation Study on the Radius Transformer Layer

The second experiment evaluates our radius transformer layer and illustrates that our approach is capable of extracting valuable features from sparse point clouds, enhancing the overall performance. We replace the radius transformer layer with the point transformer layer used for radar signal processing, introduced in Chapter 7, the stratified transformer layer [108], and KPConv [210], illustrating the ablation results in Table 9.2. Additionally, we evaluate the performance of our approach based on the selection of the hyperparameters, namely the radius r and the maximum number of neighbors $N^{\max}$.

The gain obtained by our radius transformer layer is directly visible by the increase compared to the other transformer modules, enhancing the performance by more than 6 absolute percentage points compared to the point transformer layer. We argue that the kNN-based processing results in the interconnection of distant points, and the grid representation does not align with the instances, both harming the accuracy. As expected, the KPConv approach, Table 9.2 [C], leads to the best result using the ball query to determine the local areas. However, our attention-based processing enhances the PQ by 2 absolute percentage points compared to KPConv.

Furthermore, the comparison of the setup of $r = 5\,\mathrm{m}$ and $N^{\max} = 6$, Table 9.2 [I], which utilizes the same number of neighbors as the point transformer, underlines that the additional neighborhood restrictions by the radius are advantageous. We argue that, especially for small instances such as pedestrians but also cars or bikes that are far away, this leads to too many interconnections that harm the accuracy. However, for larger instances, we observe that increasing the

Model	PQ
Ground truth semantics	77.1
Ours w/o data augmentation	81.5
Ours	**83.0**

Table 9.3: Ablation study on the instance refinement on the RadarScenes validation set in terms of PQ and mIoU scores. The refinement is essential to enhance the prediction performance and correct the instance assignments. The best results are in bold.

number of neighbors helps to classify them reliably.

The performance decreases if we increase the maximum number of points from 24 to 34, Table 9.2 [D] and [G], respectively. We assume that too many interactions result in a mix of information during self-attention, which hinders the clear separation of the information. We also see that for a larger number of neighbors, the performance of smaller instances, such as pedestrians, drops from 65.2 % to 64.0 % for $N^{\max} = 24$ and $N^{\max} = 34$, respectively. Since close instances might still be included within the radius, the performance of small instances, especially, is affected. Therefore, the radius and the maximum number of parameters depend on the task and the segmentation of the specific instances. Additionally, the parameters may depend on the resolution of the radar sensor, which is an interesting starting point for future research to automatically determine the optimal settings. Overall, both parameters are essential to enhance radar-based panoptic segmentation.

9.3.4 Ablation Study on the Instance Refinement

The third experiment supports the claim that our method incorporates valuable instance refinement to enhance overall performance, as depicted in Table 9.3. The central parts that enable instance refinement are dedicated data augmentation for incorporating static points and point-wise processing. To verify the assumptions, we first evaluate the performance of the moving instance segmentation prediction in combination with the semantic ground truth labels. We, therefore, do not optimize the instance prediction of the Radar Instance Transformer to illustrate the limitation of keeping the instance assignments under perfect segmentation results. In the second step, we remove the data augmentation introduced in Section 9.1.4. The resulting method accounts for false predictions within the different object classes but cannot correct false static predictions because these errors are excluded in the training process. However, our instance refinement helps to optimize the instance prediction compared to our Radar Instance Transformer, even under the condition that our semantic predictions are not perfect. Furthermore,

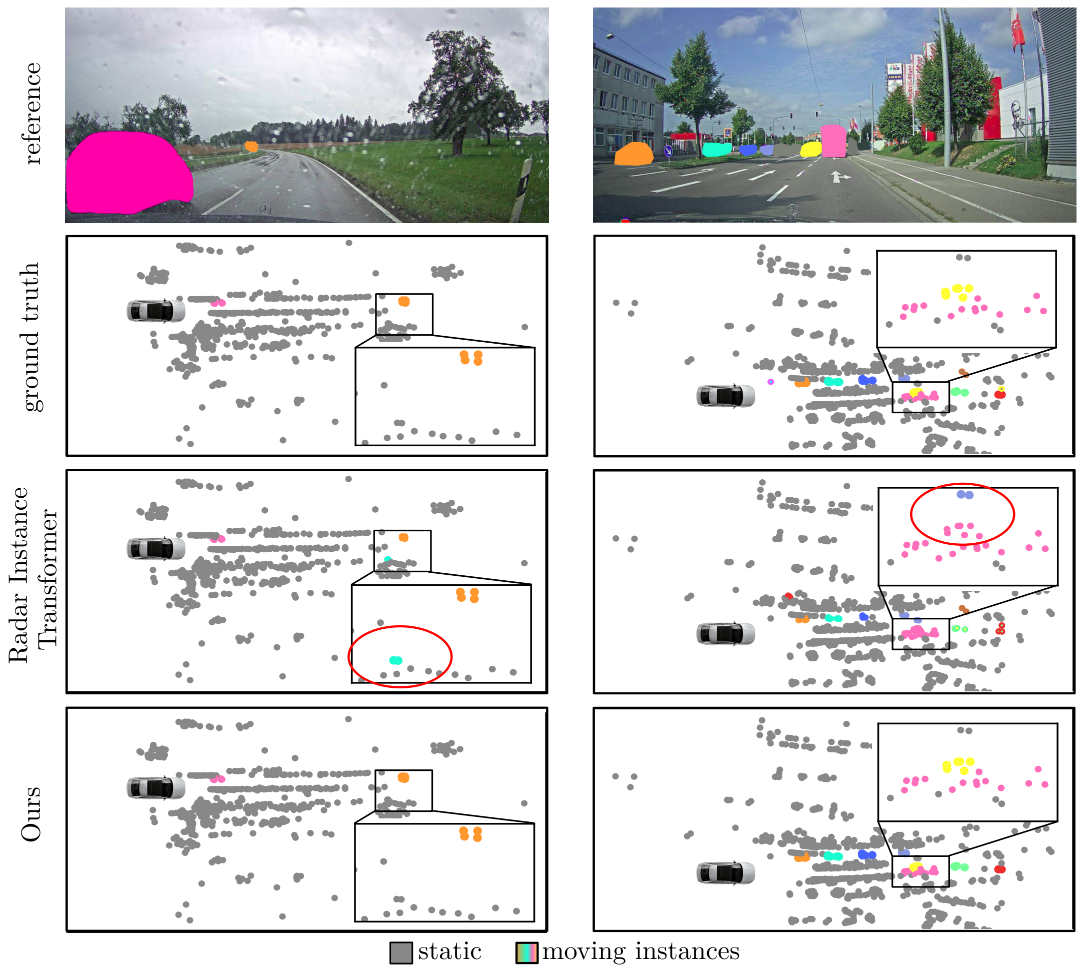


Figure 9.4: Qualitative results of our SemRaFiner and our Radar Instance Transformer on the RadarScenes dataset. The camera images are anonymized and shown for reference. The left column is from sequence 79 (fog), and the right is from sequence 107 (urban environment). Each color in the image of the predictions represents a different instance (static is grey).

the data augmentation enhances the panoptic quality by more than 2 absolute percentage points. Therefore, data augmentation and point-wise processing are both essential to improve performance, especially for the annotated moving objects in RadarScenes. Additionally, we performed an additional experiment and removed the consistency loss for the class consistency within instances, which resulted in a PQ of 82.9 %, leading to a small decrease in performance. Furthermore, our SemRaFiner works reliably under different and challenging driving scenarios, as illustrated in Figure 9.4. Our approach refines the moving instances under different and changing scenarios.

