UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2025-017
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 13/02/2025 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

Faisal HAWLADER
Born on 5 January 1993 in Madaripur, (Bangladesh)

INFRASTRUCTURE ASSISTED COOPERATIVE AND DISTRIBUTED PERCEPTION STRATEGIES FOR CONNECTED VEHICLES

Dissertation Defence Committee

Dr Raphaël FRANK, Dissertation Supervisor
Assistant Professor, Université du Luxembourg

Dr Francesco VITI, Chairman
Associate Professor, Université du Luxembourg

Dr Ion TURCANU, Vice Chairman
Senior Researcher, Luxembourg Institute of Science and Technology, Luxembourg

Dr Christoph SOMMER
Professor, TUD Dresden University of Technology, Germany

Dr Ana AGUIAR
Associate Professor, University of Porto, Portugal

UNIVERSITY OF LUXEMBOURG

Abstract

Interdisciplinary Centre for Security, Reliability and Trust (SnT)

UBIX Research Group

Infrastructure Assisted Cooperative and Distributed Perception Strategies for Connected Vehicles

by Faisal HAWLADER

The research presented in this dissertation addressed the challenge of efficiently processing sensor data across vehicles, edge, and cloud platforms to support resource-intensive perception tasks in autonomous driving.

We investigated distributed processing of perception data from three perspectives to enhance perception accuracy, optimize network and computing resources, and meet real-time latency constraints: (1) offloading computationally intensive tasks to edge or cloud platforms for greater processing capacity and reduced on-board load, (2) evaluating the impact of compression techniques such as H.265 and JPEG on perception quality and transmission latency, and (3) exploring feature-vector transmission as an alternative to raw or compressed data to reduce bandwidth usage while maintaining perception accuracy and minimizing end-to-end delay.

To evaluate performance under realistic conditions, the study utilized both simulation-based assessments and real-world experiments. Using Vehicle-to-Everything (V2X) communication technologies, we analyzed compression and feature-sharing strategies for seamless data exchange between vehicles and infrastructure. Our findings demonstrated that the proposed cooperative and distributed perception strategies significantly improved detection accuracy and reduced processing delays compared to standalone on-board systems.

Acknowledgments

First, I would like to express my deepest gratitude to my PhD supervisor, Prof. Raphaël Frank, for his guidance and mentorship throughout my academic journey. Over the past four years, his insightful feedback and unwavering support have been instrumental in improving the outcomes of various projects we worked on. I am particularly grateful to him for allowing me the freedom to explore topics that genuinely interest me, while providing steady guidance to navigate challenges. This not only led to high-quality published work but also earned a Best Paper Award at WONS 2023. I sincerely thank him for guiding me in crafting a research proposal that secured FNR funding, which was another key milestone in my early career.

I would also like to express my heartfelt gratitude to Prof. Francesco Viti and Prof. Christoph Sommer for reviewing my progress over the years and offering detailed, constructive feedback. I extend my sincere appreciation to my colleagues and collaborators, whose insights and encouragement made this journey both productive and enjoyable. Special thanks to Dr. François Robinet, whose technical expertise was critical in overcoming numerous challenges.

I am thankful to many of my co-workers from the UBIX and SEDAN teams for the enjoyable lunchtime discussions, which often blended research topics with insightful conversations about life. I am also deeply grateful to Valérie Gregoire for her continued support in planning conference trips.

I cannot thank my friends and family enough for their unwavering support and belief in me. I am especially grateful to my father for shaping me into the person I am today. His constant support and faith made this opportunity possible. A heartfelt thanks to the Bangladeshi community in Luxembourg for their warm gatherings, which made me feel at home and eased the distance from my family. These events brought comfort and renewed my spirit for the week ahead.

This dissertation is a reflection of the collective contributions of those who inspired, supported, and guided me, including many teachers from school, college, and university. To all of you, I owe my deepest thanks.

Lastly, I acknowledge the funding and resources provided by the Luxembourg National Research Fund (FNR), without which this research would not have been possible.

The research presented in this dissertation was carried out with the support of the Luxembourg National Research Fund (FNR) through the AFR grant (agreement No. 17020780).

Contents

Acronyms

BEV Bird's-eye view. 17, 103, 105, 106, 114

C-V2X Cellular Vehicle-to-Everything. 7, 17, 21, 22, 25, 55, 57, 59, 61, 63, 118, 119

CAV Connected and Autonomous Vehicle. 5–7, 15–20, 30, 33, 35, 42–45, 52, 87, 90, 101, 117, 119–121, 123, 125–127

CN Core Network. 62, 63

CPM Cooperative Perception Message. 18, 21, 24, 45, 89, 90, 93, 95, 102–105, 107–110

CPS Cooperative Perception System. 31, 32, 40, 103, 108

DNN Deep Neural Network. 17, 19

ETSI European Telecommunications Standards Institute. 45, 90, 103

FoV Field of View. 32–35, 40

MEC Multi-access Edge Computing. 56, 57, 59, 61, 62

NDS NuScenes Detection Score. 106, 108, 109

RAN Radio Access Network. 61, 62

UE user equipment. 62, 63

V2I Vehicle-to-Infrastructure. 57, 63

V2V vehicle-to-vehicle. 49, 51, 52, 118

V2X Vehicle-to-Everything. 3, 5, 17–19, 23, 25, 42–44, 59, 61, 87, 89, 101, 127

List of Figures

List of Tables

Part I

Introduction

Chapter 1

Introduction

1.1 Introduction

Despite significant progress in autonomous driving technologies [120], achieving fully autonomous systems capable of navigating complex [19], dynamic [6], and uncertain environments remains a formidable challenge [12]. At the core of these challenges lies the perception system [82]. It is responsible for sensing, interpreting, and understanding the surrounding environment. Robust perception systems are essential for detecting obstacles, interpreting road conditions, and enabling safe navigation [69]. However, the standalone capabilities of individual vehicles are often insufficient. These limitations are particularly evident in scenarios with occlusions, poor visibility, or restricted sensing ranges [5].

Cooperative perception has emerged as a transformative solution to address these shortcomings [133]. By leveraging Vehicle-to-Everything (V2X) communication [68, 70], cooperative perception extends the sensing and interpretation capabilities of individual vehicles [19]. It facilitates data sharing among vehicles, and infrastructure, creating a collaborative network of perception systems [80]. This approach enhances situational awareness, enabling autonomous vehicles to detect and respond to hazards beyond their direct line of sight [16]. As a result, cooperative perception significantly improves safety and efficiency in dynamic traffic environments [111]. However, cooperative perception introduces several challenges [171]. These include managing the latency and bandwidth constraints of real-time data exchange [57], minimizing network resource consumption [51], and preserving critical details within shared data to maintain detection accuracy [119].

These challenges require innovative strategies that can seamlessly balance real-time processing demands with resource constraints. Distributed processing frameworks combining edge and cloud computing have gained prominence [94, 50]. Edge computing supports low-latency, localized processing of critical perception tasks [124]. It reduces reliance on onboard hardware by enabling real-time decision making at the network edge. In contrast, cloud computing offers scalable resources for computationally intensive tasks, such as high-resolution data aggregation and deep learning inference [104]. By strategically balancing these complementary approaches, distributed processing frameworks effectively address trade-offs between latency, bandwidth, and computational efficiency. This combination paves the way for robust and scalable cooperative perception systems [116].

This dissertation advances the state-of-the-art in cooperative perception for autonomous driving. It comprehensively explores the following aspects:

- **Simulation Framework Development:** Designed and implemented a comprehensive simulation framework to rigorously evaluate cooperative perception systems under controlled conditions [53, 54, 55]. The framework facilitates systematic analysis of critical metrics, such as latency, detection accuracy, communication reliability, and resource utilization. It serves as a foundational tool for testing and benchmarking innovative cooperative perception methodologies.
- **Distributed Processing Strategies:** Investigated advanced hybrid edge-cloud offloading techniques to optimize the performance of perception systems [57, 59]. These strategies effectively balance latency, bandwidth, and computational demands by partitioning tasks such as feature extraction and detection across edge and cloud resources. Lightweight feature sharing techniques, including clipping and quantization, were introduced to reduce bandwidth requirements [60]. These techniques ensure efficient data transmission while preserving detection accuracy. This ensures the system meets the real-time requirements of dynamic driving environments.
- **Communication Technologies:** Evaluated the performance of communication standards, including ITS-G5 and C-V2X, in supporting cooperative perception [39, 56, 60]. The analysis focused on trade-offs between latency, bandwidth usage, and perception quality. It provided actionable guidelines for selecting suitable communication technologies. Additionally, compression standards such as H.265 and JPEG were assessed to determine their impact on data transmission efficiency and detection accuracy [57].
- **Experimental Validation:** Conducted extensive real-world trials to validate the proposed frameworks and strategies [56, 60]. These experiments assessed the feasibility, scalability, and robustness of cooperative perception systems. They demonstrated the practical applicability and operational effectiveness of the proposed approaches in diverse deployment scenarios.
- **Release of Driving Datasets:** Curated and publicly released comprehensive datasets encompassing both real-world and simulation-based driving scenarios [59, 57]. These datasets include diverse environmental conditions, sensor modalities, and driving contexts. They serve as benchmarks for cooperative perception research. Their release aims to advance the field and foster collaboration within the autonomous driving research community.

Together, these contributions bridge the gap between theoretical advancements and practical deployments. By addressing challenges in computational efficiency, communication reliability, and scalability, they provide validated solutions for cooperative perception.

1.2 Objectives & Research Questions

Recent advancements in high-speed, low-latency networks have made it possible to transmit sensor data between vehicles and infrastructure, improving road safety and extending situational awareness beyond the limitations of local sensors. This dissertation investigates the hypothesis that V2X-based data sharing can enhance safety while offering a cost-effective perception solution for autonomous driving.

The primary objective of this work is to develop robust perception frameworks for Connected and Autonomous Vehicles (CAVs). By utilizing cooperative perception and distributed processing, it addresses challenges related to real-time data handling, communication latency, and perception quality.

1.2.1 Objectives

The following objectives are aimed to address the stated hypothesis through this dissertation. The general objective of this research is to determine how sensor data, whether raw, partially processed or fully processed, can be efficiently shared among connected vehicles using V2X communication technologies.

Research Objective 1: Develop a comprehensive understanding of the literature to formally identify limitations and define the requirements for sensor data offloading using V2X systems. Specify key parameters that fulfill these requirements and address the identified challenges. Determine key performance indicators (KPIs) to be evaluated in connected and autonomous driving applications.

Research objective 2: Create an integrated system that combines a realistic sensor simulation environment with a network simulator to facilitate the exchange of sensor data using V2X technologies. The framework will replicate real-world scenarios to evaluate cooperative perception systems by generating perception data closely resembling the output of actual sensors. Key performance metrics such as latency, bandwidth usage, and perception accuracy will be analyzed under diverse traffic and weather conditions.

Rather than building a new framework from scratch, the focus will be on integrating and optimizing existing open-source tools. This approach will foster collaboration between the CAVs perception and V2X networking research communities. By bridging these domains, the framework will enable scalability and reliability testing across a wide range of realistic scenarios, addressing critical challenges in cooperative perception.

Research objective 3: Design novel distributed data processing and learning techniques to extend resource-intensive computations beyond vehicles, ensuring high reliability and low latency. Achieving fast processing and low latency requires careful consideration of where data is processed and which communication mode is used (V2V or V2I).

For instance, cloud-based processing necessitates the exchange of raw sensor data, which may exceed the bandwidth capabilities of certain V2X technologies. Additionally, the high mobility of vehicles and the short connection times between vehicles and infrastructure further complicate data transmission. To address these challenges, this research will focus on developing efficient data-sharing strategies and identifying optimal trade-offs between data transmission and processing to enable real-time operations.

Research Objective 4: Design and evaluate hybrid edge-cloud architectures to balance computational workloads effectively, minimize latency, and enhance resource efficiency for perception tasks in CAV systems.

Research Objective 5: Evaluate the effectiveness of the proposed perception frameworks by testing them in real-world conditions. This includes evaluating system performance under varying traffic densities, environmental conditions, and communication scenarios to ensure scalability, reliability, and practical feasibility.

Research Objective 6: Develop and release comprehensive datasets to support benchmarking, collaborative research, and the training of perception models in cooperative perception systems. These datasets will encompass diverse scenarios, including varying traffic densities, weather conditions, and sensor configurations, and will include ground truth annotations such as bounding boxes to facilitate training and evaluation. By addressing the current lack of standardized datasets, this objective aims to enable consistent assessment and comparison of system performance under realistic conditions, fostering innovation and advancing cooperative perception technologies.

Research Objective 7: Conduct research throughout the dissertation to systematically identify unresolved challenges in cooperative perception systems and propose actionable directions for future work. The aim is to explore innovative techniques and technologies that address emerging issues as they arise during the study. Potential areas of interest include adaptive data handling strategies, advanced computational methods, and the integration of next-generation communication technologies such as 6G. This objective seeks to ensure that the dissertation contributes to setting a foundation for continued progress and innovation in the field.

1.2.2 Research Questions

This dissertation addresses the following key questions, all of which revolve around two primary challenges in autonomous driving: (1) the development of robust evaluation frameworks and (2) the efficient processing of sensor data across vehicles, edge, and cloud platforms to support resource-intensive real time perception tasks.

Research Question 1 (RQ1)

How can existing vehicular simulation frameworks be leveraged to effectively evaluate cooperative perception solutions?

This question examines how current vehicular simulation tools can be adapted to test and validate cooperative perception systems. It explores their potential to replicate realistic driving scenarios, simulate sensor data exchange, and evaluate performance metrics such as perception accuracy and communication reliability. Addressing this ensures that simulation frameworks provide meaningful insights before transitioning to real-world testing.

Research Question 2 (RQ2)

Can raw sensor data be efficiently shared for cooperative perception?

This question examines the feasibility and efficiency of sharing raw sensor data in V2X-enabled cooperative perception systems. It investigates how such systems can address the limitations of standalone perception, including restricted sensor coverage, occlusions, and environmental variability. By enabling data exchange between vehicles and infrastructure, cooperative perception aims to extend the situational awareness of connected vehicles and improve detection accuracy in complex driving environments. Addressing this question is critical for evaluating the trade-offs between communication overhead, latency, and perception quality in real-world scenarios.

Research Question 3 (RQ3)

What can be the best offloading strategies for real-time perception?

This question seeks to identify the most effective platform for processing sensor data by comparing the onboard system, edge, and cloud. It examines trade-offs between end-to-end delay, detection accuracy, and resource efficiency to determine which platform best supports real-time perception tasks. The analysis helps guide decisions on where processing should occur to achieve optimal performance in distributed perception systems.

Research Question 4 (RQ4)

How can model partitioning and lightweight data-sharing techniques improve the efficiency of cloud-based real-time perception?

This question investigates the role of model partitioning and lightweight data-sharing techniques, such as quantization and compression, in optimizing cloud-based real-time perception systems. It examines how these strategies can reduce bandwidth usage and computational demands while maintaining detection accuracy. By leveraging ITS-G5 and Cellular Vehicle-to-Everything (C-V2X) communication technologies, the study evaluates the trade-offs between latency, perception quality, and system reliability, providing insights into the effective integration of edge-cloud architectures for CAVs.