9.3.5 Runtime

Finally, we analyze the runtime and the number of parameters of the approaches and, in this way, support the claim that our approach runs fast enough to support

Model	parameters (M)	mean runtime (ms)
STA-Net [269]	7.4	34.9
RadarPNv1 [188]	1.8	278.1
Mask3D [187]	39.6	85.2
Gaussian Radar Transformer (Chapter 5)	8.4	24.0
Radar Instance Transformer (Chapter 7)	3.8	31.7
Ours + Radar Instance Transformer (Chapter 7)	4.5	42.1

Table 9.4: The evaluation of the mean runtime and the number of parameters of the models on an Nvidia RTX A6000 GPU based on 1000 randomly sampled point clouds of the RadarScenes dataset. The runtime and the parameters of STA-Net and RadarPNv1 are copied from Zhang et al. [269] using a different hardware setup.

online processing. We follow Section 5.3.5 and evaluated the runtime of our approach on an Nvidia RTX A6000 GPU on 1,000 real-world radar scans that we randomly selected from the RadarScenes validation set. The mean runtime for one radar scan is illustrated in Table 9.4. Our Gaussian Radar Transformer has the lowest runtime utilizing optimized farthest point sampling and kNN algorithm in C++ [271]. The multi-scan approaches increase the runtime compared to the Gaussian Radar Transformer, but the comparison is difficult since the reported results are evaluated on different hardware. Additionally, Mask3D uses four times more parameters to enhance the panoptic segmentation. The Radar Instance Transformer and our approach combined still have fewer parameters compared to STA-Net since our model only adds a small overhead. The combined runtime of 42.1 ms, equal to 24 Hz, is faster than the frame rate of 17 Hz of the radar sensors.

In summary, our evaluation suggests that our method provides competitive panoptic segmentation in sparse radar point clouds. The instance refinement and the utilization of moving instance predictions outperform state-of-the-art approaches. Thus, we support all our claims made in this chapter through our experimental evaluation.

9.4 Conclusion

Semantic scene understanding, including the prediction and classification of moving agents, is essential to enabling safe and robust driving behaviors of autonomous vehicles. In this chapter, we presented a novel approach for panoptic segmentation of sparse and noisy radar point clouds. Our method exploits the advantages of moving instance segmentation to refine predictions and enhance the overall system performance. Our method efficiently leverages sparse representations of moving points by limiting the self-attention mechanism within a local neighborhood. We overcome the limitations of grid representations and unbounded local regions in transformer networks to predict semantic classes. Furthermore, we exploit the self-attention mechanism within the optimized network to improve overall performance. Our point-wise processing enables us to refine instance prediction and correct assignment errors.

This allows us to enhance feature extraction by optimizing the training procedure and successfully correcting instance assignments. We implemented and evaluated our approach on the RadarScenes dataset and provided comparisons to other existing techniques. The experiments suggest that our proposed modules and training strategy are essential to achieve good performance on panoptic segmentation, supporting all claims made in this chapter. Overall, our approach outperforms the state-of-the-art methods, incorporating essential knowledge for radar-based scene understanding.

Chapter 10

Conclusion

Semantic scene understanding of the surroundings of a self-driving vehicle is crucial for operating autonomously in complex, real-world scenarios. Different environmental conditions, such as changing lighting and adverse weather, can significantly challenge perception systems. Especially, commonly used sensors such as cameras and LiDARs face limitations under rain, fog, and snow. In this thesis, we tackle these limitations by proposing novel approaches that leverage radar sensors, which work reliably under challenging conditions, including adverse weather. The interpretation of a scene depends on various aspects, making dedicated algorithms for the desired task crucial. Moreover, the applicability of the approaches in real-world scenarios depends on the runtime. For instance, collision avoidance relies on immediate feedback to ensure safety and reduce accidents.

Our work presented in Chapter 5 to Chapter 9 addresses several challenges and focuses on radar-based scene understanding by processing single-scan radar point clouds. Consequently, our approaches do not rely on processing aggregated data throughout the whole network, which induces latency and increases memory requirements. Since single-scan radar point clouds are sparse and noisy, we propose advanced algorithms that explore radar-specific properties such as the Doppler velocity to enable reliable perception algorithms.

We divide this thesis into two parts to examine the capabilities of radar-based perception approaches in detail. In Part I, we focus on the semantic understanding of radar point clouds by developing specialized learning-based approaches. Starting in Chapter 5, we present our novel approach for semantic segmentation of sparse and noisy radar point clouds. We introduce an advanced architecture and optimize the transformer layer by replacing the commonly used softmax function with a scaled Gaussian function to enable an individual weighting of sparse radar points. The dedicated modules improve feature extraction and are important for deriving valuable information and enhancing segmentation performance.

Furthermore, the processing of radar-specific properties such as the radar cross section and Doppler velocity support the identification of moving instances and help to classify nearby objects reliably. We compared our approach to several other state-of-the-art methods, showing superior semantic segmentation performance with comparable results to data aggregation methods.

Despite the good performance, the semantic segmentation does not include underrepresented classes, and the strong class imbalance within radar point clouds entails challenges for learning-based approaches. Instead of exploring multi-class semantic information, we focused on moving object segmentation in Chapter 6, which is especially suitable for radar signal processing because of the provided Doppler velocity. As a result, moving objects are directly identifiable within single radar scans. This is an advantage compared to LiDAR and camera processing, which often rely on temporal dependencies based on data aggregation. We incorporate the velocity information within our transformer-based network to enhance feature extraction. Moreover, we optimize the upsampling strategy within the encoder-decoder architecture to improve information exchange for sparse radar data. The data-driven approaches enhance performance compared to the threshold-based method for the Doppler velocity due to the noise in radar scans. Therefore, deep neural networks are essential to handle sparse and noisy radar scans and provide reliable predictions to leverage the full potential of downstream tasks such as path planning.

Moving object segmentation provides information about which parts of the environment are moving and which are static. For scene understanding, the knowledge of how many agents are present is crucial to operate safely in real-world environments. To account for the missing information, we propose our Radar Instance Transformer in Chapter 7. The novel approach incorporates temporal information in an effective way to overcome the limitations of passing aggregated scans through the whole network. Therefore, we enhance segmentation performance by enriching the single current radar scans and reducing the computations. We optimize the backbone architecture to keep the valuable information of individual detection intact. Furthermore, we leverage the self-attention mechanism to incorporate local and global instance knowledge, enhancing feature extraction and segmentation performance. We propose an attentive graph-based instance partitioning to reliably identify moving agents without relying on semantic information. As a result, we enhanced the accuracy for moving instance segmentation and addressed the challenging task of instance assignment within sparse radar point clouds. We established a new benchmark based on the RadarScenes dataset, which allows further comparisons with future work and is publicly available.

Our moving instance segmentation approach shows exceptional results and includes valuable information for various tasks. Therefore, we exploit the predictions in Part II to address additional tasks and enhance scene understanding. Besides identifying instances within individual scans, tracking and motion information of agents over time is essential for path planning and reliable collision avoidance of autonomous vehicles in real-world environments. In Chapter 8, we address this task and utilize geometric and appearance features to associate moving instances over time. We propose an advanced tracking algorithm based on the self-attention mechanism to enhance tracking quality. We address the difficulties of sparse point clouds and include the tracking of instances that comprise individual detections. Our experimental results demonstrate that our method surpasses existing approaches.

Semantic information about the corresponding class plays a crucial role in future trajectory prediction and pose estimation of individual instances, contributing to scene understanding. In particular, precise planning algorithms rely on this semantic understanding to model interactions between instances, as the maneuvers vary between classes. For example, the movement of cars is more restricted than that of pedestrians. Therefore, we address the panoptic segmentation of the radar point clouds in Chapter 9. In contrast to other approaches, we leverage the moving instance prediction to directly predict the semantic class for the individual points and refine the instance segmentation. We optimize the transformer layer to process the sparse moving object predictions to account for the changing density and extract valuable features. Furthermore, we introduce dedicated data augmentation techniques to account for false predictions and include the semantic information in the instance refinement. The combined approach adds little overhead to the moving instance segmentation network to enable online processing. We outperform other methods by a solid margin and achieve similar results to data aggregation segmentation methods, illustrating the strong performance of our transformer-based approaches.

Overall, this thesis presents novel approaches to enhance scene understanding for sparse and noisy radar point clouds in real-world automotive environments. We introduced modules to incorporate valuable radar sensor information and optimized network architecture to extract discriminative features. We addressed a variety of tasks contributing to scene understanding and presented solutions to several single-scan processing challenges. We evaluated and tested all our approaches on publicly available large-scale radar data and demonstrated that our transformer-based methods outperformed state-of-the-art. Our approaches work reliably under adverse weather, compensating for the limitations of other sensor modalities to enhance performance. The efficient processing and the availability of radar sensors within vehicles make these advancements practical and widely accessible. In summary, we took a step forward towards sensor redundancy to enhance scene understanding for autonomous vehicles and improve safety.