Research Question 5 (RQ5)

How can cooperative perception systems be validated under real-world conditions to ensure reliable real-time performance?

This question examines methodologies for validating cooperative perception systems in real-world environments, focusing on ensuring reliable real-time performance. It investigates how strategies like data compression, feature sharing, and hybrid edge-cloud processing can address challenges such as high latency and bandwidth limitations. The goal is to identify practical validation techniques that maintain detection accuracy while optimizing resource usage, enabling scalable deployment in dynamic driving scenarios.

Research Question 6 (RQ6)

How can real-time cooperative perception systems be advanced to achieve robust 360-degree 3D detection in real-world V2X environments?

This question investigates advancements in real-time cooperative perception systems to enable robust 360-degree 3D detection in V2X environments. It focuses on leveraging cloud computing combined with techniques such as data compression and feature sharing to optimize network resources without compromising perception quality. The study aims to validate these systems in real-world scenarios, addressing challenges like latency, bandwidth constraints, and detection accuracy in dynamic operating conditions.

Answer to Research Questions: 1-6

Research Question 1, which focuses on leveraging simulation frameworks to evaluate cooperative perception solutions, is primarily discussed in Chapter 3 and Chapter 4. Research Question 2, addressing the feasibility of efficiently sharing raw sensor data for cooperative perception, is explored in Chapter 4. Research Question 3, analyzing optimal offloading strategies for real-time perception, is covered in Chapter 5 and Chapter 6. Research Question 4, which investigates model partitioning and lightweight data-sharing techniques for cloud-based perception, is detailed in Chapter 6. Research Question 5, focusing on the validation of cooperative perception systems under real-world conditions, is examined in Chapter 7. Finally, Research Question 6, emphasizing the advancement of 360-degree 3D detection in real-world V2X environments, is discussed in Chapter 8.

1.3 Methodology

This dissertation takes a systematic approach to address the challenges of sharing sensor data over V2X networks while optimizing cooperative perception systems. The work begins with an in-depth literature review to establish the requirements, key parameters, and limitations of V2X-based sensor data sharing. Building on this foundation, a realistic simulation framework is developed to model the cooperative perception pipeline. The framework dissects the pipeline into its main components, including local processing delay, communication delay, and object detection delay, and evaluates their combined impact on end-to-end performance, focusing on tasks involving camera sensors.

To support these efforts, open-source tools such as CARLA [32], SUMO [99], Veins [135], and Simu5G [114] are utilized, along with widely used libraries like OpenCV, TensorFlow, and YOLO, to measure critical performance indicators such as detection accuracy and latency. Given the significant network resources required for raw sensor data sharing, the methodology incorporates compression techniques, including H.265 and JPEG, for streaming camera data. Additionally, lightweight feature vector sharing is employed through the use of deep neural networks. To further optimize system performance, detection models are partitioned between vehicles and the cloud, creating a hybrid processing setup that reduces computational strain and minimizes latency. This comprehensive methodology provides a cost-effective and adaptable platform for analyzing and improving cooperative perception systems under a variety of traffic conditions, environmental scenarios, and network configurations.

1.4 Contributions of the Thesis

This thesis contributes to cooperative perception and autonomous driving by addressing challenges in simulation, distributed data processing, and experimental validation. Key contributions include developing a customized simulation framework, designing offloading and feature sharing strategies for efficient resource use, and conducting real-world experiments to evaluate perception quality and latency in V2X communication.

1.4.1 Development of Simulation Frameworks

A key contribution of Part-II of this dissertation is the development of customized simulation tools for evaluating cooperative perception systems. These tools enable controlled and repeatable testing of algorithms and frameworks, addressing challenges such as scalability and diverse environmental scenarios. This work is prominently featured in the VNC-2021 [55], CCNC-2023 [54], and MT-ITS-2023 [53] papers, where the following advancements were achieved:

- **Advanced Simulation Tools**: Existing tools were extended to model complex traffic scenarios and cooperative perception setups, supporting detailed performance evaluations in dynamic environments [53, 54].
- **Integrated Communication Channel**: An open-source simulator was enhanced with a communication layer for realistic sensor data exchange over a V2X network. This addition enabled the evaluation of throughput requirements for raw sensor data, including cameras, LiDAR, and radar [55].

1.4.2 Offloading and Distributed Processing Strategies

Part-III of this work focuses on understanding data offloading and distributed processing strategies in infrastructure-supported cooperative perception systems. We examine various data compression techniques for sensor data streaming and explore how processing tasks can be effectively distributed between edge devices and cloud resources. The research addresses critical challenges, such as balancing end-to-end delay and perception accuracy. These findings are detailed in the WONS-2023 [59], VNC-2023 [58] and Computer Communication [57] papers, with notable contributions including:

- **Data Compression**: Evaluated the impact of compression techniques, including H.265 and JPEG, on perception quality and transmission latency [59].
- **End-to-End Delay and Accuracy Trade-Offs**: Analyzed how distributed processing strategies affect end-to-end delay and perception accuracy, across platforms and compression levels. The end-to-end delay corresponds to the total of network latency, compression, decompression, and inference time [57].
- **Feature-vector Transmission:** as an alternative to raw or compressed data sharing we evlauated feature vector transmission to reduce bandwidth usage, maintain perception accuracy, and minimize end-to-end delay [58].

1.4.3 Field Trials and Experimental Evaluation

A key focus of part-III of this dissertation is the real-world testing and validation of cooperative perception frameworks. Field trials were conducted to evaluate the performance of these systems under practical conditions, bridging the gap between simulation and deployment. These efforts are detailed in the VTC-2024 [56] and WONS-2025 [60] papers and include the following contributions:

- **Real-World Testing**: Development and implementation of experimental setups for testing cooperative perception systems in diverse traffic conditions [56].
- **360 degree 3D perception**: A systematic comparison of simulated results with real-world observations to identify discrepancies and validate the reliability of simulation tools [60].

1.5 Organization & Structure of the Thesis

Each chapter has been structured to progressively build on the previous one, addressing distinct aspects of the research problem and contributing to the overarching narrative of the study. The following sections offer a detailed overview of each chapter, highlighting primary objectives and contributions. The remainder of the thesis is organized as follows:

Part I: Introduction
This part provides a comprehensive introduction to the thesis, outlining the motivation, background, and key terms essential for understanding the research.

- **Chapter 1: Introduction**
  Chapter 1 highlights the importance of efficient perception systems in autonomous driving and the challenges associated with real-time processing, data sharing, and cooperative perception. It defines the research objectives, key questions, and methodology, establishing the foundation for the subsequent chapters. The chapter also outlines the thesis structure and summarizes its major contributions.

- **Chapter 2: Background**
  Chapter 2 introduces key terms and definitions, such as perception and object detection and distributed computing strategies, establishing the foundation for understanding the technical details in subsequent chapters.

Part II: Evaluating Perception in Simulation
This group establishes the foundational work in simulating perception systems and developing an evaluation framework for cooperative perception.

- **Chapter 3: Simulation-Based Perception Models**
  Chapter 3 explores the use of simulation for evaluating perception systems in autonomous vehicles. It reviews major perception models (e.g., YOLO) and describes the simulation environments and tools, such as CARLA and OMNeT++, employed to assess detection accuracy, latency, and robustness. This chapter sets the groundwork for evaluating cooperative perception in a controlled setting.

  Faisal Hawlader and Raphael Frank. "Realistic Cooperative Perception for Connected and Automated Vehicles: A Simulation Review". In: *2023 8th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2023. DOI: 10.1109/MT-ITS56129.2023.10241653

  Faisal Hawlader and Raphael Frank. "The Ugly Truth of Realistic Perception in Vehicular Simulations". In: *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*. IEEE, 2023. DOI: 10.1109/CCNC51644.2023.10060697

- **Chapter 4: Evaluation Framework for Cooperative Perception**
  Chapter 4 presents a custom evaluation framework tailored for cooperative perception. The framework incorporates metrics such as latency, perception accuracy, and communication reliability. Simulation results and analyses provide insights into the effectiveness and limitations of cooperative perception under various conditions.

  Faisal Hawlader and Raphael Frank. "Towards a Framework to Evaluate Cooperative Perception for Connected Vehicles". In: *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2021. DOI: 10.1109/VNC52810.2021.9644667

Part III: Edge and Cloud Computing for Cooperative Perception
This group focuses on the distributed processing of perception tasks, examining edge and cloud computing as strategies for improving cooperative perception performance.

- **Chapter 5: Distributed Processing and Offloading Strategies**
  Chapter 5 investigates various distributed processing and offloading strategies for autonomous vehicles, emphasizing latency reduction and bandwidth efficiency. Key contributions include experimental results on the trade-offs between processing time, perception accuracy, and network bandwidth in edge-cloud frameworks.

  Faisal Hawlader, François Robinet, and Raphaël Frank. "Leveraging the edge and cloud for V2X-based real-time object detection in autonomous driving". In: *Computer Communications*. Elsevier, 2024. DOI: 10.1016/j.comcom.2023.11.025

  Faisal Hawlader, François Robinet, and Raphaël Frank. "Vehicle-to-Infrastructure Communication for Real-Time Object Detection in Autonomous Driving". In: *18th Wireless On-Demand Network Systems and Services Conference (WONS)*. IEEE, 2023. DOI: 10.23919/WONS57325.2023.10061953

- **Chapter 6: Optimizing Cooperative Perception Through Compression**
  Chapter 6 explores data compression techniques to reduce bandwidth usage without significantly compromising perception accuracy. Experimental results highlight the impact of compression on latency, and real-time processing capabilities, offering insights into optimizing cooperative perception in data-intensive scenarios.

  Faisal Hawlader, François Robinet, and Raphaël Frank. "Lightweight Features Sharing for Real-Time Object Detection in Cooperative Driving". In: *2023 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2023. DOI: 10.1109/VNC57357.2023.10136339

Part IV: Field Trials and Real-World Deployment
Group 3 shifts the focus to real-world testing and validation of cooperative perception, examining the practical challenges and robustness of perception systems.

- **Chapter 7: Experimental Validation of Cooperative Perception Systems**
  Chapter 7 details the methodology and results from real-world field trials that validate the cooperative perception models. Contributions include observations on end-to-end latency, detection quality, and communication reliability, with a comparison to simulation results.

  Faisal Hawlader, François Robinet, and Raphaël Frank. "Cooperative Perception Using V2X Communications: An Experimental Study". In: *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*. IEEE. 2024. DOI: 10.1109/VTC2024-Fall63153.2024.10757448

- **Chapter 8: Advanced Hybrid Computing for 360-Degree Perception**
  Chapter 8 introduces a hybrid edge-cloud computing model for 360-degree perception, using multi-camera systems to capture comprehensive situational awareness. Field test results assess the feasibility and performance of this model, providing lessons for real-time perception in complex urban environments.

  Faisal Hawlader, François Robinet, Elghazaly Gamal, and Raphaël Frank. "Cloud-Assisted 360-Degree 3D Perception for Autonomous Vehicles Using V2X Communication and Hybrid Computing". In: *20th Wireless On-demand Network systems and Services Conference (WONS)*. IEEE. 2025

Part V: Challenges, Conclusion, and Future Directions
This final group synthesizes the findings, addresses remaining challenges, and discusses future research directions.

- **Chapter 9: Open Challenges and Lessons Learned**
  Chapter 9 examines the technical and methodological challenges encountered during the research, including data synchronization, latency management, and computational constraints. Lessons learned from the translation of simulation results to real-world applications are highlighted, with potential solutions proposed for these challenges.
- **Chapter 10: Conclusion**
  Chapter 10 concludes the thesis by summarizing the key contributions and their implications for cooperative perception and autonomous driving. The chapter also offers a forward-looking perspective on future research opportunities in cooperative perception, V2X communication, and distributed processing.

Each chapter builds upon the previous one, creating a cohesive narrative that progresses from simulation-based evaluation to practical implementation and validation, addressing the research questions and advancing the field of cooperative perception in autonomous driving.

Chapter 2

Background

2.1 Key Terms and Definitions

Connected and autonomous driving involves the integration of technologies that enable vehicles to communicate with other vehicles and infrastructure, such as edge and cloud computing platforms. Vehicles that participate in this ecosystem are known as CAVs. By utilizing advanced sensors, communication systems, and computing technologies, CAV can perceive their driving environment, make decisions, and navigate safely. Figure 2.1 shows an overview of connected and autonomous driving ecosystem, given the complexity, it is essential to define the fundamental terms used throughout this dissertation. Therefore, this section provides definitions of these key terms, organized into relevant subsections for clarity: Perception and Object Detection, Communication and Networking, and Computing and Data Processing.

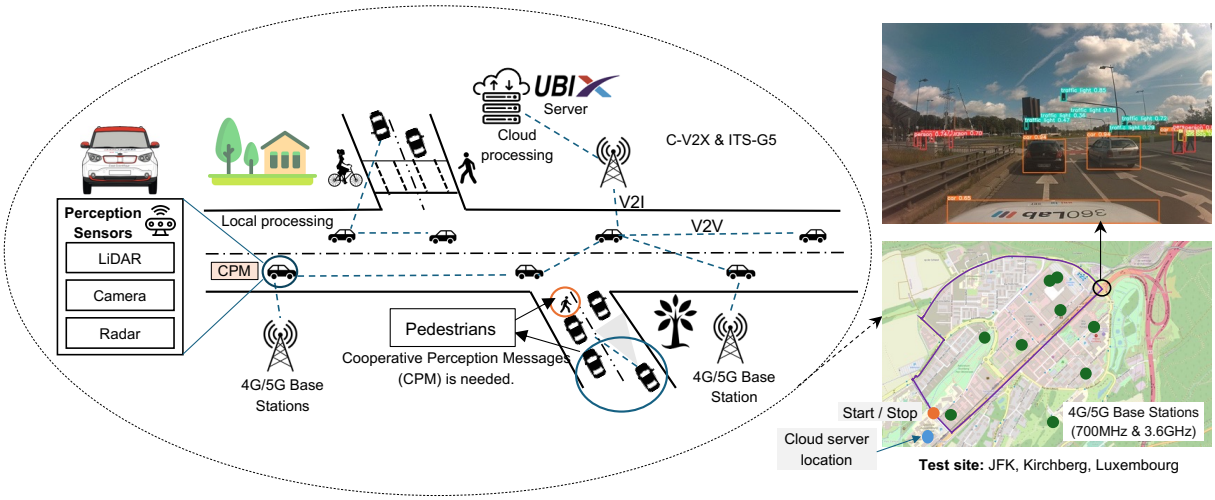


FIGURE 2.1: Overview of the connected driving ecosystem, illustrating the interconnected components, including the vehicle and cloud infrastructure.

FIGURE 2.2: Demonstration of the sensor configuration employed in the autonomous vehicle at 36Lab, SnT, University of Luxembourg. The setup includes a LiDAR for 360-degree environmental mapping, multiple cameras for visual data acquisition, and a GNSS-INS for high-precision localization. **Source:** Adapted from [142], with permission.

2.1.1 Perception and Object Detection

Perception and sensors are fundamental components that enable CAV to understand and interact with their driving environment.