Chapter 11

Future Work

In this thesis, we explored radar-based scene understanding from sparse and noisy single-scan point clouds. We proposed various approaches to leverage the potential of radar sensors to enhance object identification and segmentation. While our methods achieved encouraging results, novel techniques and promising future works have the opportunity to improve performance and bring us closer to full autonomy. We primarily focused on the perception of moving instances and the differentiation of moving and static parts of the environment. This information is fundamental and contains the essential aspects for many downstream tasks, such as path planning. The differentiation and identification of the semantic classes of the static environment enhances scene understanding. This knowledge can help to precisely localize autonomous vehicles within complex scenes and improve context information for safe and efficient decision-making.

Furthermore, the separate classification of noise conceals multiple possibilities for improving the performance of radar-based perception. The differentiation can improve the overall segmentation performance, support free space detection, and reduce the memory requirements for consecutive tasks by removing noise. However, labeling the static environment and noise, in particular, is challenging. Therefore, transferring annotations from different sensor modalities can support the annotation process. Additionally, the development of so-called imaging radars simplifies the labeling efforts due to higher resolution, resulting in denser point clouds. Consequently, the time and the cost of the annotation process and the required synchronized camera and LiDAR data can be reduced. The processing of the dense point clouds brings benefits, such as detailed information about the environment. However, the architectures and modules need adjustment to handle denser point clouds in real time.

One central challenge in this thesis and also state-of-the-art approaches results from the identification of instances. The runtime of instance association based on clustering algorithms depends typically on the number of points or occupied voxels. The attentive graph-based clustering in Chapter 7 includes similar drawbacks since the construction of the graph and clustering correlate with the number of moving detections. Consequently, crowded scenes where latency requirements are high often have higher runtime. Masked-based predictions [135, 186] have a near-constant runtime. However, the number of masks needs to exceed the number of instances to reliably identify agents in the surroundings. As a result, the runtime is always high because the masks are determined for each scan and not adapted to the number of instances. Advanced algorithms with an optimized runtime independent of the scenes are desirable to overcome these limitations. The restriction of the instance assignment to local regions and performing the optimization in parallel can be interesting for future research. Additionally, the reduction of mask predictions without restricting accuracy is another solution.

Besides runtime optimization, the performance of the segmentation is often class-dependent and influenced by the distance to the sensor. The detection performance of the most vulnerable road users, including pedestrians and two-wheelers, is often worse than that of vehicles, see Part I. Data augmentation and specific data selection algorithms can enhance performance. However, there are still difficulties in the identification of small instances, which have to be addressed to enable safe driving functionalities within all scenarios. The effect is often more severe for sparse radar data, making the reliable segmentation of small and distant instances challenging since they usually comprise single detections. The information within the point cloud is scarce, and adequate feature extraction is key to enhancing the overall performance. The refinement introduced in Chapter 9 is a promising solution, and restricting the local area to extract the features improved the accuracy. However, the approach does not account for class-specific restrictions and distance-based adjustments due to the fact that distant radar data is sparser. Therefore, the adaptation of the specific parameters or data-driven optimization of these are interesting research directions. Furthermore, the radar parameters and sensor design influence the nature of the radar point cloud, as explained in Chapter 2. Since the optimization of the algorithms depends on specific input data, the adaptation to different sensors is challenging. Consequently, the simultaneous optimization of the radar parameters and the algorithm to explore the dependencies and automatically determine optimal settings can be crucial to leverage the full potential and provide valuable insights.

Despite optimization opportunities for the algorithms and sensor configurations, major challenges in radar signal processing and the whole perception stack are unknown objects. In Part I, we presented the approach to handle all moving objects within one central class, the moving class, which can include the long-tailed distribution important to handle real-world scenarios. However, scene understanding can benefit from the differentiation of individual instances within the static environment and include semantic information for unknown objects. Therefore, an extension focusing on open-world instances and semantic segmentation is an interesting research area. Despite the requirements for annotated point clouds mentioned earlier, open-world segmentation benefits from unsupervised approaches. Unsupervised learning approaches learn implicit patterns in data without using annotated data. This is especially useful for clustering data into different classes and identifying class-agnostic instance segments. For open-world segmentation, this includes several possibilities to derive instance clusters for static and moving detection as well as to reason about objects belonging to similar classes, which has already been addressed in LiDAR data processing [154]. Therefore, unsupervised approaches are interesting for exploring the variety of long-tailed class distributions and incorporating the knowledge within the perception algorithms. As a consequence, these approaches can provide additional knowledge for semantic refinement, introduced in Chapter 9, to enhance scene understanding.

Last but not least, multi-modal perception algorithms and information fusion can provide detailed information for autonomous vehicles to enhance scene understanding. The redundancy in perception algorithms can improve the robustness and reliability of autonomous vehicles in real-world environments. The sensor suites of autonomous vehicles include LiDARs, cameras, radars, and others, such as global navigation satellite systems and inertial measurement units, which can provide important information. Due to the variety of data and individual representations, fusing multi-modal sensor information remains challenging. Therefore, advanced fusion schemes that handle various data representations and consider the advantages and limitations of individual sensors need to be explored to enhance overall safety. Another open question is how to solve information fusion, especially when one sensor modality faces limitations such as under adverse weather, and how to make sure that we identify the degeneration of the data. We believe that exploring the advantages of individual sensors and the fusion of data is essential for future research to improve scene understanding and enable robust and safe autonomous mobility.

Bibliography

[1] N. Abraham and N.M. Khan. A Novel Focal Tversky Loss Function With Improved Attention U-Net for Lesion Segmentation. In *Proc. of the IEEE Intl. Symposium on Biomedical Imaging*, 2019.

[2] D. Adolfsson, M. Magnusson, A. Alhashimi, A.J. Lilienthal, and H. Andreasson. Lidar-Level Localization With Radar? The CFEAR Approach to Accurate, Fast, and Robust Large-Scale Radar Odometry in Diverse Environments. *IEEE Trans. on Robotics (TRO)*, 39(2):1476–1495, 2023.

[3] A. Agarwalla, X. Huang, J. Ziglar, F. Ferroni, L. Leal-Taixé, J. Hays, A. Osep, and D. Ramanan. Lidar Panoptic Segmentation and Tracking without Bells and Whistles. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2023.

[4] E.E. Aksoy, S. Baci, and S. Cavdar. Salsanet: Fast road and vehicle segmentation in lidar point clouds for autonomous driving. In *Proc. of the IEEE Vehicles Symposium (IV)*, 2020.

[5] A. Ambardar. *Analog and digital signal processing.* PWS Boston, MA, USA, 1995.

[6] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. on Signal Processing*, 50(2):174–188, 2002.

[7] M. Aygün, A. Osep, M. Weber, M. Maximov, C. Stachniss, J. Behley, and L. Leal-Taixé. 4d panoptic lidar segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[8] J.L. Ba, J.R. Kiros, and G.E. Hinton. Layer Normalization. *arXiv preprint*, arXiv:1607.06450, 2016.

[9] C.A. Balanis. *Advanced engineering electromagnetics.* John Wiley & Sons, 2012.

[10] B. Barboy, A. Lomes, and E. Perkalski. Cell-averaging CFAR for multiple-target situations. *IEEE Proceedings F: Communications, Radar and Signal Processing*, 133(2):176–186, 1986.

[11] D.K. Barton. *Radar equations for modern radar.* Artech House, 2013.

[12] S.A. Baur, D.J. Emmerichs, F. Moosmann, P. Pinggera, B. Ommer, and A. Geiger. SLIM: Self-Supervised LiDAR Scene Flow and Motion Segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[13] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[14] J. Behley, A. Milioto, and C. Stachniss. A Benchmark for LiDAR-Based Panoptic Segmentation Based on KITTI. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.

[15] J. Behley and C. Stachniss. Efficient Surfel-Based SLAM using 3D Laser Range Data in Urban Environments. In *Proc. of Robotics: Science and Systems (RSS)*, 2018.

[16] N. Benbarka, J. Schröder, and A. Zell. Score refinement for confidence-based 3D multi-object tracking. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[17] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade: Second edition*, pages 437–478. Springer, 2012.

[18] Y. Bengio and O. Delalleau. On the expressive power of deep architectures. In *Proc. of the Intl. Conf. on Algorithmic Learning Theory*, 2011.

[19] Y. Bengio, O. Delalleau, and N. Roux. The curse of highly variable functions for local kernel machines. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 18:107–114, 2005.

[20] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal on Machine Learning Research (JMLR)*, 13(2), 2012.