Perception: CAV are equipped with sensors, including cameras, LiDAR, and radar, to collect comprehensive information about their surroundings. Cameras capture high-resolution visual data for object detection and scene understanding, while LiDAR constructs three-dimensional point clouds that provide precise depth information enhancing object detection. Radar detects objects at various ranges and operates effectively even in adverse weather conditions. The perception module processes sensor data to create a real-time understanding of the environment. Key tasks include tracking and segmenting objects such as pedestrians, vehicles, and traffic signs. Figure 2.2 illustrates the perception sensor setup used in our experimental vehicle. Details of hardware and sensor configurations for different experimental evaluations are provided in the following chapters.

Object Detection: Object detection is the computational process of identifying and localizing objects of interest within sensor data, such as pedestrians, vehicles, and traffic signs. This process involves not only recognizing the presence of objects but also accurately determining their positions and sizes. Modern object detection algorithms, such as You Only Look Once (YOLO) [122], leverage deep neural networks applied to camera images to achieve real-time performance and high accuracy. These networks extract hierarchical features from input images, enabling precise detection and classification even in complex environments. Figure 2.3 illustrates this object detection process, where input camera images are processed by a deep neural network that identifies and localizes objects, providing the probability of each detection.

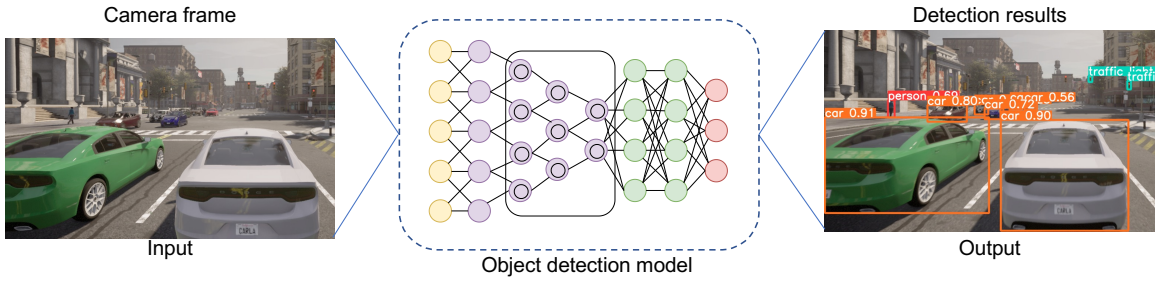


FIGURE 2.3: Object detection process using a deep neural network, where an input camera image is processed to detect objects, assign bounding boxes, and estimate detection probabilities.

Deep Neural Network (DNN) and Feature Vector: DNNs are machine learning models composed of multiple interconnected layers designed to learn and extract complex patterns from large datasets. They excel in tasks such as object detection, efficiently processing and interpreting vast sensor data in real time. A feature vector is a numerical representation of attributes extracted from input data, used by machine learning models to identify patterns and make predictions. For instance, in CAVs, a DNN analyzes images captured by cameras to detect pedestrians and vehicles. These images are transformed into feature vectors, which encode essential characteristics like shapes, textures, and spatial relationships. The DNN processes these feature vectors through its interconnected layers, progressively refining the information to achieve accurate object detection.

Bird's Eye View (BEV): the Bird's-eye view (BEV) is a method that converts sensor data into a top-down image, similar to how a map appears when viewed from above. By combining information from multiple sensors, such as multi-view cameras, LiDAR, and radar, the BEV creates a clear and comprehensive picture of the environment. For example, a BEV provides a top-down view of nearby vehicles, pedestrians, and road signs, thus improving situational awareness of CAV.

2.1.2 Communication and Networking

Communication and networking are essential for CAV to exchange information with other vehicles, as well as edge and cloud infrastructure.

V2X Communication: V2X communication technology is specifically designed for CAVs, enabling seamless data exchange between vehicles, infrastructure, pedestrians, and other road users. This technology enhances situational awareness and facilitates cooperative driving, contributing to safer and more efficient transportation systems. V2X is implemented through two primary standards: ITS-G5 and C-V2X. ITS-G5 [140], a European standard based on the IEEE 802.11p protocol, supports short-range, low-latency communication by allowing direct data exchange without relying on cellular networks. In contrast, C-V2X, defined by 3GPP [7], utilizes cellular networks for communication and supports both direct device-to-device interaction and network-assisted communication via cellular infrastructure. This dual capability enables C-V2X to offer broader coverage and enhanced reliability, making it well-suited for diverse vehicular communication scenarios.

Data Communication: Data communication refers to the transfer of data between CAVs and infrastructure through wired or wireless transmission channels. This process involves the exchange of information in various formats, including images, videos, feature vectors, and binary data, governed by protocols to ensure accurate, efficient, and secure transmission. In the context of V2X communication, data communication enables seamless interaction between vehicles, infrastructure, and other entities, supporting applications such as situational awareness and cooperative perception. Data communication performance is measured by bandwidth, throughput, and latency. Bandwidth is the maximum data transmission capacity of a channel, which determines the ability to handle large data volumes. Throughput is the amount of data successfully transmitted and processed over time, reflecting network efficiency. Latency refers to the delay in data transfer, critical for real-time applications, including cooperative perception.

Cooperative Perception Message (CPM): A CPM is a standardized communication message format used in V2X systems to exchange perception data between vehicles and infrastructure. These messages are generated from sensor data collected onboard a vehicle and contain information about detected objects, such as their position, size, and motion. By sharing these data, CPMs allow nearby vehicles to build a more comprehensive view of the environment, enhancing situational awareness and enabling cooperative driving. This is particularly valuable in scenarios where vehicle sensors may have limited visibility, improving overall safety and efficiency on the road.

2.1.3 Computing and Data Processing

Efficient computing and data processing are essential for managing the vast volumes of sensor data required to enable real-time decision making in autonomous driving systems. These capabilities ensure timely responses to dynamic driving environments, enhancing safety and performance.

Computing Platform: A computing platform refers to the integrated hardware and software environment used to execute computational tasks. In the context of CAVs, it includes onboard systems, such as GPUs and processors, for real-time data processing, and edge or cloud platforms for offloading computationally intensive operations. In local processing scenarios, perception tasks, such as object detection, are performed exclusively on onboard hardware, with detection results being encoded into CPMs and then broadcast to nearby vehicles via ITS-G5 communication technology to enable situational awareness. In cloud processing scenarios, data captured by vehicle sensors are transmitted to edge or cloud platforms using C-V2X communication, where processing is performed on more powerful hardware. The results are then transmitted back to nearby vehicles, supporting cooperative perception and enhancing overall system performance. A combined approach that uses both onboard and cloud computing resources for optimal processing is referred to as hybrid computing. Figure 2.4 illustrates a scenario where vehicles, edge platforms, and cloud infrastructure are interconnected to facilitate distributed computing. This setup enables seamless collaboration between local and remote resources, ensuring efficient processing and enhanced performance.

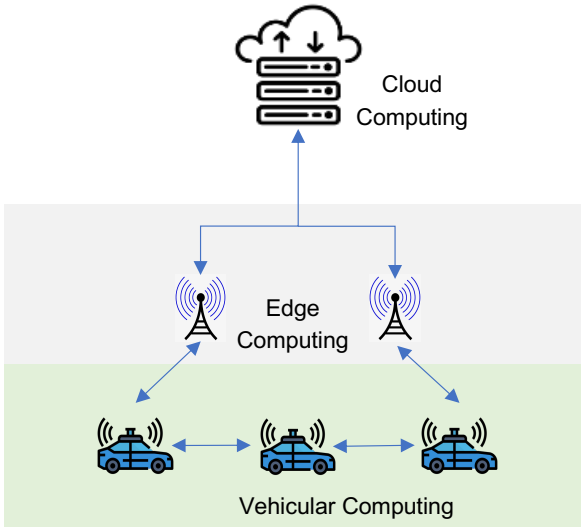


FIGURE 2.4: Cooperative and distributed processing integrating vehicular, edge, and cloud resources for intensive tasks.

Data Compression and Lightweight Features: Data compression is the process of reducing data size to optimize transmission and storage while preserving essential information. Compression techniques can be categorized as lossless, which ensures no data is lost during the process, and lossy, where some data is irreversibly discarded to achieve higher compression ratios. In the context of CAVs, methods such as H.265 or JPEG are often applied to sensor data, including images, to minimize bandwidth usage and enable efficient communication between vehicles and infrastructure.

Lightweight features are condensed versions of raw sensor data that preserve only the essential information needed for tasks like object detection. These features are often derived using DNNs, which process the data to extract meaningful patterns while reducing its size and complexity. This simplification enables faster processing and more efficient transmission, making lightweight features ideal for resource-limited environments like onboard systems or low-latency V2X communication.

2.2 Evaluation Framework

Developing an evaluation framework is essential for validating the reliability and feasibility of cooperative and distributed perception solutions. Two main approaches can be taken: (1) simulation-based methods and (2) real-world experiments. While real-world experiments are often the preferred choice, they pose significant challenges, such as high costs and extensive time requirements due to the expensive hardware involved. This dissertation explores both approaches, starting with simulation-based experiments and culminating in real-world validation. By combining these methods, a thorough and well-rounded assessment of the proposed methodologies is achieved. This section outlines the strategies for simulation and real-world validation, details the experimental setup, and highlights the key challenges and limitations encountered during the evaluation process.

2.2.1 Simulation and Real-World Validation

This dissertation focuses on developing evaluation frameworks for cooperative and distributed perception solutions. The aim is to explore how sensor data processing can be shared between onboard vehicle systems, edge devices, and cloud platforms to strike the right balance between efficient data transmission and processing time.

Simulation-Based Evaluation: Simulations provide a controlled and repeatable environment for testing a wide range of scenarios, including edge cases that may be impractical or unsafe to replicate in real-world settings. To develop the evaluation framework, we leveraged existing vehicular simulators, enhancing and customizing them beyond their default configurations. Specific details about these settings and their applications are discussed in Chapter-3 and Chapter-4. These customizations allowed for evaluations specifically designed for cooperative perception solutions. The resulting framework enables the simulation of realistic scenarios and supports a comprehensive assessment of performance across key metrics, including perception quality, data transmission latency, resource utilization, and communication reliability.

- **Tools and Platforms**: We utilized a combination of existing simulation platforms, including VEINS, Simu5G, INET, CARLA, and SUMO, along with Python-based libraries such as Socket and OpenCV.
- **Evaluation Focus Areas**: Assessing perception quality under varying weather conditions, including day, night, fog, and rain. Comparing grid-based and photo-realistic perception performance in simulations across intersections, downtown areas, parking lots, and highways. Determining throughput requirements for transmitting various sensor data types in distributed processing systems.
- **Advantages**: Creates a safe, controlled environment to test scenarios that would be too risky or impractical in real-world settings. Allows scaling experiments with thousands of simulated vehicles and varying traffic patterns, while enabling tests under extreme conditions like heavy occlusions or challenging weather. Save time and cost compared to real-world evaluations while still providing a way to simulate complex and realistic situations.

Real-World Validation: While simulation-based evaluations provide significant flexibility, real-world testing of real-time perception capabilities is essential to fully unlock the potential of CAVs. Real-world experiments play a vital role in validating the practical feasibility of these systems and addressing the challenges of transitioning from theoretical models to real-world applications. One critical aspect requiring validation in real-world settings is end-to-end delay, as perception systems depend on ultra-low latency to function effectively. These systems must process sensor data and respond within milliseconds to ensure safe and reliable performance in dynamic driving environments. This dissertation provides valuable insights into the design of efficient perception systems within the connected driving ecosystem. Detailed discussions of these advancements are presented in Chapter 7 and Chapter 8.

- **Tools and Platforms**: The 360Lab vehicle at University of Luxembourg, features advanced sensors like cameras, LiDAR, and V2X modules (ITS-G5 and C-V2X) integrated with 5G-enabled cellular communication. Test routes in the Kirchberg area of Luxembourg are designed to cover diverse environments, including urban areas with real traffic. Integrating the onboard vehicle platform with remote edge or cloud systems, including UniLu HPC, enables real-time data processing to evaluate perception and communication performance in dynamic scenario.
- **Evaluation Focus Areas**: End-to-end delay analysis for real-time perception during driving scenarios. Perception accuracy compared to ground truth results. Assessment of communication reliability and latency in real-world conditions.
- **Advantages**: Demonstrates system robustness in dynamic and unpredictable environments. Provides insights into real-world challenges, including environmental noise and mobility issues. Validates the scalability of sensor data offloading and distributed processing across different traffic densities and network conditions.

2.2.2 Experimental Design

The experimental scenarios are designed to systematically evaluate the performance, reliability and scalability of cooperative and distributed perception systems.

Evaluation Scenarios: To thoroughly evaluate the proposed systems, experiments are designed around diverse scenarios that explore various data-sharing strategies, computational demands, and network conditions. These scenarios are intended to uncover the trade-offs and performance benefits of cooperative and distributed perception under realistic operational constraints.

- **Baseline (Local Perception)**: All perception tasks are performed entirely onboard without external data sharing.
- **Cooperative (Distributed)**: Vehicles and infrastructure nodes exchange perception data, facilitating CPM sharing and collaborative perception improvement.
- **Hybrid (Edge-Cloud Integration)**: Integrates local processing with selective offloading to edge or cloud servers to optimize performance and resource utilization.

Metrics for Evaluation: To ensure a thorough and context relevant evaluation of the proposed approach, the following metrics are considered:

- **Perception Quality**: Detection accuracy evaluates how well the model identifies and locates objects in input sensor data, acting as a key measure of its performance in object detection tasks. Mean Average Precision (mAP) is a comprehensive metric used to assess detection performance by comparing the predictions with ground truth data across diverse scenarios. It considers Intersection over Union (IoU) [28], which measures the overlap between predicted and ground truth bounding boxes, ensuring a precise evaluation. mAP provides a detailed assessment of detection effectiveness by accounting for both precision and recall at varying confidence levels.

- **Processing and Network Resources:** Evaluate critical aspects of computational and communication performance required for real-time operations. End-to-end delay is analyzed, including the time needed for compression, transmission, and model inference during detection tasks. Bandwidth usage and packet loss rates are examined under varying network conditions to assess the reliability and efficiency of data transmission. Onboard and cloud inference times are also tracked to provide deeper insights into system responsiveness during distributed perception tasks.

Tools and Hardware: A variety of protocol stacks, software, and hardware are employed to meet the specific objectives outlined in each chapter. While detailed explanations are provided in the respective sections, a brief summary includes:

- **Onboard Systems:** NVIDIA Jetson Orin, optimized for low-power, real-time perception tasks, ensuring efficient local processing.
- **Cloud Infrastructure:** Tesla V100 GPUs, designed for resource-intensive operations such as object detection inference on a large scale.
- **Communication Modules:** ITS-G5 and C-V2X integrated with 5G capabilities, enabling tests on bandwidth and latency impacts during edge and cloud offloading.

This detailed evaluation combines tools, techniques, and objectives, establishing a practical and comprehensive framework for evaluating the proposed systems in both simulations and real-world experiments.

2.2.3 Challenges and Limitations

While cooperative and distributed perception systems hold great potential, their implementation comes with a range of technical and practical challenges. Recognizing and addressing these challenges is vital for improving existing approaches and guiding future research. This subsection focuses on the key difficulties encountered during simulations and real-world validation efforts.

Simulation Constraints: While simulations provide a controlled and flexible environment for testing, they cannot fully replicate real-world complexities. Challenges such as sensor noise, dynamic traffic, and environmental factors like weather and lighting variability are often oversimplified. Similarly, simulated network conditions may overlook practical issues, including interference, packet loss, and bandwidth fluctuations. These gaps highlight the importance of real-world testing to ensure accurate validation.

Real-World Constraints: Conducting tests in real-world settings provides essential insights into how systems perform under actual conditions but also presents considerable challenges. These experiments demand significant resources, including high-cost hardware, extensive infrastructure, and adherence to strict safety protocols. Furthermore, real-world environments are inherently unpredictable, with factors like varying network conditions, environmental noise, and dynamic traffic patterns complicating both testing

and data collection. Such challenges often make real-world validation an expensive, time-intensive, and less reproducible process, emphasizing the importance of simulations as a complementary approach.

Scalability and Resource Demands: Testing large-scale systems with multiple vehicles presents significant challenges. Both simulations and real-world experiments demand substantial computational resources, often pushing the limits of available infrastructure. In addition, urban network congestion can result in packet loss and delays, further affecting the reliability and performance of the system.

Generalization: Findings from tests conducted on specific routes or in controlled scenarios may not easily apply to other settings, particularly those with different traffic dynamics, infrastructure designs, or network characteristics. This challenge highlights the difficulty of ensuring that results are relevant and reliable across a wide range of real-world environments.

In summary, this evaluation framework offers a comprehensive method for testing and improving cooperative and distributed perception systems. By tackling key challenges and recognizing its limitations, the framework ensures that the proposed methodologies are both scientifically sound and applicable in real-world scenarios.

2.3 Cooperative and Distributed Perception

Accurate and real-time perception is essential for autonomous driving. Traditional systems rely on onboard sensors such as cameras, LiDAR, and radar. However, these sensors face inherent limitations, including restricted range, susceptibility to occlusions, and sensitivity to environmental conditions. These challenges become even more pronounced in complex scenarios like urban intersections or dense traffic.

Cooperative perception addresses these limitations by enabling vehicles and infrastructure to share data. By combining information from multiple sources, this approach extends the perception range and enhances accuracy, allowing for more informed decision-making. Distributed perception further advances this concept by dividing computational tasks among vehicles, edge devices, and cloud platforms. This strategy reduces the computational burden on individual vehicles and supports real-time performance.

This section explores strategies for distributed processing and offloading, focusing on the role of cloud computing and the significance of V2X communication. It also highlights key applications, challenges, and research opportunities, providing a comprehensive foundation for the discussions in this dissertation.

2.3.1 Distributed Processing and Offloading Strategies

The growing complexity of perception and decision-making tasks in autonomous driving requires efficient strategies to manage the large volumes of sensor data generated in real time. Distributed processing and offloading optimize system performance by utilizing onboard, edge, and cloud resources to address computational and latency challenges.

Offloading strategies determine how and where tasks are allocated, depending on the system requirements and constraints. The following strategies are commonly used:

- **Static Offloading:** Predefined rules allocate perception tasks based on their computational complexity and latency requirements. For instance, a vehicle may process immediate object detection tasks onboard to minimize latency while offloading global perception tasks, such as 3D mapping or environment modeling, to the cloud.
- **Dynamic Offloading:** Perception tasks are allocated in real time based on network conditions, computational load, and environmental factors. For example, critical tasks like obstacle detection are offloaded to the cloud when bandwidth allows.
- **Hybrid Offloading:** This approach combines static and dynamic strategies to achieve a balance between performance and adaptability. For example, initial data processing is handled locally onboard, while computationally intensive tasks are selectively offloaded to the cloud for enhanced efficiency.

2.3.2 Cloud Computing for Intensive Processing

Cloud computing provides scalable, high-performance resources for handling computationally intensive tasks in autonomous driving, including perception inference and CPM data aggregation. By offloading these tasks, vehicles can overcome onboard hardware limitations and utilize advanced GPUs and large-scale detection models, enabling improved real-time performance and scalability.

Key Benefits: Cloud computing offers several advantages for autonomous driving systems. It provides scalability by dynamically allocating resources based on computational demands, ensuring efficient use of processing power. By aggregating data from multiple sources, it enhances decision-making processes, such as route planning, through a global perception. Additionally, its high-performance capabilities handle complex tasks with advanced computation, improving both accuracy and reliability.

Challenges: Cloud computing in autonomous driving presents several significant challenges. Latency is a critical concern, as communication delays can impact the responsiveness of safety-critical applications, such as collision avoidance. Bandwidth constraints further complicate operations, as transmitting high-resolution sensor data requires extensive network resources. Additionally, privacy concerns arise due to the need to share sensitive sensor data.

2.3.3 V2X-Enabled Perception: Applications and Benefits

V2X communication is revolutionizing the way autonomous vehicles perceive and interact with their surroundings. By facilitating real-time data exchange among vehicles, infrastructure, pedestrians, and other entities, V2X significantly enhances perception capabilities, improves safety, and optimizes traffic efficiency. This subsection examines the applications and benefits of V2X-enabled perception, emphasizing its role in cooperative and distributed perception systems.

Benefits: Integrating V2X communication into cooperative perception systems offers several clear advantages. It enables early detection of hazards, reducing the risk of accidents in scenarios with occlusions or limited visibility. Data sharing among connected entities provides vehicles with a broader and more accurate understanding of their surroundings. Offloading computational tasks to infrastructure reduces the processing load on individual vehicles and improves energy efficiency. Real-time updates help manage traffic more effectively, easing congestion and enhancing overall road network performance.

Applications: V2X-enabled perception provides practical solutions to challenges that traditional onboard systems cannot address. For example, shared sensor data allows vehicles to detect pedestrians hidden by parked cars or other obstructions. It also enables the identification of road hazards, such as debris or stalled vehicles, through alerts transmitted by upstream vehicles or infrastructure. By extending perception beyond the immediate range of onboard sensors, V2X enhances vehicle response capabilities in complex driving scenarios, such as urban intersections or high-traffic areas.

2.3.4 Challenges in Cooperative and Distributed Perception

Despite its advantages, distributed processing presents several technical challenges that must be resolved to ensure reliable and scalable performance. Key challenges include:

- Sharing high-resolution sensor data, such as raw camera images, requires substantial bandwidth, often pushing the limits of communication technologies like ITS-G5 and C-V2X. Additionally, transmission and processing delays can compromise the responsiveness of real-time perception systems.
- High-speed vehicle mobility leads to frequent connectivity changes, disrupting consistent data exchange. Limited communication windows and network handovers can cause delays or packet losses, impacting the reliability of cooperative perception.
- Transmitting sensor data over open networks raises significant privacy concerns, potentially exposing sensitive information to unauthorized access.
- High vehicle densities in urban environments exacerbate network congestion, complicating simultaneous data exchanges. Distributed systems must scale effectively to manage large data volumes without overwhelming communication networks.

2.3.5 Research Opportunities and Motivation

Research Opportunities: Cooperative perception for autonomous driving presents numerous opportunities to address existing challenges and advance the field. However, this dissertation focuses specifically on the following:

- **Simulation Frameworks:** Developing tools to evaluate cooperative perception systems under controlled conditions and analyze key performance metrics.
- **Offloading Strategies:** Investigating perception task offloading between edge and cloud to optimize real-time performance.
- **Communication Technologies:** Analyzing the trade-offs in latency, bandwidth, and perception quality across different standards such as ITS-G5 and C-V2X.
- **Real-World Validation:** Conducting experiments to assess the feasibility and scalability of proposed frameworks and strategies.
- **Dataset Contributions:** Creating and releasing comprehensive datasets to support benchmarking and collaboration in cooperative perception research.

Motivation: These opportunities stem from the need for efficient, reliable, and low-latency cooperative perception systems that can perform in real-world conditions. Advancing these areas will enhance safety and efficiency in autonomous driving while addressing communication and computation challenges and optimizing resource utilization.

Part II

Evaluating Perception in Simulation

Chapter 3

Realistic Perception in Vehicular Simulations

This chapter investigates the limitations of modeling onboard perception sensors in vehicular simulations and examines how realistic simulation environments can help address these challenges. We begin by discussing the constraints of onboard sensors in object detection, especially in complex scenarios involving occlusion, adverse weather, and limited sensor range. Following this, we provide an overview of existing simulation frameworks, emphasizing key differences between grid-based and photorealistic approaches that impact the fidelity of perception outcomes.

We then present our methodologies for implementing and evaluating realistic perception simulations, using both SUMO and CARLA to demonstrate the advantages of photorealistic sensors in capturing dynamic environmental conditions. Through a series of controlled experiments, we analyze the effectiveness of vision-based sensors in improving detection accuracy across various scenarios, including weather variations and vehicle density. By comparing traditional grid-based methods with advanced photorealistic simulations, this chapter contributes to the understanding of how simulation can support the development of more reliable perception models for cooperative autonomous vehicles, laying the groundwork for scalable and robust testing in future studies.

This chapter builds upon the following publications:

Faisal Hawlader and Raphael Frank. "Realistic Cooperative Perception for Connected and Automated Vehicles: A Simulation Review". In: *2023 8th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2023. DOI: 10.1109/MT-ITS56129.2023.10241653

Faisal Hawlader and Raphael Frank. "The Ugly Truth of Realistic Perception in Vehicular Simulations". In: *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*. IEEE, 2023. DOI: 10.1109/CCNC51644.2023.10060697

The chapter is organized as follows: Section 3.1 introduces the foundational concepts and challenges of onboard perception. Section 3.2 details the simulation frameworks and methodologies applied to develop realistic perception solutions. Section 3.3 presents our experimental findings, comparing detection accuracy across various simulation setups. Section 3.4 discusses the limitations of the current approach and situates our work within related studies. Finally, in Section 3.5, we summarize our findings and suggest future directions for enhancing simulation-based perception modeling in vehicular environments.

3.1 Introduction

CAV rely on an array of sensors such as cameras, RADAR, and LiDAR to perceive their surroundings [16, 149]. Numerous studies have highlighted the limitations of onboard sensors in environmental perception [154, 129, 16], prompting the development of cooperative perception to address these challenges [18]. Sensors may struggle to detect objects that are either occluded by other road users within their field of view [16] or situated beyond the effective range of the sensing vehicle. Adverse environmental and weather conditions can further amplify perception challenges by diminishing visibility [98].

Perception outcomes are typically influenced by factors like the proportion of connected vehicles and their spatial arrangement [132]. Therefore, it is crucial to test perception-dependent applications to ensure their reliability. However, real-world testing with actual vehicles is uncommon due to the associated complexity, high costs, and potential risks [55]. Simulation emerges as a viable alternative to tackle these challenges. Simulators can conduct large-scale experiments involving numerous road users. In this context, simulation environments need to be highly realistic to effectively replicate real-world deployment challenges. Artery [127] and Veins [135], both integrated with the mobility simulator SUMO [99], are among the most popular open-source simulation frameworks. These simulators enable large-scale cooperative perception studies without the need for time-consuming and resource-intensive photorealistic simulations. Researchers extend these simulators by adding components that implement perception sensors. This method primarily relies on grid-based projections [8] or simplified geometric computations [49], rendering it less realistic.

Conversely, simulators like CARLA [32, 54], AirSim [130], and SVL [128] are capable of producing realistic sensor data such as RGB images. Although these simulators are computationally demanding and require substantial processing power, their outputs closely resemble those from real sensors. This trade-off becomes more manageable as computational capabilities and efficiencies continue to improve. Nevertheless, most cited studies have employed the first approach without considering detection accuracy [49, 127, 8], potentially leading to unrealistic outcomes. Given that simulated sensor data must coherently represent real-world conditions, configuring such simulations is non-trivial. In this chapter, we address the challenges of modeling perception in vehicular simulations by demonstrating and comparing two distinct approaches. The first approach implements a sensor model in SUMO using geometric calculations, while the second, more realistic approach employs computer vision techniques.

Specifically, we use the CARLA simulator to create photorealistic environments and generate RGB camera images, which are subsequently processed using the pre-trained object detection model YOLOv5 [75].

This chapter particularly focuses on enhancing our understanding of perception sensors in vehicular simulations that emulate real-world challenges, thereby yielding more pragmatic results. Our findings indicate that factors such as vehicle layout and connection ratios affect perception accuracy, similar to real-world scenarios, and can be effectively modeled using a simulator. We observed that detection accuracy varies with weather conditions and distance owing to reduced visibility, which can also be realistically simulated in vehicular environments.

We make the following key contributions in this chapter:

Contributions

- We investigate realistic modeling of perception in vehicular simulations using two well-established frameworks, SUMO and CARLA, with the aim of developing an evaluation framework for cooperative perception systems.
- We compare two distinct methods grid-based and vision-based for modeling realistic perception sensors within vehicular simulation environments.
- We analyze the impact of vehicle layout, connection ratios, weather conditions, and distance on perception accuracy in simulations.

3.2 Related work

Research in Cooperative Perception Systems (CPSs) is emerging and gaining increased attention across various areas, including sensor data processing [104], sensor fusion [18, 166], and V2X communication [98]. Despite this progress, cooperative perception introduces numerous challenges that require further investigation and resolution [3, 8].

Several studies, such as [163, 67], have proposed frameworks to simulate cooperative driving. However, these frameworks do not address the accuracy of local perception sensors, which is crucial for reliable applications. Other works, including [168, 104], explore techniques for sharing sensor data, focusing on transmitting partially or fully processed data rather than raw data. While this approach can conserve network resources, it may compromise the performance of object detectors due to information loss during data processing. On the other hand, [166] advocates sharing raw sensor data, which can maintain higher accuracy but comes at the cost of increased processing and transmission demands, as discussed in [3, 119]. Simulations in [55] show that raw sensor data can indeed be exchanged between vehicles; however, the network throughput required for this level of data transfer is substantial and may exceed the capabilities of current technology.

To address bandwidth constraints, [109] proposed a Layered Costmap-based protocol to optimize data exchange by reducing transmission overhead. While this protocol focuses on minimizing data transmission, it does not evaluate the impact on perception accuracy, which is essential for reliable object detection.

In [49], collective perception was introduced to publish lists of detected objects within a vehicular network. This study utilized the Veins framework [135] along with Artery [127], integrated with the SUMO mobility simulator [99] and the OMNeT++ network simulator [152]. Enhancements were made by adding local perception sensors to the ego vehicle, detecting objects based on positional geometry and assuming that objects are visible if they lie within the direct line of sight of the sensor [107, 42]. Additionally, a filter was applied to exclude objects outside the ego vehicle Field of View (FoV). However, this method fails to account for non-connected vehicles, as their positions remain unknown, which limits the realism of the model.

In summary, while existing research has primarily focused on the networking aspects of CPSs, there has been limited evaluation of perception accuracy. This oversight could lead to unrealistic outcomes, underscoring the need for further study into realistic perception modeling within cooperative perception systems.

Algorithm 1 Grid-based Perception Model in SUMO. **Source:** Reprinted from [53], with permission, ©2023 IEEE.