[21] M. Berman, A.R. Triki, and M.B. Blaschko. The Lovász-Softmax Loss: A Tractable Surrogate for the Optimization of the Intersection-Over-Union Measure in Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[22] I. Bilik. Comparative analysis of radar and lidar technologies for automotive applications. *IEEE Trans. on Intelligent Transportation Systems Magazine*, 15(1):244–269, 2023.

[23] C.M. Bishop. *Neural Networks for Pattern Recognition.* Oxford University Press, 1995.

[24] C.M. Bishop. *Pattern Recognition and Machine Learning*, volume 4. Springer, 2006.

[25] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proc. of the Intl. Conf. on Computational Statistics*, 2010.

[26] T. Brown, B. Mann, N. Ryder, M. Subbiah, J.D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language Models are Few-Shot Learners. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2020.

[27] R.J. Campello, D. Moulavi, and J. Sander. Density-based clustering based on hierarchical density estimates. In *Proc. of Pacific-Asia. Conf. on Knowledge Discovery and Data Mining*, 2013.

[28] C. Campos, R. Elvira, J.J.G. Rodríguez, J.M. Montiel, and J.D. Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Trans. on Robotics (TRO)*, 37(6):1874–1890, 2021.

[29] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2020.

[30] D. Casado Herraez, L. Chang, M. Zeller, L. Wiesmann, J. Behley, M. Heidingsfeld, and C. Stachniss. SPR: Single-Scan Radar Place Recognition. *IEEE Robotics and Automation Letters (RA-L)*, 9(10):9079–9086, 2024.

[31] D. Casado Herraez, M. Zeller, L. Chang, I. Vizzo, M. Heidingsfeld, and C. Stachniss. Radar-Only Odometry and Mapping for Autonomous Vehicles. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024.

[32] S. Chen, J. Fang, Q. Zhang, W. Liu, and X. Wang. Hierarchical Agregation for 3D Instance Segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[33] X. Chen, S. Li, B. Mersch, L. Wiesmann, J. Gall, J. Behley, and C. Stachniss. Moving Object Segmentation in 3D LiDAR Data: A Learning-based Approach Exploiting Sequential Data. *IEEE Robotics and Automation Letters (RA-L)*, 6(4):6529–6536, 2021.

[34] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[35] Y. Chen, J. Liu, X. Zhang, X. Qi, and J. Jia. Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[36] B. Cheng, I. Misra, A.G. Schwing, A. Kirillov, and R. Girdhar. Masked-attention Mask Transformer for Universal Image Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[37] C. Chi, P. Li, X. Chen, and X. Yang. Pua-mos: End-to-end point-wise uncertainty weighted aggregation for moving object segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[38] H.k. Chiu, J. Li, R. Ambruş, and J. Bohg. Probabilistic 3d multi-modal, multi-object tracking for autonomous driving. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2021.

[39] H.K. Chiu, A. Prioletti, J. Li, and J. Bohg. Probabilistic 3d multi-object tracking for autonomous driving. *arXiv preprint*, arXiv:2001.05673, 2020.

[40] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Empirical Methods in Natural Language Processing*, 2014.

[41] C. Choy, J. Gwak, and S. Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[42] D. Comaniciu and P. Meer. Mean Shift: A Robust Approach Toward Feature Space Analysis. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 24:603–619, 2002.

[43] T. Cortinhal, G. Tzelepis, and E. Erdal Aksoy. SalsaNext: Fast, uncertainty-aware semantic segmentation of LiDAR point clouds. In *Proc. of the Intl. Symp. on Visual Computing*, 2020.

[44] Y. Cui, Z. Fang, J. Shan, Z. Gu, and S. Zhou. 3d object tracking with transformer. *arXiv preprint*, arXiv:2110.14921, 2021.

[45] A. Danzer, T. Griebel, M. Bach, and K. Dietmayer. 2d car detection in radar data with pointnets. In *Proc. of the IEEE Intl. Conf. on Intelligent Transportation Systems (ITSC)*, pages 61–66, 2019.

[46] Y.N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2014.

[47] D.B. Reid. An algorithm for tracking multiple targets. *IEEE Trans. on Automatic Control*, 24(6):843–854, 1979.

[48] J. Deng, S. Shi, P. Li, W. Zhou, Y. Zhang, and H. Li. Voxel r-cnn: Towards high performance voxel-based 3d object detection. In *Proc. of the Conference on Advancements of Artificial Intelligence (AAAI)*, 2021.

[49] J. Devlin, M.W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint*, arXiv:1810.04805, 2018.

[50] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2021.

[51] A. Dubey, A. Santra, J. Fuchs, M. Lübke, R. Weigel, and F. Lurz. HARadNet: Anchor-free target detection for radar point clouds using hierarchical attention and multi-task learning. *Machine Learning with Applications (MLWA)*, 8:100275, 2022.

[52] L.H. Eriksson and B.O. As. A high performance automotive radar for automatic AICC. In *Proc. of the Intl. Radar Conference (RADAR)*, 1995.

[53] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proc. of the Conf. on Knowledge Discovery and Data Mining (KDD)*, 1996.

[54] M. Everingham, L. Van Gool, C. Williams, J. Winn, and A. Zisserman. The Pascal Visual Object Classes (VOC) Challenge. *Intl. Journal of Computer Vision (IJCV)*, 88(2):303–338, 2010.

[55] H. Fan, Y. Yang, and M. Kankanhalli. Point 4D Transformer Networks for Spatio-Temporal Modeling in Point Cloud Videos. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[56] R. Feger, H. Haderer, and A. Stelzer. Optimization of codes and weighting functions for binary phase-coded fmcw mimo radars. In *Proc. of the Intl. Conf. on Microwaves for Intelligent Mobility (ICMIM)*, 2016.

[57] P.F. Felzenszwalb and D.P. Huttenlocher. Efficient graph-based image segmentation. *Intl. Journal of Computer Vision*, 59:167–181, 2004.

[58] D. Frossard and R. Urtasun. End-To-End Learning of Multi-Sensor 3D Tracking by Detection. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[59] S. Gasperini, M.N. Mahani, A. Marcos-Ramiro, N. Navab, and F. Tombari. Panoster: End-To-End Panoptic Segmentation of LiDAR Point Clouds. *IEEE Robotics and Automation Letters (RA-L)*, 6(2):3216–3223, 2021.

[60] S. Giancola, J. Zarzar, and B. Ghanem. Leveraging shape completion for 3d siamese tracking. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[61] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. of the Intl. Conf. on Artificial Intelligence and Statistics*, 2010.

[62] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning.* MIT Press, 2016.

[63] B. Graham, M. Engelcke, and L. van der Maaten. 3D Semantic Segmentation with Submanifold Sparse Convolutional Networks. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[64] D. Grimes and T. Jones. Automotive radar: A brief review. *Proc. of the IEEE*, 62(6):804–822, 1974.

[65] M.H. Guo, J. Cai, Z.N. Liu, T.J. Mu, R.R. Martin, and S. Hu. PCT: Point Cloud Transformer. *Computational Visual Media*, 7(2):187–199, 2021.

[66] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep Learning for 3D Point Clouds: A Survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 43:4338–4364, 2020.

[67] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep learning for 3d point clouds: A survey. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 43(12):4338–4364, 2021.

[68] L. Han, T. Zheng, L. Xu, and L. Fang. OccuSeg: Occupancy-Aware 3D Instance Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[69] X.F. Han, Z.Y. He, J. Chen, and G.Q. Xiao. 3CROSSNet: Cross-level cross-scale cross-attention network for point cloud representation. *IEEE Robotics and Automation Letters (RA-L)*, 7(2):3718–3725, 2022.

[70] S. Hanson and L. Pratt. Comparing biases for minimal network construction with back-propagation. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 1988.

[71] J. Hastad. Almost optimal lower bounds for small depth circuits. In *Proc. of the ACM Symposium on Theory of Computing*, 1986.

[72] S. Haykin and C. Deng. Classification of radar clutter using neural networks. *IEEE Trans. on Neural Networks*, 2(6):589–600, 1991.

[73] C. He, R. Li, S. Li, and L. Zhang. Voxel set transformer: A set-to-set approach to 3d object detection from point clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[74] C. He, R. Li, G. Zhang, and L. Zhang. Scatterformer: Efficient voxel transformer with scattered linear attention. *arXiv preprint*, arXiv:2401.00912, 2024.