Given:
$Ego_e, P_e, S_e, S_r, S_f$

Initialize $Ego_e, P_e, S_e, S_r, S_f$
while Ego_e has S_e **do**
    Determine position of Ego_e
    for each vehicle V_i in environment **do**
        $E_{d_i} \leftarrow$ Euclidean Distance(P_e, V_i)
        if $E_{d_i} \leq S_r$ **then**
            $a_i \leftarrow$ angle(Ego_e, V_i)
            if $|a_i| \leq S_f$ **then**
                Add V_i to detected vehicles
                for each V_i in detected vehicles list **do**
                    Perform position plausibility check (V_i, P_i)
                    Perform grid-based projection check (V_i, S_f)
                    Perform line-of-sight check between V_i and V_{p_i}
Return: Detected Vehicles

3.3 Methodology

Various simulation frameworks have been developed to test perception solutions in vehicular applications. These frameworks generally fall into two categories based on the abstraction level of the perception module: grid-based perception and optical sensor-based perception, as illustrated in Figure 3.1.

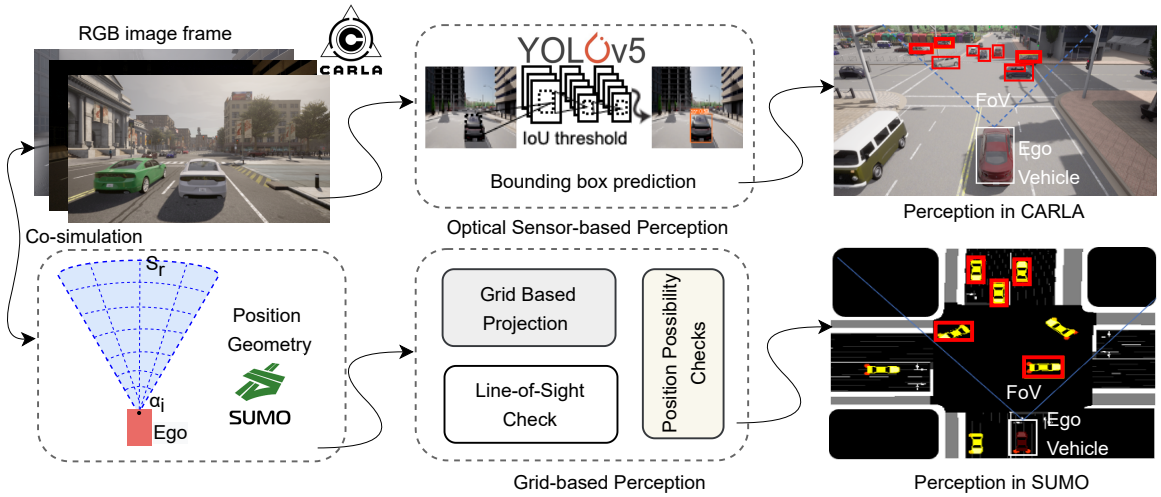


FIGURE 3.1: A robust framework for realistic evaluation of perception sensor models using co-simulation with CARLA and SUMO. The ego vehicle, which is fitted with a sensor, identifies nearby vehicles highlighted by red boxes within the simulation environment. **Source:** Reprinted from [54] with permission, ©2023 IEEE.

Simulations play a crucial role in evaluating perception solutions and advancing connected vehicle technologies by providing a safe, cost-effective way to test and refine perception models. A well-designed simulator can closely replicate real-world conditions, allowing developers to assess how perception systems perform under various scenarios without the risks and expenses of physical testing. In these simulations, optical sensor models are commonly used to mimic how real sensors on CAV would detect objects as they enter into the sensor FoV. Once an object is within this FoV, it can be identified and processed by the perception system. To further enhance realism, photorealistic rendering can be employed to produce detailed sensor data frames, such as RGB images, that closely resemble what actual sensors would capture.

Computer vision-based object detection methods, like those used in real-world systems, are then applied to each frame produced by the simulated ego vehicle. These methods identify objects in the image, draw bounding boxes around them, and assign confidence scores that estimate the likelihood of correct detection. This approach allows simulations to account for key factors that influence perception, such as occlusions (when objects block each other) and confidence levels, which add depth and realism to the sensor model. For further technical details on the object detection process, please refer to [75].

Researchers today frequently rely on probabilistic or geometric-based sensor models, as they balance complexity with computational efficiency, making large-scale, fast simulations possible. However, these models tend to be overly deterministic, lacking the variability and richness of real-world scenarios. To bridge this gap, 3D simulators with photorealistic features provide a compelling alternative, creating realistic environments that incorporate elements such as weather variations, thereby bringing simulations closer to real-world conditions. Although photorealistic simulations are computationally intensive, improvements in processing power and cost-effectiveness are making them more accessible, supporting more detailed and nuanced testing [32].

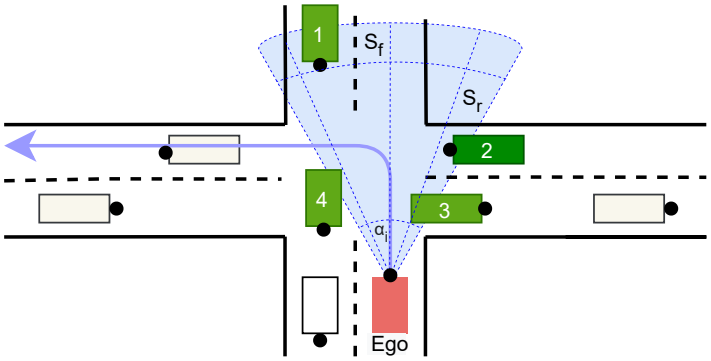


FIGURE 3.2: Grid-based perception model in SUMO. This model simulates the ego vehicle perception capabilities within a large-scale traffic scenario. **Source:** Reprinted from [53] with permission, ©2023 IEEE.

This section compares two approaches for simulating perception sensors: one based on geometric calculations and the other on photorealistic rendering, both designed to address real-world challenges. Developing a simulation framework that accurately replicates real-world conditions is complex, requiring that sensor data remain readable, synchronized, and representative of actual driving environments.

3.3.1 Grid-based Perception

We created a Python-based perception model inspired by foundational work [49, 8] and integrated it with the SUMO simulator, an open-source platform capable of modeling large-scale road networks with both vehicles and pedestrians. Since SUMO does not include native support for perception sensors, we defined an ego vehicle and implemented a custom perception sensor to allow it to detect its surroundings. This setup enables the ego vehicle to detect nearby objects within a predefined FoV using a filtering mechanism. The perception model is detailed in Algorithm 1, which shows the pseudocode used to simulate perception on a broader scale within SUMO.

Once the ego vehicle detects other vehicles in its FoV, the model conducts additional checks, including line-of-sight and grid-based projection validations, to accurately place each detected vehicle on a grid. As illustrated in Figure 3.2, this implemented sensor enables the ego vehicle to identify specific nearby vehicles, such as vehicles 1, 3, and 4, within its immediate environment. Each grid that contains a detected vehicle is marked as occupied while the remaining grids are categorized as either empty or obstructed. To simulate this, we configured a forward-facing sensor mounted on the ego vehicle, designated as Ego_e, and divided the sensor FoV, S_e, into grids for detailed spatial analysis. Each detected vehicle is represented by V_i, with its position denoted as P_i. The perception model classifies each grid cell into one of three states: empty, occupied, or obstructed. These classifications are based on a series of position plausibility checks (V_i, P_i), grid-based projection checks (V_i, S_f), and line-of-sight verifications (V_i, V_{p_i}). The model is optimized to reduce computational demands by performing line-of-sight and grid-based projection checks only for vehicles within the sensor FoV.

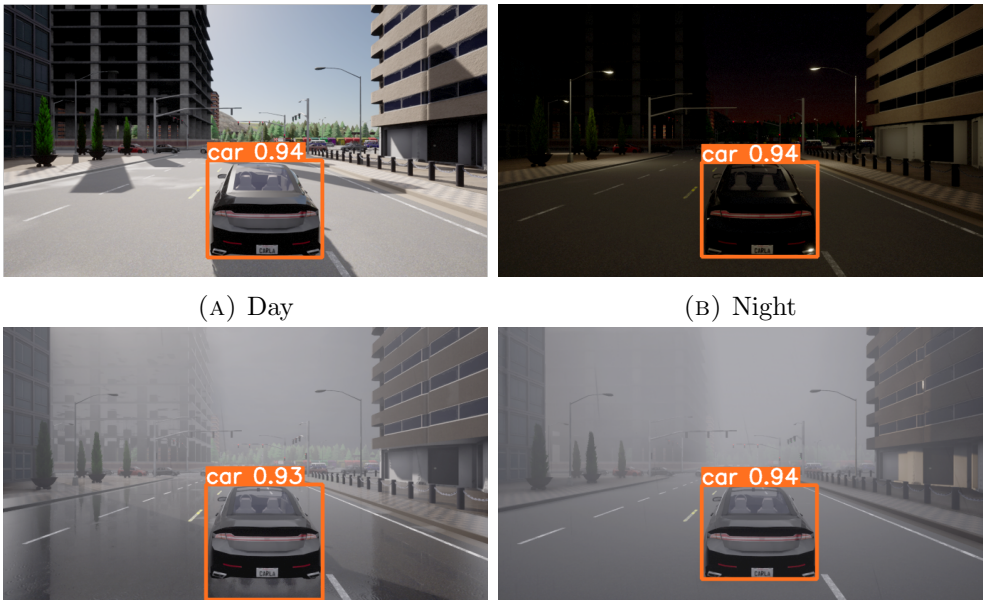
The line-of-sight check relies on the predefined sensor range S_r and angle α_i to identify the exact locations of surrounding vehicles, while excluding those outside the sensor reach. This helps narrow down the list of detectable vehicles to those within the operational view of an Ego vehicle. For example, Figure 3.2 shows that vehicle 2, although present in the scene, is obstructed by vehicle 3 and thus cannot be detected. This approach enables efficient simulation of complex perception scenarios by accurately differentiating visible from non-visible objects in the driving environment for CAVs.

3.3.2 Photo realistic Perception

Our aim is to create a perception module for CAV that closely replicates real-world conditions. To achieve this, we utilize the CARLA simulator, a high-fidelity autonomous driving research platform that enables the generation of photorealistic perception data. To ensure consistency between simulation platforms, we configured a camera sensor for the ego vehicle in CARLA to match the FoV used in SUMO. This alignment was achieved by carefully calibrating the camera parameters in CARLA to replicate the visual perspective and coverage of the SUMO setup, enabling a uniform evaluation of perception scenarios across both simulation environments.

CARLA offers extensive functionalities for realistic perception simulation, including a flexible Python API that allows fine-grained control over each component. Although CARLA provides a variety of features, we focused specifically on simulating adverse weather conditions to examine how environmental factors affect the perception accuracy of CAV. We tested four weather scenarios, clear day, night, rain and fog, to capture a range of visibility and environmental challenges that affect sensor performance, as shown in Figure 3.3. These varied conditions allowed us to evaluate how factors like lighting, visibility, and moisture influence the ability of the perception module to detect and interpret objects in the environment.

In addition to cameras, CARLA supports various onboard sensors, such as LiDAR, with adjustable settings to meet specific project requirements. These sensors can be customized in terms of mounting location, resolution, and FoV, allowing researchers to simulate different sensor configurations and placements. However, CARLA does not come with an integrated object detection system, so we implemented our own using a pre-train YOLOv5 [75], a state-of-the-art detection model renowned for its speed and accuracy. Trained on the COCO dataset [88], which contains 91 object classes and 2.5 million labeled instances, YOLOv5 is highly capable and includes advanced features like multiscale anchor optimization [167] to enhance detection accuracy for small or overlapping objects. For a detailed description of the YOLOv5 detection process, see the official documentation [75].



(A) Day (B) Night

(C) Rainy (D) Foggy

FIGURE 3.3: Performance of Detection Under Various Weather Conditions. **Source:** Reprinted from [53] with permission, ©2023 IEEE.

3.3.3 Vehicle Detection and Counting

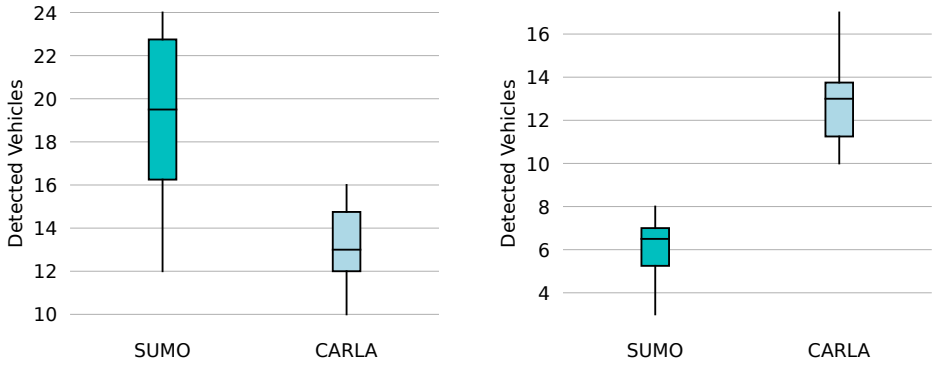
Accurate vehicle detection is crucial, in general, a detector computes a confidence score (C_s) to estimate how likely a bounding box contains the target object [122]. A common formula to calculate C_s is given in Equation 3.1.

$$C_s = P_r(C_i|O) \times P_r(O) \times IoU \tag{3.1}$$

Where $P_r(O)$ is the probability that the bounding box contains an object, $P_r(C_i \mid O)$ is the probability that this object belongs to class i, and IoU is the Intersection over Union between the predicted and ground-truth bounding boxes. For the total number of detected vehicles (D_v), only those with confidence scores exceeding a predefined threshold C_{th} are counted, as shown in Equation 3.2.

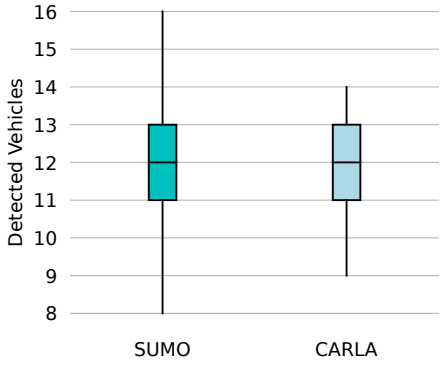
$$D_v = \sum_{k=1}^{V} \mathbf{1}(C_s^k \geq C_{th}) \tag{3.2}$$

Where $\mathbf{1}(C_s^k \geq C_{th})$ is an indicator function that returns 1 if $C_s^k \geq C_{th}$, and 0 otherwise. A typical threshold of $C_{th} = 0.5$ filters out low-confidence detections, thus improving overall detection precision when counting the total number of detected vehicles. In photorealistic models (e.g., YOLOv5), C_s varies continuously from 0 to 1. However, grid-based perception systems treat detections as binary ($C_s = 0$ or $C_s = 1$), omitting partial probabilities.