[75] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[76] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2017.

[77] D. Hendrycks and K. Gimpel. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *arXiv preprint:1606.08415*, 2016.

[78] G.E. Hinton, S. Osindero, and Y.W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[79] G.E. Hinton and R.R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[80] S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997.

[81] F. Hong, L. Kong, H. Zhou, X. Zhu, H. Li, and Z. Liu. Unified 3d and 4d panoptic segmentation via dynamic shifting networks. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 46(5):3480–3495, 2024.

[82] F. Hong, H. Zhou, X. Zhu, H. Li, and Z. Liu. Lidar-based panoptic segmentation via dynamic shifting network. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[83] F. Hong, H. Zhou, X. Zhu, H. Li, and Z. Liu. Lidar-based 4d panoptic segmentation via dynamic shifting network. *arXiv preprint*, arXiv:2203.07186, 2022.

[84] J. Hou, A. Dai, and M. Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[85] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[86] L. Hui, L. Wang, M. Cheng, J. Xie, and J. Yang. 3d siamese voxel-to-bev tracker for sparse point clouds. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2021.

[87] L. Hui, H. Yang, M. Cheng, J. Xie, and J. Yang. Pyramid point cloud transformer for large-scale place recognition. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[88] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, 2015.

[89] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv preprint*, arXiv:1502.03167, 2015.

[90] H. Jiang, F. Yan, J. Cai, J. Zheng, and J. Xiao. End-to-end 3D Point Cloud Instance Segmentation without Detection. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[91] R. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME – Journal of Basic Engineering*, 82:35–45, 1960.

[92] A. Kendall, Y. Gal, and R. Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[93] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2020.

[94] D. Kim, S. Woo, J. Lee, and I.S. Kweon. Video Panoptic Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[95] J. Kim, J. Woo, and Sunghoon. RVMOS: Range-View Moving Object Segmentation Leveraged by Semantic and Motion Features. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):8044–8051, 2022.

[96] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *Proc. of the Int. Conf. on Learning Representations (ICLR)*, 2015.

[97] A. Kolesnikov, A. Dosovitskiy, D. Weissenborn, G. Heigold, J. Uszkoreit, L. Beyer, M. Minderer, M. Dehghani, N. Houlsby, S. Gelly, T. Unterthiner, and X. Zhai. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2021.

[98] M. Kolodiazhnyi, A. Vorontsova, A. Konushin, and D. Rukhovich. Top-down beats bottom-up in 3d instance segmentation. In *Proc. of the IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2024.

[99] M. Kolodiazhnyi, A. Vorontsova, A. Konushin, and D. Rukhovich. Oneformer3d: One transformer for unified point cloud segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[100] L. Kong, Y. Liu, R. Chen, Y. Ma, X. Zhu, Y. Li, Y. Hou, Y. Qiao, and Z. Liu. Rethinking range view representation for lidar segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[101] L. Kreuzberg, I.E. Zulfikar, S. Mahadevan, F. Engelmann, and B. Leibe. 4D-StOP: Panoptic Segmentation of 4D LiDAR using Spatio-temporal Object

Proposal Generation and Aggregation. In *Proc. of the Europ. Conf. on Computer Vision Workshops*, 2022.

[102] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2012.

[103] A. Krogh and J. Hertz. A simple weight decay can improve generalization. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 1991.

[104] H. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[105] J. Kukacka, V. Golkov, and D. Cremers. Regularization for deep learning: A taxonomy. *arXiv preprint*, arXiv:1710.10686, 2017.

[106] S. Kullback. *Information Theory and Statistics.* Courier Corporation, 1997.

[107] A. Laddha, S. Gautam, G.P. Meyer, C. Vallespi-Gonzalez, and C.K. Wellington. Rv-fusenet: Range view based fusion of time-series lidar data for joint 3d object detection and motion forecasting. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[108] X. Lai, Y. Chen, F. Lu, J. Liu, and J. Jia. Spherical transformer for lidar-based 3d recognition. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[109] L. Landrieu and M. Simonovsky. Large-scale Point Cloud Semantic Segmentation with Superpoint Graphs. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[110] A. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. PointPillars: Fast Encoders for Object Detection From Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[111] K. Lang and G.E. Hinton. Dimensionality reduction and prior knowledge in e-set recognition. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 1989.

[112] Y. Le Cun, L.D. Jackel, B. Boser, J.S. Denker, H.P. Graf, I. Guyon, D. Henderson, R.E. Howard, and W. Hubbard. Handwritten digit recognition: Applications of neural net chips and automatic learning. In *IEEE Communications Magazine*, 1989.

[113] Y. LeCun, Y. Bengio, and G. Hinton. Deep Learning. *Nature*, 521:436–444, 2015.

[114] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, and L.D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[115] Y. LeCun. Une procedure d'apprentissage ponr reseau a seuil asymetrique. *Proceedings of Cognitiva 85*, 1985.

[116] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Säckinger, P. Simard, and V. Vapnik. Comparison of learning algorithms for handwritten digit recognition. In *Proc. of the Intl. Conf. on Artificial Neural Networks*, 1995.

[117] J. Lehner, A. Mitterecker, T. Adler, M. Hofmarcher, B. Nessler, and S. Hochreiter. Patch refinement–localized 3d object detection. *arXiv preprint*, arXiv:1910.04093, 2019.

[118] J. Li, X. He, Y. Wen, Y. Gao, X. Cheng, and D. Zhang. Panoptic-phnet: Towards real-time and high-precision lidar panoptic segmentation via clustering pseudo heatmap. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[119] R. Li, X. Li, C.W. Fu, D. Cohen-Or, and P.A. Heng. Pu-gan: a point cloud upsampling adversarial network. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[120] Y. Li, C. Liang, M. Lu, X. Hu, and Y. Wang. Cascaded kalman filter for target tracking in automotive radar. *The Journal of Engineering*, 2019(19):6264–6267, 2019.

[121] Y. Li, S. Si, G. Li, C.J. Hsieh, and S. Bengio. Learnable Fourier Features for Multi-dimensional Spatial Positional Encoding. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2021.

[122] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel. Gated graph sequence neural networks. *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2016.

[123] Z. Liang, Z. Li, S. Xu, M. Tan, and K. Jia. Instance segmentation in 3d scenes using semantic superpoint tree networks. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[124] V.E. Liong, T.N.T. Nguyen, S. Widjaja, D. Sharma, and Z.J. Chong. Amvnet: Assertion-based multi-view fusion network for lidar semantic segmentation. *arXiv preprint*, arXiv:2012.04934, 2020.

[125] J. Liu, W. Xiong, L. Bai, Y. Xia, T. Huang, W. Ouyang, and B. Zhu. Deep Instance Segmentation with Automotive Radar Detection Points. *IEEE Trans. on Intelligent Vehicles*, 8(1):84–94, 2023.

[126] X. Liu, C.R. Qi, and L.J. Guibas. FlowNet3D: Learning Scene Flow in 3D Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[127] Z. Liu, Z. Zhang, Y. Cao, H. Hu, and X. Tong. Group-free 3d object detection via transformers. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[128] Z. Liu, H. Tang, Y. Lin, and S. Han. Point-voxel cnn for efficient 3d deep learning. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 32, 2019.

[129] Z. Liu, X. Yang, H. Tang, S. Yang, and S. Han. Flatformer: Flattened window attention for efficient point cloud transformer. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[130] J. Lloyd and M. Korenberg. Improved radar range and velocity resolution using the fast orthogonal search. *IET Radar, Sonar & Navigation*, 17(8):1242–1247, 2023.

[131] J. Long, E. Shelhamer, and T. Darrell. Fully Convolutional Networks for Semantic Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[132] I. Loshchilov and F. Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2017.

[133] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2019.

[134] D. Lu, Q. Xie, M. Wei, K. Gao, L. Xu, and J. Li. Transformers in 3d point clouds: A survey. *arXiv preprint*, arXiv:2205.07417, 2022.

[135] R. Marcuzzi, L. Nunes, L. Wiesmann, E. Marks, J. Behley, and C. Stachniss. Mask4D: End-to-End Mask-Based 4D Panoptic Segmentation for LiDAR

Sequences. *IEEE Robotics and Automation Letters (RA-L)*, 8(11):7487–7494, 2023.