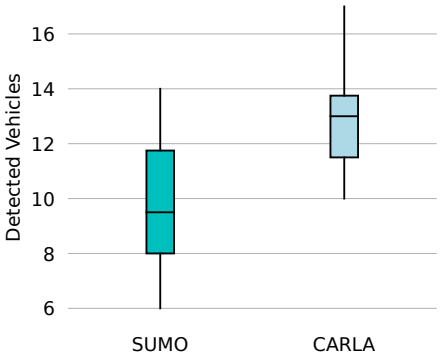


(A) Scenario-1: Highway-4 lane

(B) Scenario-2: Parking lot



(C) Scenario-3: Intersection



(D) Scenario-4: Downtown

FIGURE 3.4: Simulation results for four distinct scenarios. Subfigure presents results for grid-based and optical sensor models. **Source:** Reprinted from [54] with permission, ©2023 IEEE.

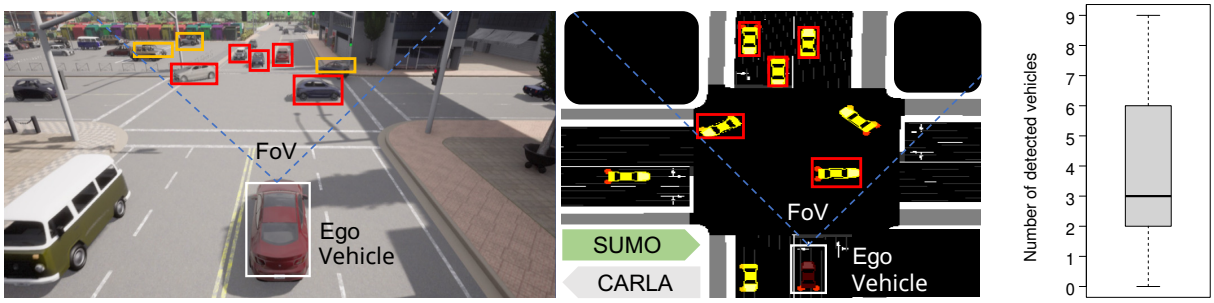
3.4 Results

To assess both perception approaches, we conducted simulations across four distinct scenarios under clear-day weather conditions: (a) a four-lane highway, (b) a parking lot, (c) a busy intersection, and (d) a downtown area. For traffic generation and scenario design, we utilized CARLA's *town03* and *town06* environments.

Parameter	Symbol	Value	SUMO	CARLA
Confidence threshold	C_{th}	50%	×	√
Field of view (FoV)	S_f	60°	√	√
Sensor range	S_r	50m	√	×
Frame rate	F_r	1Hz	√	√
Confidence score	C_s		√	√
Simulation time		240s	√	√

√: Used, ×: Not used

TABLE 3.1: Simulation parameter settings for CARLA-SUMO environments. **Source:** Reprinted from [53], with permission, ©2023 IEEE.



(a) Overview of simulations of CARLA (left) and Sumo (right). (b) Detection Difference

FIGURE 3.5: Evaluation of perception sensors in realistic scenarios utilizing CARLA-SUMO co-simulation. Red boxes (a) denote vehicles identified by the specific simulator. Yellow boxes highlight vehicles exclusively detected by CARLA. The disparity in detections between the two models is shown in 3b. **Source:** Reprinted from [53] with permission, ©2023 IEEE.

All simulations were run with the default settings provided by the CARLA-SUMO co-simulation script [32] to ensure reproducibility. Table 3.1 summarizes the important simulation parameters. Each scenario was simulated for 300 seconds in asynchronous mode to maintain consistent traffic flows across runs. The perception sensor was set to a fixed field of view (90°) and a 1 Hz frame rate in synchronous mode, enabling identical capture rates. As CARLA lacks an integrated object detection system, we incorporated a pre-trained YOLOv5 model [75] for detecting vehicles. Only true positive detections were counted to ensure accurate results in vehicle perception, and the findings are displayed in Figure 3.4.

In the highway scenario (Fig. 3.4a), the grid-based perception system identified over 16 vehicles roughly 75% of the time, with a detection range from 12 to 24 vehicles. In contrast, the optical sensor approach identified over 15 vehicles only 25% of the time, with a maximum of 16. For the parking lot scenario (Fig. 3.4b), grid-based perception detected fewer than 7 vehicles 75% of the time, with a minimum of 2, due largely to obstructions affecting sensor visibility and vehicle layout. In all scenarios, optical sensors performed consistently, with a median detection count of 13 vehicles, which held steady across both the busy intersection and downtown environments (Fig. 3.4c and Fig. 3.4d). More research is required to fully understand these performance differences, as grid-based perception exhibited instability and dependence on the scenario. Additionally, the binary nature of grid-based detection (i.e., vehicles are either detected with 100% confidence or not at all) lacks the probabilistic nuance seen in real-world detection systems, where factors like distance, weather, and vehicle speed influence detection likelihood.

3.4.1 Individual Perception

The study was designed to assess detection accuracy and compare different perception sensors. The objective was to identify specific edge scenarios conditions where individual sensor perception diverges depending on the simulation environment and determine the scenarios that facilitate a more detailed analysis.

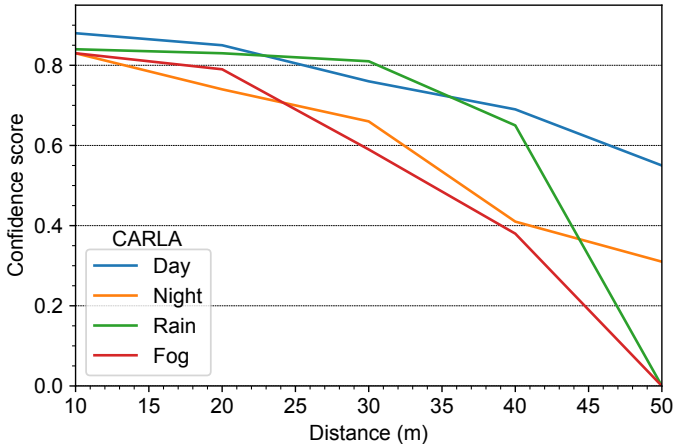


FIGURE 3.6: Detection confidence across varying distances and weather conditions. **Source:** Reprinted from [53] with permission, ©2023 IEEE.

To accomplish this, a 240-second simulation was conducted under synchronized conditions, which allowed for controlled driving behavior of the ego vehicle and enabled consistent capture of the same area by the perception sensors. This synchronization ensured that comparisons across sensors were reliable, allowing us to observe how each sensor responded to identical conditions.

The results in Figure 3.5 indicate that CARLA achieves a more accurate detection compared to SUMO, particularly in high-density traffic scenarios. Detection in SUMO is based solely on vehicle location data and basic geometric calculations, rather than vision-based sensors. Consequently, vehicles are detected only if the sensor mounting point has a clear line of sight. In contrast, vision-based sensors in CARLA can detect vehicles even in partially obstructed layouts. For parked vehicles that do not broadcast position updates, the ego vehicle in SUMO lacks real-time location information, making it unable to perform line-of-sight or grid-based projection checks effectively. Conversely, the position plausibility check marks the actual positions of non-connected vehicles as empty grid spaces, leading to their non-detection.

To gain a clearer understanding, we compared detection between SUMO and CARLA at each time step. The results, shown in Figure 3.5, indicate that the CARLA sensor performed better, with the maximum detection difference reaching up to 9. These tests were conducted under clear weather conditions. The SUMO sensor detects vehicles in a binary fashion, merely indicating the presence or absence of a vehicle without any margin for detection error. This binary detection is always reported with a fixed 100% confidence score or not at all, which lacks the realism. In real-world scenarios, object detection typically involves a probabilistic distribution influenced by factors such as distance, weather, and speed. The vision-based sensor addresses this challenge by providing a confidence score for each detection, offering a more realistic approach, as depicted in Figure 3.3.

To examine how dynamic weather conditions affect detection, we conducted tests across four distinct weather scenarios (Figure 3.3), positioning the detected vehicle at fixed distances of 10 m, 20 m, 30 m, 40 m, and 50 m. We averaged 240 detections at each distance under varying weather conditions.

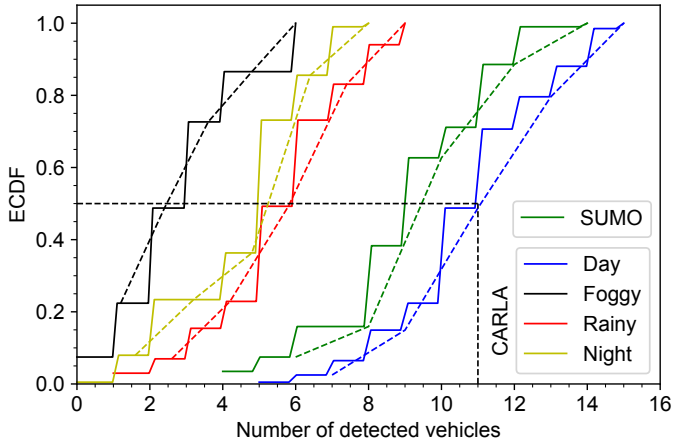


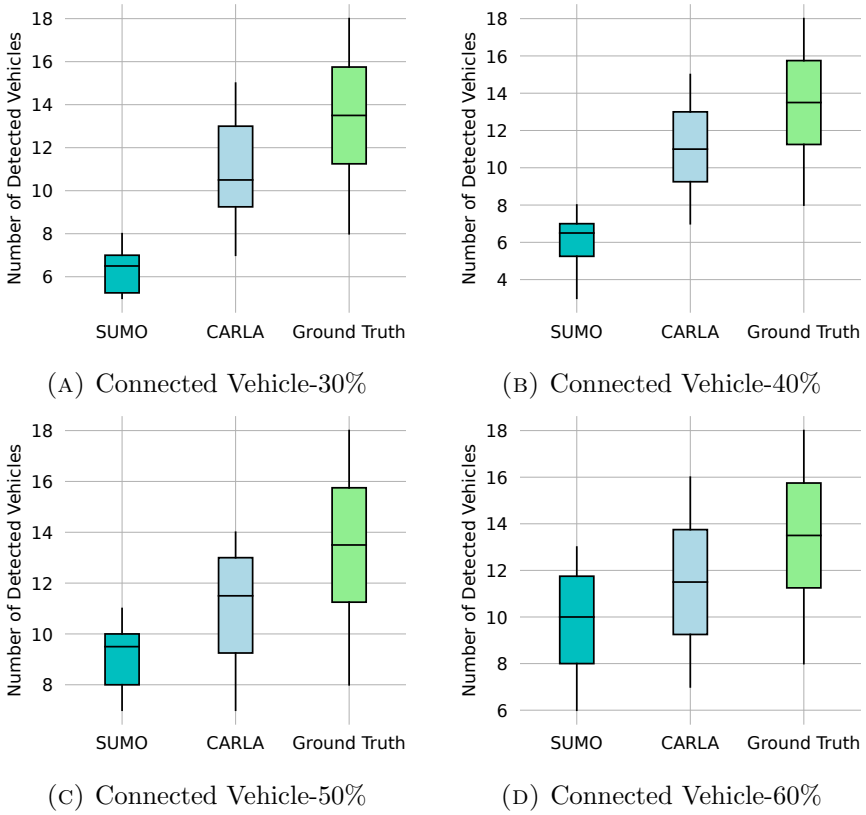
FIGURE 3.7: Empirical cumulative distribution of vehicles detected under varying conditions. **Source:** Reprinted from [53] with permission, ©2023 IEEE.

Figure 3.3 shows the detection results at a distance of 10 m, though confidence scores decline markedly as the distance increases. Even in clear daylight, detection accuracy drops by over 50% at 50 m. Under foggy conditions, accuracy diminishes significantly, with no detections at 50 m or beyond. A summary of these findings is provided in Figure 3.6. This visualization provides insight into how detection performance varies with distance, emphasizing key patterns and trends observed in the analysis.

We applied the empirical cumulative distribution function (ECDF) to calculate the distribution of detected vehicles, with the results displayed in Figure 3.7. As shown, the vision-based camera sensor identifies more vehicles in clear daytime conditions, though detection rates drop significantly in low-light, foggy, and rainy conditions. This outcome aligns with realistic expectations, suggesting that a vision-based sensor can more accurately simulate perception in complex, photorealistic environments, whereas a simplistic sensor model may struggle to capture these nuanced dynamics.

3.4.2 Cooperative Perception

In this section, we assess the performance of our sensor model within a cooperative environment. Intersections are common areas where perception sensors face challenges due to potential obstructions. For example, external objects near the front of an ego vehicle can easily limit its FoV, and a CPS is designed to mitigate such visibility restrictions. To test the CPS, we selected a multi-lane intersection. In the experiment, two ego vehicles, each equipped with camera sensor setups comparable to those in standalone perception, were used. The objective was to compare sensor models implemented in SUMO and CARLA, identifying the most accurate approach for simulating CPS in a cooperative context. We set $d_{max} = 100m$ as the maximum communication range between vehicles, given the compact nature of both the simulation environment and the intersection. Vehicles positioned within $d \leq d_{max}$ can communicate via a socket-based channel, as described in [55].



(A) Connected Vehicle-30%

(B) Connected Vehicle-40%

(C) Connected Vehicle-50%

(D) Connected Vehicle-60%

FIGURE 3.8: Cooperative perception scenario compares the number of vehicles detected using SUMO and CARLA against the ground truth count. The results demonstrate a significant decline in SUMO detector as the proportion of nonconnected vehicles increases. In contrast, CARLA maintains reliable detection performance, showing independence from the proportion of connected vehicles. **Source:** Reprinted from [53] with permission, ©2023 IEEE.

The Euclidean distance was used to calculate the distance between the two ego vehicles. We recorded the total vehicle detections while both ego vehicles stayed within communication range. Figure 3.8 presents the total vehicle count at the intersection and the combined detection count for the two ego vehicles in CARLA and SUMO, using identical scenarios in synchronous mode.

Overall, CARLA consistently detected more vehicles at all stages. Analyzing the average detection discrepancy, we observed that CARLA identified approximately 40% more vehicles compared to SUMO. Performance was further evaluated across four distinct scenarios, all in clear weather and under sunny conditions, with varying levels of vehicle connectivity (30%, 40%, 50%, 60%). The results in Figure 3.8 indicate that the effectiveness of the SUMO detector diminished as the number of nonconnected vehicles increased. Conversely, CARLA maintained reliable performance regardless of the proportion of connectivity, demonstrating stability in all scenarios. Thus, we suggest that an optical vision-based sensor should be utilized when evaluating cooperative perception solutions, as simplifying sensor models too much could overlook crucial details.

3.5 Conclusion and Future Work

In this chapter, we examine the implementation of realistic perception in vehicular simulations for CAV. Our approach leverages the CARLA-SUMO cosimulation framework, an open source tool specifically crafted for automated driving research, which enables the generation of photo realistic perception data. Previous research has primarily relied on simplified grid-based sensor models that often overlook detection accuracy, potentially leading to unrealistic outcomes.

We evaluated the commonly used grid-based sensor model within the research community and compared its performance to that of a photorealistic perception model, which provides a more accurate representation. Our findings demonstrate that detection accuracy is influenced by factors such as vehicle positioning, the ratio of connected vehicles, weather conditions, and the distance between the sensing vehicles and the target vehicle. Our results reveal that a vision-based optical sensor outperforms the grid-based sensor in cooperative perception scenarios, offering a more realistic approach that closely aligns with real-world complexities. Our analysis underscores the importance of incorporating realistic perception models in vehicular simulations, as simplifications can compromise the reliability of results. Future work will focus on embedding a realistic communication channel using standard V2X technologies to enable sensor data exchange between vehicles, thus advancing the evaluation of realistic cooperative perception solutions.

Chapter 4

Framework for Cooperative Perception Evaluation

This chapter presents efforts to extend the CARLA simulator to enable cooperative perception in CAV. We developed a client-server extension that allows simulated vehicles to exchange sensor data, thus enabling cooperative perception scenarios that are typically challenging and costly to test in real-world environments. The chapter begins by discussing the importance of cooperative perception in enhancing road safety and situational awareness. By sharing sensory data through V2X communication, vehicles can expand their perception range. A review of existing CAV simulation platforms highlights the lack of open-source options that effectively combine realistic perception with robust communication models.

To address this gap, we introduce a CARLA extension that supports sensor data exchange among multiple vehicles, enabling cooperative perception experiments in a photorealistic simulation setting. Preliminary evaluations demonstrate the successful sharing of various sensor data, including camera images, LiDAR point clouds, and radar data, between simulated vehicles. We also analyze the bandwidth requirements for transmitting raw sensor data, underscoring current network limitations and the critical need for optimized data transmission strategies.

This chapter builds upon the following publication:

Faisal Hawlader and Raphael Frank. "Towards a Framework to Evaluate Cooperative Perception for Connected Vehicles". In: *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2021. DOI: 10.1109/VNC52810.2021.9644667

The chapter is organized as follows: Section 4.2 reviews related work in cooperative perception and existing simulation platforms for CAVs. Section 4.3 describes the architecture of our extended CARLA simulator and the implementation of the communication framework. Section 4.5 presents preliminary evaluations, showcasing the effectiveness of inter-vehicle sensor data sharing and assessing bandwidth requirements. Finally, in Section 4.6, we summarize our findings and discuss future work, including the integration of more realistic communication models.

4.1 Introduction

CAV present valuable opportunities across numerous applications, supporting everything from improved road safety and awareness to enhanced comfort. A key application area is cooperative perception, which enables vehicles to share sensory data through V2X networks, thereby expanding their perception range without the need for an extensive array of costly sensors [87]. For cooperative perception systems to operate effectively, it is essential that vehicles can accurately observe road conditions and identify moving objects, such as other vehicles, pedestrians, and bicycles, in real time.

Significant research has focused on the transmission and processing of high data volumes between CAV in recent years [131, 8]. With advancements in high bandwidth, low latency networks like 5G, exchanging and handling large amounts of raw sensory information between vehicles is increasingly feasible [64]. Collaborative processing of perception data brings two primary benefits: first, it extends the perception range by combining data from multiple vehicles, minimizing the risk of missed detections due to obstacles or challenging weather conditions [18, 104]. Second, it enables the distribution of data processing tasks among several vehicles, roadside units, and potentially the cloud, where more resource-intensive computations can be performed [14, 10]. This distributed approach supports a high level of situational awareness, even with a simplified sensor setup. Despite its advantages, cooperative perception can produce redundant data [67], making it essential to balance data transmission and processing times. For instance, in dense traffic, several vehicles may transmit substantial information about the same object, straining network resources and adding to processing delays [8]. Testing these applications with real vehicles is crucial, but challenging, due to the high costs and complexity involved [131]. Simulation emerges as a practical alternative to overcome these limitations. Currently, there appears to be no open source simulator that integrates a realistic perception layer with a comprehensive communication model. This chapter seeks to bridge this gap by introducing an extension to CARLA [32].

We make the following key contributions in this chapter:

Contributions

- An extension for the CARLA was developed to enable the exchange of sensors data, such as camera, LiDAR, and radar, between vehicles through a client-server architecture.
- Demonstrated how vehicles can effectively share sensor data, enabling cooperative perception within a photorealistic simulation environment.
- Initial tests on bandwidth requirements for transmitting sensor data between vehicles reveal constraints in current network technologies, highlighting the need for optimized transmission strategies.

4.2 Related work

Related work falls into two key categories: (1) research on cooperative perception and (2) existing simulation platforms for CAV.

Research on cooperative perception, though still in its early stages, has been rapidly advancing [104, 131, 144, 41]. For example, in [143], the impact of various CPMs policies, as established by the European Telecommunications Standards Institute (ETSI) [34], on both network and application performance is examined. Similarly, [104, 133] provides a comprehensive analysis of cooperative perception techniques, discussing their requirements, limitations, and overall performance in detection and communication. In [8, 146], a deep reinforcement learning-based method is proposed to intelligently manage data transmission with the goal of reducing network load. Another approach, as explored in [104, 18], involves transmitting processed or partially processed data instead of raw sensor data.

Among open source platforms that integrate network and mobility simulation, Veins is widely used [135] , offering a broad set of models to enhance vehicular network simulations while maintaining efficient execution speeds. Similar capabilities are provided by the Artery V2X Simulation Framework [63, 127]. However, these platforms lack a photorealistic driving environment, which is essential to evaluate cooperative perception methods. Recently, CARLA has gained popularity as an open source simulator for automated driving research [32]. CARLA supports a wide range of features, including flexible sensor configurations, environmental controls, and management of all static and dynamic entities, allowing users to simulate the full autonomous vehicle stack within a visually realistic 3D environment. Another notable option is AirSim [130, 17], which provides high-fidelity simulation for both ground and aerial vehicles, featuring detailed visual and physical realism. Currently, few frameworks combine a photorealistic environment for generating synthetic perception data with a robust network simulator [52, 47]. This chapter examines the network resources needed for sensor data transmission and discusses the approaches required to achieve such an integration.

4.3 Methodology

Our methodology uses the CARLA client-server architecture to facilitate data exchange between multiple vehicles. In this section, we provide an overview of the key components of the simulator and detail our integration of a communication framework, as illustrated in Fig.4.1. For an in-depth description of the CARLA components, refer to [32].

Although the client-server structure is designed for distributed computing across multiple machines to reduce the hardware burden on individual devices, our current implementation operates on a single machine. The server manages data handling and routing, enabling large-scale simulations even in a centralized setup. This modular design also

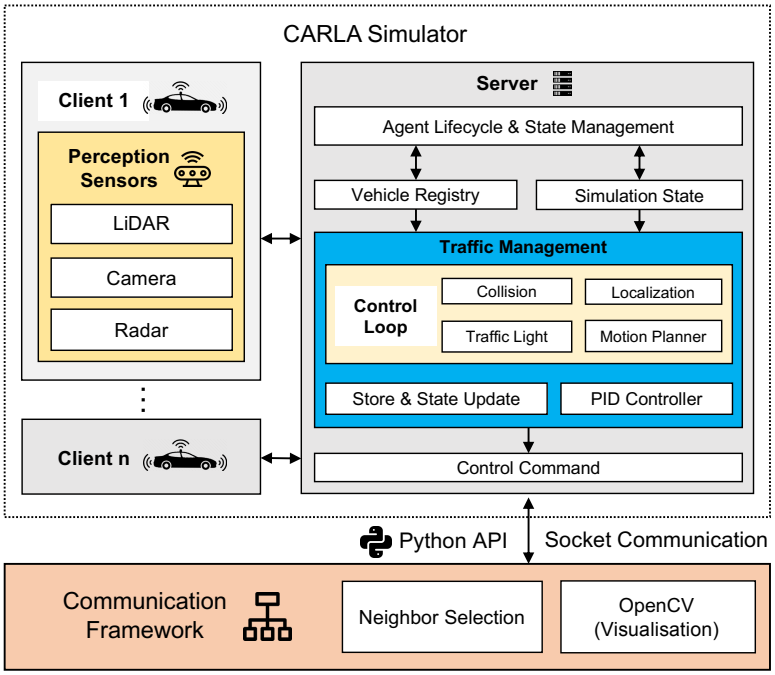


FIGURE 4.1: Simulation architecture illustrating the integrated framework for cooperative perception and data communication between vehicles. **Source:** Reprinted from [58] with permission, ©2023 IEEE.

allows for future enhancements, such as integrating realistic network models and more advanced perception solutions, which we plan to investigate in subsequent studies.

4.3.1 Server

The CARLA server functions as the core of the simulator, managing the registration and life cycles of ego vehicles (clients) and coordinating interactions with them. It maintains the simulation state, including vehicle positions, speeds, and integrates with the traffic management module to control the behavior of all simulated agents, such as vehicles, pedestrians, and traffic lights. This simulation state includes essential details for collision management and motion planning. In the lower layers, a PID controller handles lateral and longitudinal control, leveraging data from the motion planner to generate precise commands to accelerate, brake, and steer vehicles within the simulation.

4.3.2 Client

The provided API enables full control over simulation elements, such as traffic management and perception sensors for one or more ego vehicles, establishing CARLA as a strong foundation for developing a framework to test cooperative perception scenarios. Each client in CARLA represents a simulated ego vehicle, managed by exchanging messages with the server via a Python API. Ego vehicles differ from other simulated agents

FIGURE 4.2: In the cooperative perception scenario, the ego vehicle operating as a client can access camera sensor data from other vehicles within communication range. This data is processed and visualized in real-time using OpenCV, enabling the ego vehicle to analyze shared visual information effectively. **Source:** Reprinted from [58] with permission, ©2023 IEEE.

as they implement the full autonomous driving stack, including essential perception sensors. Although CARLA is typically used with a single client, its client-server architecture enables the generation of multiple clients, a critical feature of our application. However, it is important to note that increasing the number of ego vehicles requires substantial hardware resources to maintain smooth simulation performance [32].

Each client can configure various sensor types, including the following:

- **Camera**: Vision sensors, such as standard RGB and stereo cameras, configurable with different resolutions and fields of view.
- **LiDAR**: A rotating laser scanner that generates point clouds, creating a 3D representation of the environment around the vehicle.
- **Radar**: Produces a conic view that translates into a point map of nearby objects and their speeds, aiding in object shaping and movement assessment.

4.4 Network Layer Implementation

This section provides an overview of the network layer and the neighbor selection module implemented for efficient inter-vehicular communication among ego vehicles in a simulated environment. The setup leverages socket communication for data exchange and uses OpenCV for sensor data serialization and visualization.

To facilitate information exchange between ego vehicles, we developed a socket-based network layer, enabling data transmission between the client (ego vehicle) and the server, as well as among clients. In this setup, an ideal communication channel is assumed, with plans to investigate more realistic channel models in future work. The network layer employs a client-server architecture, where each ego vehicle functions as a client, and the server serves as a central node that coordinates data exchange.

4.4.1 Socket Communication Setup

Each ego vehicle (client) establishes a TCP socket connection with the server. TCP ensures reliable data transmission, maintaining packet integrity and order, which is essential for synchronizing data across vehicles. The server listens for incoming connections from multiple clients, creating and maintaining a persistent communication channel with each. This setup allows clients to broadcast data to vehicles within range via the server, which functions as a message broker.

4.4.2 Data Transmission Protocol

Upon connection, each client is assigned a unique identifier, allowing the server to manage and route data accurately between clients. Data is serialized (e.g., in JSON format) before transmission to reduce size and transmission time. Each packet contains only essential information, such as vehicle ID, position, and relevant sensor data, to optimize bandwidth efficiency. The server asynchronously forwards received data to other clients within range, facilitating real-time inter-vehicle data sharing and supporting connected driving.

4.4.3 Position Management

To emulate inter-vehicular communication, we developed a neighbor selection module, which determines the communication range between vehicles based on their spatial proximity. Each client periodically sends its position to the server, which maintains an up-to-date map of all ego vehicles positions. The server calculates the distance d between any two vehicles using the Euclidean distance in the 3D Cartesian space defined in CARLA:

$$\mathrm{d} = \sqrt{(\mathrm{x}_2 - \mathrm{x}_1)^2 + (\mathrm{y}_2 - \mathrm{y}_1)^2 + (\mathrm{z}_2 - \mathrm{z}_1)^2} \tag{4.1}$$

where (x_1, y_1, z_1) and (x_2, y_2, z_2) represent the positions of two vehicles. If $d \leq d_{max}$, where d_{max} is the maximum communication range, the two vehicles are considered within the communication range.

4.4.4 Sensor Data Serialization

For efficient sensor data sharing between vehicles, we use OpenCV [1], a widely used library for computer vision and data handling. Sensor data, such as raw RGB images, LiDAR, and radar data, is prepared before transmission. The data is serialized into a binary format, enabling fast transmission over the socket connection without additional data compression. This approach ensures that raw sensor information is retained, allowing for high-fidelity data exchange across vehicles.

[1] **OpenCV. Accessed: June. 9, 2023:** https://github.com/opencv/opencv

4.4.5 Data Transmission and Visualization

Each client receives raw sensor data from nearby vehicles, processes it using OpenCV, and displays the information in real time. This visualization emulates the perception of each ego vehicle in its environment, enhancing situational awareness. By handling socket data asynchronously, the system supports multiple image streams simultaneously, enabling real-time, multi-vehicle sensor sharing for cooperative perception. An example of such a visualization is shown in Figure 4.2.

4.5 Results

To thoroughly evaluate the performance of our proposed framework, we conducted a series of simulations using the Town01 3D environment in CARLA. Town01 is designed to mimic real-world urban settings and includes a range of dynamic and static elements that contribute to a realistic simulation. Dynamic elements include various types of vehicles, motorcycles, bicycles, and pedestrians. These entities interact with static infrastructure elements, such as traffic signs, street trees, and buildings, creating a rich environment for testing vehicular communication and perception.

4.5.1 Vehicular Communication and Scalability

In our initial simulation scenario, we configured CARLA to simulate five ego vehicles. Each vehicle operated as an independent client within the environment. This setup was designed to test the robustness and scalability of our communication framework to manage simultaneous data exchange. To evaluate performance, each vehicle periodically broadcasted a simple beacon message at a frequency of 1 Hz. These transmissions allowed us to track the communication range and assess scalability across multiple clients.

The simulation was executed on a laptop equipped with an Intel Core i9 2.4GHz CPU, 32GB of RAM, and a Nvidia GeForce GTX 1650 GPU. Despite this relatively limited hardware, the framework handled the simulation smoothly. This demonstrates its efficiency and adaptability in multi-client scenarios. We set a communication range threshold, d_{max}, at 200 meters to represent a realistic limit for vehicle-to-vehicle (V2V) communication in urban settings. The simulation ran continuously for 240 minutes. During this period, each ego vehicle navigated the Town01 road network in loops, adhering to standard traffic regulations to simulate realistic driving patterns.

Throughout the simulation, we monitored each vehicle communication performance. This included recording the number of beacons received from other vehicles and measuring how long each vehicle stayed within the communication range of the others. Table 4.1 presents these results in a matrix format. A value of 1 between two vehicles indicates that they remained within the communication range for the entire duration of the simulation.

	Car_1	Car_2	Car_3	Car_4	Car_5
Car_1	–	0.634	0.740	0.479	0.832
Car_2	0.634	–	0.743	0.837	0.671
Car_3	0.740	0.743	–	0.947	0.847
Car_4	0.479	0.837	0.947	–	0.768
Car_5	0.832	0.671	0.847	0.768	–

TABLE 4.1: Contact duration between vehicles as a proportion of the total simulation time, indicating the length of time pairs of vehicles remained within communication range throughout the simulation. **Source:** Reprinted from [55], with permission, ©2021 IEEE.