[136] R. Marcuzzi, L. Nunes, L. Wiesmann, I. Vizzo, J. Behley, and C. Stachniss. Contrastive instance association for 4d panoptic segmentation using sequences of 3d lidar scans. *IEEE Robotics and Automation Letters (RA-L)*, 7(2):1550–1557, 2022.

[137] H.H. Meinel and J. Dickmann. Automotive radar: From its origins to future directions. *Microwave Journal*, 56(9):24–28, 2013.

[138] H.Y. Meng, L. Gao, Y.K. Lai, and D. Manocha. Vv-net: Voxel vae net with group convolutions for point cloud segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[139] A. Merlo. Automotive radar for the prevention of collisions. *IEEE Trans. on Industrial Electronics and Control Instrumentation*, 1(1):1–6, 1964.

[140] B. Mersch, X. Chen, I. Vizzo, L. Nunes, J. Behley, and C. Stachniss. Receding Moving Object Segmentation in 3D LiDAR Data Using Sparse 4D Convolutions. *IEEE Robotics and Automation Letters (RA-L)*, 7(3):7503–7510, 2022.

[141] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss. RangeNet++: Fast and Accurate LiDAR Semantic Segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2019.

[142] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Trans. on Pattern Analalysis and Machine Intelligence (TPAMI)*, 44(7):3523–3542, 2022.

[143] T. Mitchell. *Machine Learning.* McGraw-Hill Science, 1997.

[144] J. Moody. The effective number of parameters: An analysis of generalization and regularization in nonlinear learning systems. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 1991.

[145] R. Moradi, R. Berangi, and B. Minaei. A survey of regularization strategies for deep models. *Artificial Intelligence Review*, 2020.

[146] H. Moravec. *Mind children: The future of robot and human intelligence.* Harvard University Press, 1988.

[147] N. Morgan and H. Bourlard. Generalization and parameter estimation in feedforward nets: Some experiments. *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 1989.

[148] K. Murphy. *Machine Learning – A Probabilistic Perspective.* MIT Press, 2012.

[149] L. Nagy and J. Lyon. An ultrashort pulse radar sensor for vehicular precollision obstacle detection. *IEEE Trans. on Vehicular Technology*, 24(4):41–45, 1975.

[150] V. Nair and G. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, 2010.

[151] M.E. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences*, 103(23):8577–8582, 2006.

[152] A. Nguyen and B. Le. 3D point cloud segmentation: A survey. In *IEEE Conf. on Robotics, Automation and Mechatronics*, 2013.

[153] F. Nobis, F. Fent, J. Betz, and M. Lienkamp. Kernel Point Convolution LSTM Networks for Radar Point Cloud Segmentation. *Applied Sciences*, 11:2599–2618, 2021.

[154] L. Nunes, X. Chen, R. Marcuzzi, A. Osep, L. Leal-Taixé, C. Stachniss, and J. Behley. Unsupervised class-agnostic instance segmentation of 3d lidar data for autonomous vehicles. *IEEE Robotics and Automation Letters (RA-L)*, 7(4):8713–8720, 2022.

[155] N. O'Mahony, S. Campbell, A. Carvalho, S. Harapanahalli, G.V. Hernandez, L. Krpalkova, D. Riordan, and J. Walsh. Deep learning vs. traditional computer vision. In *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC)*, 2020.

[156] A. Paigwar, O. Erkent, C. Wolf, and C. Laugier. Attentional PointNet for 3D-Object Detection in Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, 2019.

[157] A. Palffy, J. Dong, J.F.P. Kooij, and D.M. Gavrila. CNN Based Road User Detection Using the 3D Radar Cube. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):1263–1270, 2020.

[158] A. Palffy, E. Pool, S. Baratam, J.F.P. Kooij, and D.M. Gavrila. Multi-Class Road User Detection With 3+1D Radar in the View-of-Delft Dataset. *IEEE Robotics and Automation Letters (RA-L)*, 7(2):4961–4968, 2022.

[159] Z. Pang, Z. Li, and N. Wang. Simpletrack: Understanding and rethinking 3d multi-object tracking. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022.

[160] C. Park, Y. Jeong, M. Cho, and J. Park. Fast Point Transformer. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[161] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2019.

[162] L. Prechelt. Automatic early stopping using cross validation: quantifying the criteria. *Neural networks*, 11(4):761–767, 1998.

[163] G. Puy, A. Boulch, and R. Marlet. FLOT: Scene Flow on Point Clouds Guided by Optimal Transport. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2020.

[164] C.R. Qi, H. Su, K. Mo, and L.J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[165] C.R. Qi, X. Chen, O. Litany, and L.J. Guibas. Imvotenet: Boosting 3d object detection in point clouds with image votes. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[166] C. Qi, K. Yi, H. Su, and L.J. Guibas. PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2017.

[167] G. Qian, Y. Li, H. Peng, J. Mai, H. Hammoud, M. Elhoseiny, and B. Ghanem. PointNeXt: Revisiting PointNet++ with Improved Training and Scaling Strategies. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022.

[168] H. Qiu, B. Yu, and D. Tao. GFNet: Geometric Flow Network for 3D Point Cloud Semantic Segmentation. *Trans. on Machine Learning Research (TMLR)*, 2022.

[169] S. Qiu, S. Anwar, and N. Barnes. Pu-transformer: Point cloud upsampling transformer. In *Proc. of the Asian Conf. on Computer Vision (ACCV)*, 2022.

[170] P. Ramachandran, N. Parmar, A. Vaswani, I. Bello, A. Levskaya, and J. Shlens. Stand-alone self-attention in vision models. *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2019.

[171] R. Razani, R. Cheng, E. Li, E. Taghavi, Y. Ren, and L. Bingbing. GP-S3Net: Graph-Based Panoptic Sparse Semantic Segmentation Network. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[172] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2015.

[173] G. Riegler, A. Ulusoy, and A. Geiger. OctNet: Learning Deep 3D Representations at High Resolutions. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[174] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

[175] H. Rohling. Radar CFAR thresholding in clutter and multiple target situations. *IEEE Trans. on Aerospace and Electronic Systems*, 19(4):608–621, 1983.

[176] O. Ronneberger, P.Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Proc. of the Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015.

[177] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386, 1958.

[178] R. Roy and T. Kailath. ESPRIT-estimation of signal parameters via rotational invariance techniques. *IEEE Trans. on Acoustics, Speech, and Signal processing*, 37(7):984–995, 1989.

[179] D. Rumelhart, G. Hinton, and R. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[180] D. Rumelhart, G. Hinton, and R. Williams. *Learning Representations by Back-Propagating Errors*, pages 696–699. MIT press, 1988.

[181] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *Intl. Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[182] S. Santurkar, D. Tsipras, A. Ilyas, and A. Madry. How does batch normalization help optimization? *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2018.

[183] N. Scheiner, F. Kraus, N. Appenrodt, J. Dickmann, and B. Sick. Object detection for automotive radar point clouds–a comparison. *In Proc. of AI Perspectives*, 3, 2021.

[184] N. Scheiner, F. Kraus, F. Wei, B. Phan, F. Mannan, N. Appenrodt, W. Ritter, J. Dickmann, K. Dietmayer, B. Sick, and F. Heide. Seeing Around Street Corners: Non-Line-of-Sight Detection and Tracking In-the-Wild Using Doppler Radar. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[185] R. Schmidt. Multiple emitter location and signal parameter estimation. *IEEE Trans. on Antennas and Propagation*, 34(3):276–280, 1986.

[186] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe. Mask3D for 3D Semantic Instance Segmentation. *arXiv:2210.03105*, 2022.

[187] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[188] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler. Semantic Segmentation on Radar Point Clouds. In *Proc. of the Intl. Conf. on Information Fusion*, 2018.

[189] O. Schumann, M. Hahn, N. Scheiner, F. Weishaupt, J.F. Tilly, J. Dickmann, and C. Wöhler. RadarScenes: A real-world radar point cloud data set for automotive applications. In *Proc. of the Intl. Conf. on Information Fusion*, 2021.

[190] O. Schumann, J. Lombacher, M. Hahn, C. Wöhler, and J. Dickmann. Scene Understanding With Automotive Radar. *IEEE Trans. on Intelligent Vehicles*, 5(2):188–203, 2019.

[191] C.E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

[192] H. Shi, G. Lin, H. Wang, T.Y. Hung, and Z. Wang. SpSequenceNet: Semantic Segmentation Network on 4D Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[193] S. Shi, X. Wang, and H. Li. PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[194] W. Shi and R. Rajkumar. Point-gnn: Graph neural network for 3d object detection in a point cloud. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[195] X. Shi, Z. Chen, H. Wang, D.Y. Yeung, W.K. Wong, and W.C. Woo. Convolutional LSTM Network: A Machine Learning Approach for Precipitation Nowcasting. In *Proc. of the Advances in Neural Information Processing Systems (NIPS)*, 2015.

[196] Y. Shi and K. Ma. SAFIT: Segmentation-Aware Scene Flow with Improved Transformer. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[197] S. Shin, K. Zhou, M. Vankadari, A. Markham, and N. Trigoni. Spherical Mask: Coarse-to-Fine 3D Point Cloud Instance Segmentation with Spherical Representation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[198] D.A. Shnidman. Radar detection in clutter. *IEEE Trans. on Aerospace and Electronic Systems*, 41(3):1056–1067, 2005.

[199] C. Shorten and T.M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

[200] M. Simonovsky and N. Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[201] K. Sirohi, R. Mohan, D. Büscher, W. Burgard, and A. Valada. EfficientLPS: Efficient LiDAR Panoptic Segmentation. *IEEE Trans. on Robotics (TRO)*, 38(3):1894–1914, 2021.

[202] M. Stolz, M. Wolf, F. Meinl, M. Kunert, and W. Menzel. A new antenna array and signal processing concept for an automotive 4d radar. In *Proc. of the IEEE European Radar Conference (EuRAD)*, 2018.

[203] S. Su, J. Xu, H. Wang, Z. Miao, X. Zhan, D. Hao, and X. Li. Pups: Point cloud unified panoptic segmentation. In *Proc. of the Conference on Advancements of Artificial Intelligence (AAAI)*, 2023.

[204] J. Sun, Y. Dai, X. Zhang, J. Xu, R. Ai, W. Gu, and X. Chen. Efficient Spatial-Temporal Information Fusion for LiDAR-Based 3D Moving Object Segmentation. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[205] J. Sun, C. Qing, J. Tan, and X. Xu. Superpoint transformer for 3d scene instance segmentation. In *Proc. of the Conference on Advancements of Artificial Intelligence (AAAI)*, 2023.

[206] P. Sun, M. Tan, W. Wang, C. Liu, F. Xia, Z. Leng, and D. Anguelov. Swformer: Sparse window transformer for 3d object detection in point clouds. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022.

[207] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han. Searching Efficient 3D Architectures with Sparse Point-Voxel Convolution. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2020.

[208] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese. SEGCloud: Semantic Segmentation of 3D Point Clouds. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, 2017.

[209] G. Te, W. Hu, A. Zheng, and Z. Guo. Rgcnn: Regularized graph cnn for point cloud segmentation. In *Proceedings of the ACM Intl. Conference on Multimedia*, 2018.

[210] H. Thomas, C. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas. KPConv: Flexible and Deformable Convolution for Point Clouds. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[211] G. Vasilescu. *Electronic noise and interfering signals: principles and applications*. Springer Science & Business Media, 2006.

[212] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2017.

[213] P. Velicković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio. Graph Attention Networks. *Proc. of the Intl. Conf. on Learning Representations (ICLR)*, 2018.

[214] T. Vu, K. Kim, T.M. Luu, T. Nguyen, and C.D. Yoo. SoftGroup for 3D Instance Segmentation on Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[215] H. Wang, C. Shi, S. Shi, M. Lei, S. Wang, D. He, B. Schiele, and L. Wang. Dsvt: Dynamic sparse voxel transformer with rotated sets. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[216] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao. Deep high-resolution representation learning for visual recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 43(10):3349–3364, 2020.

[217] N. Wang, C. Shi, R. Guo, H. Lu, Z. Zheng, and X. Chen. Insmos: Instance-aware moving object segmentation in lidar data. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2023.

[218] S. Wang, Y. Sun, C. Liu, and M. Liu. Pointtracknet: An end-to-end network for 3-d object detection and tracking from point clouds. *IEEE Robotics and Automation Letters (RA-L)*, 5(2):3206–3212, 2020.

[219] X. Wang, S. Liu, X. Shen, C. Shen, and J. Jia. Associatively Segmenting Instances and Semantics in Point Clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[220] Y. Wang, Y. Sun, Z. Liu, S.E. Sarma, M.M. Bronstein, and J.M. Solomon. Dynamic graph cnn for learning on point clouds. *ACM Trans. on Graphics (TOG)*, 38(5):1–12, 2019.

[221] Z. Wang, S. Li, H. Howard-Jenkins, V. Prisacariu, and M. Chen. Flownet3d++: Geometric losses for deep scene flow estimation. In *Proc. of the IEEE Winter Conf. on Applications of Computer Vision (WACV)*, 2020.

[222] Z. Wang and F. Lu. Voxsegnet: Volumetric cnns for semantic part segmentation of 3d shapes. *Proc. of the IEEE Trans. on Visualization and Computer Graphics*, 26(9):2919–2930, 2019.

[223] X. Weng, J. Wang, D. Held, and K. Kitani. 3d multi-object tracking: A baseline and new evaluation metrics. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2020.

[224] D. Williams. Millimetre wave radars for automotive applications. In *Proc. of IEEE MTT-S Microwave Symposium Digest*, 1992.

[225] V. Winkler. Range Doppler detection for automotive FMCW radars. In *Proc. of the IEEE European Radar Conference (EuRAD)*, 2007.

[226] H. Winner, B. Danner, and J. Steinle. Adaptive cruise control. *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*, pages 478–521, 2009.

[227] P.M. Woodward. *Probability and information theory, with applications to radar: international series of monographs on electronics and instrumentation*, volume 3. Elsevier, 2014.

[228] World Health Organization. Global status report on road safety 2018, 2018.

[229] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer. SqueezeSegV2: Improved Model Structure and Unsupervised Domain Adaptation for Road-Object Segmentation from a LiDAR Point Cloud. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2019.

[230] B. Wu, A. Wan, X. Yue, and K. Keutzer. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2018.

[231] H. Wu, W. Han, C. Wen, X. Li, and C. Wang. 3d multi-object tracking in point clouds based on prediction confidence-guided data association. *IEEE Trans. on Intelligent Transportation Systems (ITS)*, 23(6):5668–5677, 2021.

[232] K. Wu, H. Peng, M. Chen, J. Fu, and H. Chao. Rethinking and improving relative position encoding for vision transformer. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[233] P. Wu, S. Chen, and D.N. Metaxas. MotionNet: Joint Perception and Motion Prediction for Autonomous Driving Based on Bird's Eye View Maps. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[234] W. Wu, Z. Qi, and L. Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[235] X. Wu, L. Jiang, P.S. Wang, Z. Liu, X. Liu, Y. Qiao, W. Ouyang, T. He, and H. Zhao. Point Transformer V3: Simpler, Faster, Stronger. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[236] X. Wu, Y. Lao, L. Jiang, X. Liu, and H. Zhao. Point transformer v2: Grouped vector attention and partition-based pooling. *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2022.

[237] P. Xiang, X. Wen, Y.S. Liu, H. Zhang, Y. Fang, and Z. Han. Retro-fpn: Retrospective feature pyramid network for point cloud semantic segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023.

[238] T. Xiao, P. Dollar, M. Singh, E. Mintun, T. Darrell, and R. Girshick. Early Convolutions help Transformers see Better. In *Proc. of the Conf. on Neural Information Processing Systems (NeurIPS)*, 2021.

[239] Z. Xiao, W. Zhang, T. Wang, C.C. Loy, D. Lin, and J. Pang. Position-guided point cloud panoptic segmentation transformer. *arXiv preprint*, arXiv:2303.13509, 2023.

[240] C. Xie, Y. Xiang, A. Mousavian, and D. Fox. Unseen object instance segmentation for robotic environments. *IEEE Trans. on Robotics (TRO)*, 37(5):1343–1359, 2021.

[241] S. Xie, S. Liu, Z. Chen, and Z. Tu. Attentional shapecontextnet for point cloud recognition. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[242] L. Xin, L. Jianhui, J. Li, W. Liwei, Z. Hengshuang, L. Shu, Q. Xiaojuan, and J. Jiaya. Stratified Transformer for 3D Point Cloud Segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[243] R. Xiong, Y. Yang, D. He, K. Zheng, S. Zheng, C. Xing, H. Zhang, Y. Lan, L. Wang, and T. Liu. On Layer Normalization in the Transformer Architecture. In *Proc. of the Intl. Conf. on Machine Learning (ICML)*, 2020.

[244] J. Xu, R. Zhang, J. Dou, Y. Zhu, J. Sun, and S. Pu. Rpvnet: A deep and efficient range-point-voxel fusion network for lidar point cloud segmentation. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[245] M. Xu, R. Ding, H. Zhao, and X. Qi. Paconv: Position adaptive convolution with dynamic kernel assembling on point clouds. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[246] T.X. Xu, Y.C. Guo, Y.K. Lai, and S.H. Zhang. CXTrack: Improving 3D point cloud tracking with contextual information. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[247] B. Yang, W. Luo, and R. Urtasun. PIXOR: Real-Time 3D Object Detection From Point Clouds. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[248] B. Yang, S. Wang, A. Markham, and N. Trigoni. Robust attentional aggregation of deep feature sets for multi-view 3D reconstruction. *Intl. Journal of Computer Vision (IJCV)*, 128(1):53–73, 2020.

[249] J. Yang, Q. Zhang, B. Ni, L. Li, J. Liu, M. Zhou, and Q. Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[250] Z. Yang, Y. Sun, S. Liu, X. Shen, and J. Jia. STD: Sparse-to-Dense 3D Object Detector for Point Cloud. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2019.

[251] L. Yi, W. Zhao, H. Wang, M. Sung, and L. Guibas. GSPN: Generative Shape Proposal Network for 3D Instance Segmentation in Point Cloud. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[252] T. Yin, X. Zhou, and P. Krähenbühl. Center-Based 3D Object Detection and Tracking. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[253] X. Yu, L. Tang, Y. Rao, T. Huang, J. Zhou, and J. Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[254] L. Yuan, Y. Chen, T. Wang, W. Yu, Y. Shi, Z.H. Jiang, F.E. Tay, J. Feng, and S. Yan. Tokens-to-Token ViT: Training Vision Transformers From Scratch on ImageNet. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[255] M. Zeller, J. Behley, M. Heidingsfeld, and C. Stachniss. Gaussian Radar Transformer for Semantic Segmentation in Noisy Radar Data. *IEEE Robotics and Automation Letters (RA-L)*, 8(1):344–351, 2023.

[256] M. Zeller, D. Casado Herraez, J. Behley, M. Heidingsfeld, and C. Stachniss. Radar Tracker: Moving Instance Tracking in Sparse and Noisy Radar Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2024.

[257] M. Zeller, D.C. Herraez, , J. Behley, M. Heidingsfeld, and C. Stachniss. Semrafiner: Panoptic segmentation in sparse and noisy radar point clouds. *IEEE Robotics and Automation Letters (RA-L)*, 10(2):923–930, 2025.

[258] M. Zeller, V.S. Sandhu, B. Mersch, J. Behley, M. Heidingsfeld, and C. Stachniss. Radar Velocity Transformer: Single-scan Moving Object Segmentation in Noisy Radar Point Clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2023.

[259] M. Zeller, V.S. Sandhu, B. Mersch, J. Behley, M. Heidingsfeld, and C. Stachniss. Radar instance transformer: Reliable moving instance segmentation in sparse radar point clouds. *IEEE Trans. on Robotics (TRO)*, 40:2357–2372, 2024.

[260] A. Zhang, Z.C. Lipton, M. Li, and A.J. Smola. Dive into Deep Learning. *arXiv preprint*, arXiv:2106.11342, 2021.

[261] C. Zhang, C. Zhang, Y. Guo, L. Chen, and M. Happold. Motiontrack: End-to-end transformer-based multi-object tracking with lidar-camera fusion. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition Workshops*, 2023.

[262] C. Zhang, H. Wan, X. Shen, and Z. Wu. PVT: Point-voxel transformer for point cloud learning. *Intl. Journal of Intelligent Systems*, 37(12):11985–12008, 2022.

[263] C. Zhang, W. Luo, and R. Urtasun. Efficient convolutions for real-time semantic segmentation of 3d point clouds. In *Proc. of the Intl. Conf. on 3D Vision (3DV)*, 2018.

[264] F. Zhang, G. Sun, Y. Zhou, B. Gao, and S. Pan. Towards high-resolution imaging with photonics-based time division multiplexing mimo radar. *IEEE Journal of Selected Topics in Quantum Electronics*, 28(5):1–10, 2022.

[265] H. Zhang, C. Wang, S. Tian, B. Lu, L. Zhang, X. Ning, and X. Bai. Deep learning-based 3d point cloud classification: A systematic survey and outlook. *Displays*, 79:102456, 2023.

[266] J. Zhang and S. Singh. LOAM: Lidar Odometry and Mapping in Real-time. In *Proc. of Robotics: Science and Systems (RSS)*, 2014.

[267] Y. Zhang, Z. Zhou, P. David, X. Yue, Z. Xi, B. Gong, and H. Foroosh. Polarnet: An improved grid representation for online lidar point clouds semantic segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[268] Y. Zhang, A. Carballo, H. Yang, and K. Takeda. Perception and sensing for autonomous vehicles under adverse weather conditions: A survey. *ISPRS Journal of Photogrammetry and Remote Sensing (JPRS)*, 196:146–177, 2023.

[269] Z. Zhang, J. Liu, and G. Jiang. Spatial and Temporal Awareness Network for Semantic Segmentation on Automotive Radar Point Cloud. *IEEE Trans. on Intelligent Vehicles*, 9(2):3520–3530, 2024.

[270] H. Zhao, J. Jia, and V. Koltun. Exploring self-attention for image recognition. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[271] H. Zhao, L. Jiang, J. Jia, P.H. Torr, and V. Koltun. Point Transformer. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2021.

[272] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. Wu. Multi-Agent Tensor Fusion for Contextual Trajectory Prediction. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[273] C. Zhou, Z. Luo, Y. Luo, T. Liu, L. Pan, Z. Cai, H. Zhao, and S. Lu. Pttr: Relational 3d point cloud object tracking with transformer. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[274] H. Zhou, H. Chen, Y. Zhang, M. Wei, H. Xie, J. Wang, T. Lu, J. Qin, and X.P. Zhang. Refine-net: Normal refinement neural network for noisy point clouds. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 45(1):946–963, 2022.

[275] X. Zhou, D. Wang, and P. Krähenbühl. Objects as Points. *arXiv preprint*, arXiv:1904.07850, 2019.

[276] Y. Zhou and O. Tuzel. VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[277] Z. Zhou, Y. Zhang, and H. Foroosh. Panoptic-polarnet: Proposal-free lidar point cloud panoptic segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[278] Z. Zhou, X. Zhao, Y. Wang, P. Wang, and H. Foroosh. Centerformer: Center-based transformer for 3d object detection. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, 2022.

[279] M. Zhu, M. Ghaffari, W.A. Clark, and H. Peng. E2pn: Efficient se (3)-equivariant point network. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[280] M. Zhu, S. Han, H. Cai, S. Borse, M. Ghaffari, and F. Porikli. 4d panoptic segmentation as invariant and equivariant field prediction. In *Proc. of the IEEE/CVF Intl. Conf. on Computer Vision (ICCV)*, 2023.

[281] M. Zhu, S. Han, H. Cai, S. Borse, M.G. Jadidi, and F. Porikli. 4D Panoptic Segmentation as Invariant and Equivariant Field Prediction. *arXiv preprint*, arXiv:2303.15651, 2023.

[282] Q. Zhu, L. Fan, and N. Weng. Advancements in point cloud data augmentation for deep learning: A survey. *arXiv preprint*, arXiv:2308.12113, 2023.

[283] X. Zhu, H. Zhou, T. Wang, F. Hong, Y. Ma, W. Li, H. Li, and D. Lin. Cylindrical and asymmetrical 3d convolution networks for lidar segmentation. In *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[284] M.D. Zoltowski, M. Haardt, and C.P. Mathews. Closed-form 2-D angle estimation with rectangular arrays in element space or beamspace via unitary ESPRIT. *IEEE Trans. on Signal Processing*, 44(2):316–328, 1996.

List of Figures

List of Tables