Our findings show that, due to the compact layout of the simulated urban area, most vehicles stayed within range of each other for at least 50% of the simulation time. This supports high communication availability within the network.

4.5.2 Data Transmission and Bandwidth Feasibility

To assess the scalability of the framework with various sensor types, we tested the bandwidth requirements for transmitting raw data. In the experimental setup, data was transmitted at a uniform frequency of 1 Hz across all policies to maintain consistency among sensor types and configurations. The policies included standalone transmissions of individual sensors and a multi-sensor policy that combined data from all three types. Fig.4.3 illustrates the bandwidth requirements observed for each policy. As expected, bandwidth demands vary significantly depending on the type and configuration of the sensor. For example, even at lower resolutions, raw camera data requires substantial bandwidth around 44 Mbit/s per stream. Higher resolutions amplify these demands, reaching over 330 Mbit/s for six images from multiple cameras. LiDAR data, known for detailed 3D spatial information, require between 84 and 167 Mbit/s based on resolution and point cloud density. Radar data, which offer coarser spatial information, requires fewer bandwidths, typically between 27 Mbit/s and 72 Mbit/s.

The combined multi-sensor policy, which involves the simultaneous transmission of raw data from cameras, LiDAR, and Radar, demands a peak bandwidth of 575 Mbit/s. This substantial requirement results from the high-resolution data and the real-time frequency at which each sensor captures information. In complex urban environments with high object density and dynamic scenes, this demand increases further. Such bandwidth requirements significantly exceed the typical capacities of current technologies.

For example, the ITS-G5 protocol, widely adopted in vehicular networks, provides a maximum data rate of 54 Mbit/s [140]. Similarly, while 4G networks can support up to 100 Mbit/s under ideal conditions, this rate remains insufficient for high-bandwidth applications like raw multi-sensor data streaming. Even with the advanced capabilities of 5G, which supports peak rates up to 500 Mbit/s [112], the transmission of uncompressed raw data from multiple sensors remains impractical. This is especially true when considering network fluctuations and the low-latency requirements of real-time perception.

Sensor	Policy	Data Format
Camera	CAM1	6 RGB images, res. 640×480
	CAM2	6 RGB images, res. 1260×720
	CAM3	6 RGB images, res. 1928×1208
LiDAR	LID1	4D points, 2,621,440 points per second (pps)
	LID2	4D points, 5,242,880 points per second (pps)
Radar	RAD1	4D points, 1,500 points per second (pps)
	RAD2	4D points, 35,000 points per second (pps)
All	CLR1	CAM3 + LID2 + RAD2

TABLE 4.2: Transmission policy properties and data formats. For further details on format specifications, refer to [32]. **Source:** Reprinted from [55], with permission, ©2021 IEEE.

Consequently, to make multi-sensor data transmission feasible within current and next-generation networks, alternative strategies are essential. These include data compression techniques, selective data transmission policies, and adaptive downsampling to reduce overall data volume while preserving key information. Another approach is feature extraction, where only high-level features are transmitted, rather than raw data, significantly conserving bandwidth. Together, these strategies aim to optimize the usage of network resources, enabling scalable and reliable vehicular communication systems capable of supporting advanced perception and autonomy requirements.

4.5.3 Data Compression Strategies

Given the substantial bandwidth requirements, achieving efficient V2V perception data sharing requires optimization techniques to align communication demands with network constraints. Data compression is a key strategy that can reduce bandwidth requirements by approximately 25% in some cases [110]. By compressing raw data prior to transmission [29, 134], vehicles can reduce the data load while retaining essential information, making it feasible to operate within the bandwidth limits of current communication standards.

Beyond compression, selective data transmission offers an adaptive approach by sharing only the most relevant information, such as dynamic changes critical to the driving scenario. This approach is further enhanced by local data processing, where raw sensor data is partially processed onboard to extract high-level features, which are then shared with nearby vehicles. By minimizing the need for raw data exchange, this method significantly reduces the total data volume transmitted over the network [104, 18]. Transmitting only essential features, rather than raw sensor data, enables the network to support more vehicles simultaneously without compromising real-time responsiveness.

Together, these optimization strategies are essential for enabling real-time cooperative perception within the bandwidth limitations of current and emerging vehicular networks. They play a critical role in practically implementing the proposed connected driving ecosystem, bridging the gap between ambitious perception goals and the realistic constraints of communication infrastructure.

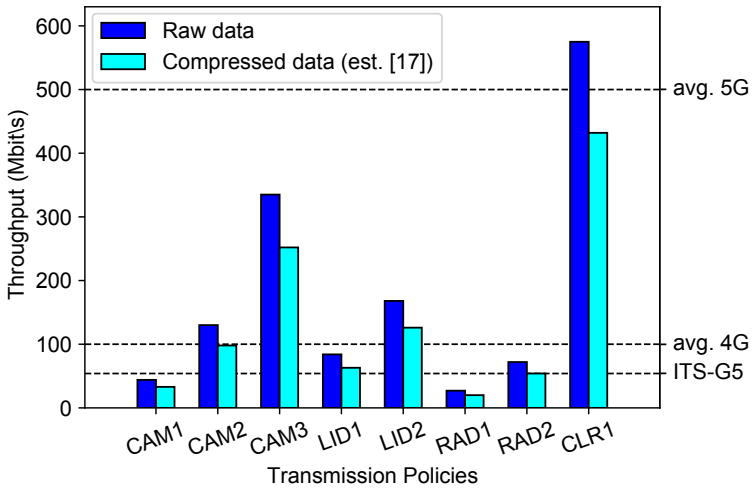


FIGURE 4.3: Required throughput for transmitting various types of raw and compressed data across different sensor types and configurations, including camera, LiDAR, and Radar. **Source:** Reprinted from [55] with permission, ©2021 IEEE.

4.6 Conclusion and Future Work

In this chapter, we introduce a framework designed to facilitate cooperative perception within a simulated environment, using the CARLA client-server architecture, an open-source simulator tailored specifically for autonomous driving research. This setup enables efficient sensor data sharing among multiple ego vehicles (clients), allowing us to model and test cooperative perception strategies in a controlled, reproducible environment. Through a straightforward V2V communication scenario, we demonstrate that multiple ego vehicles can effectively exchange sensor information, highlighting the feasibility of this setup for cooperative perception tasks in CAV.

Our results show that transmitting raw perception data imposes substantial bandwidth demands. These demands often exceed the capabilities of current vehicular communication technologies. This finding highlights the need for bandwidth optimization techniques to achieve real-time perception sharing within existing network constraints. The platform combines a realistic driving simulation environment with a robust communication framework. This integration offers a valuable tool for future research, reducing the complexity and cost traditionally associated with evaluating cooperative perception systems.

Part III

Edge and Cloud Computing for Cooperative Perception

Chapter 5

Edge and Cloud Computing for Object Detection

This chapter investigates data offloading strategies to remote edge or cloud infrastructures to address the computational limitations of onboard vehicle systems. By offloading perception sensor data processing, we leverage the superior computational power of edge and cloud platforms, enhancing perception capabilities without overloading local resources. However, offloading also introduces transmission latency, which can challenge the strict latency requirements of autonomous driving.

Emerging technologies like C-V2X and 5G offer promising solutions to address latency challenges and support efficient data offloading for real-time perception. To further reduce transmission delays, we apply H.265 and JPEG compression at various quality levels to evaluate their impact on accuracy and latency. Our evaluation uses the CARLA simulator to generate environmental data and OMNeT++ to measure network latency.

This chapter builds upon the following publications:

Faisal Hawlader, François Robinet, and Raphaël Frank. "Leveraging the edge and cloud for V2X-based real-time object detection in autonomous driving". In: *Computer Communications*. Elsevier, 2024. DOI: 10.1016/j.comcom.2023.11.025

Faisal Hawlader, François Robinet, and Raphaël Frank. "Vehicle-to-Infrastructure Communication for Real-Time Object Detection in Autonomous Driving". In: *18th Wireless On-Demand Network Systems and Services Conference (WONS)*. IEEE, 2023. DOI: 10.23919/WONS57325.2023.10061953

This chapter is structured as follows: Section 5.2 reviews the relevant literature. Section 5.3 outlines the hardware setup, network simulation parameters, and model training and evaluation procedures. Section 5.5 presents the experimental findings, emphasizing the offloading trade-offs. Lastly, Section 5.6 summarizes the contributions of the chapter and proposes future research directions.

5.1 Introduction

Achieving precise environmental awareness is essential for safe decision making and route planning in autonomous vehicles, which is dependent on real-time processing of data from sensors such as cameras, LiDAR, and radar. However, the computing limitations of the onboard hardware require balancing processing speed and detection quality.

To address these constraints, offloading some processing tasks to external platforms, such as Multi-access Edge Computing (MEC) or cloud servers, offers an alternative with greater computational power [27]. Although cloud offloading can enhance perception capabilities, it introduces transmission delays, which may be unsuitable for time-sensitive applications [55]. Recent advances in C-V2X technology [114] and 5G [39] provide opportunities for optimized selective offloading.

Our approach offloads computationally intensive perception tasks to MEC or cloud resources while keeping lighter tasks onboard to balance detection accuracy and latency. To mitigate transmission delays, we assess the impact of H.265 and JPEG compression [105, 28] on transmission latency and accuracy. Using different quality levels, we optimize the detection performance in edge and cloud configurations to achieve a target rate of 20Hz [1]. Tests with various YOLOv5 models reveal a trade-off, that is, larger models increase accuracy but may exceed the 50 ms latency threshold [57].

Our evaluation combines real hardware with simulations, using CARLA [32] for data generation and OMNeT++ [162] to simulate network latency. This hybrid approach bolsters our findings, advancing real-time perception strategies for autonomous driving.

We make the following key contributions in this chapter:

Contributions

- Using the CARLA simulator [32], we create a synthetic dataset to train and validate object detection models, enabling assessment of our offloading strategies. This dataset is open-sourced to encourage reproducibility and collaboration within the research community[a].
- We examine camera frame transfer and processing on edge and cloud platforms, utilizing real hardware to measure processing latency and network simulation to assess transmission latency, providing insights into the trade-off between prediction accuracy and end-to-end delay.
- To minimize transmission latency, we propose an integrated framework utilizing H.265 and JPEG streaming for C-V2X-based real-time object detection, achieving reduced latency for camera data transmission while maintaining high detection quality.

[a] **Dataset available:** https://zenodo.org/records/14534852

	Model Size		
	Small	Large	Large (high-res)
All (mAP)	0.64	0.66	0.85
Pedestrian	0.30	0.36	0.81
Traffic light	0.80	0.82	0.86
Vehicle	0.79	0.81	0.89

TABLE 5.1: Average Precision results for different model variants (AP@50). Model variants are detailed in Section 5.3.1. **Source:** Reprinted from [59], with permission, ©2023 IEEE.

5.2 Related work

Advances in sensor technology, communication technologies, and data processing are accelerating the development of autonomous driving. In this field, accurate real-time perception is essential for both safety and efficiency. This literature review explores the foundational technologies that support robust perception in autonomous vehicles. It focuses on three core areas: Vehicle-to-Infrastructure (V2I) communication for data offloading, object detection models and how data compression affects accuracy, and perception capabilities enabled by C-V2X communication. Each section examines the trade-offs and challenges in achieving low-latency, high-precision perception for autonomous driving.

5.2.1 V2I communication

Autonomous vehicles often utilize V2I communication to transfer sensor data processing tasks to a dedicated server, as shown in studies [65, 73]. This server, which may be positioned at the network edge using 5G MEC technology [23] or in a more powerful cloud environment [79], handles the computationally intensive tasks. V2I communications function through an upload/download channel [114], enabling the vehicle to offload data to an edge or cloud server. The server processes the data and then returns the results to the vehicle. In this approach, the vehicle onboard system focuses on essential pre-processing steps, such as data compression or encoding, before transmission [133]. Offloading sensor data to the cloud through V2I introduces additional transmission latency, as noted in [147]. To reduce this latency, MEC can be used, offering lower transmission delays compared to traditional cloud services [114]. However, edge devices have inherent limitations in computational power and storage capacity [44, 3]. These constraints may restrict their ability to support applications with high performance demands [104]. Therefore, further investigation is essential to determine whether edge computing can meet the stringent latency and computational requirements of autonomous vehicle applications, while effectively balancing resource limitations.

5.2.2 Object detection and impact of compression

Object detection: Early advances in object detection focused on designing hand-crafted feature extraction methods to identify meaningful patterns in image data [92]. These extracted features served as input to a range of object detectors [31], enabling initial breakthroughs in the field [175, 139]. Among the most influential early methods was the Viola-Jones face detector [95], which introduced the use of Haar-like features [157] to achieve rapid face detection. Another foundational approach was the Histogram of Oriented Gradients (HOG) detector [26], which utilized gradient orientation histograms to detect objects. Both methods demonstrated that specific, manually designed features could effectively highlight essential visual cues [49], laying the groundwork for subsequent advances in automated object detection [51]. However, while effective, these early detectors were limited by their dependence on manually crafted features [133], which could not easily adapt to more complex or varied visual data.

In recent years, object detection has advanced significantly with the rise of detectors based on deep neural networks [123, 122], which are generally classified into two primary categories: two-stage [46] and single-stage models [44]. Two-stage detectors first generate region proposals to identify potential object locations within an image. These proposals are then refined in a second stage by a trained model to confirm the presence of objects and classify them accurately [46, 45, 125]. While two-stage models, such as R-CNN [46], Fast R-CNN [45], achieve high levels of detection accuracy [125], their computational complexity and multi-step process limit their applicability for real-time operations [51].

To overcome the computational limitations of two-stage models, single-stage detectors such as SSD [95] and the YOLO family [122] have been developed. These models streamline the detection process by merging the proposal of the region and classification into a single operation [38], allowing faster [44], real-time performance [50]. In the YOLO framework, for example, the input image is divided into a grid of cells [74], each cell being responsible for predicting the bounding boxes and identifying objects within its area. YOLO is trained end-to-end with a loss function that optimizes multiple objectives [75], including the accuracy of the bounding box [139], the confidence scores of the object and the labels of the class. This integrated approach improves processing speed and makes single-stage models particularly suited for real-time applications. In our study, we utilize YOLOv5 [75], a refined version of the original YOLO architecture, selected for its ability to balance real-time processing capability with improved accuracy and efficiency [150, 110]. YOLOv5 introduces architectural optimizations that reduce computational load while enhancing object detection precision [16], making it particularly suitable for applications requiring both speed [33] and reliability in dynamic environments [157, 18].

Impact of compression on detection: To enable faster data transmission in cloud-based inference [50, 16], a prior study investigated the impact of H.265 (video) [61, 59] and JPEG (image) compression on object detection accuracy across various quality levels [57]. The research shows that compression can also reduce detection accuracy [158], particularly due to information loss when compression is too high [31]. This reduction becomes more pronounced at lower quality settings, where moderate to high compression leads to a sharp decline in precision [33].

Recently, H.265 has gained attention for its ability to maintain video quality with more efficient compression [137], resulting in smaller data sizes compared to older formats [165]. These properties make H.265 highly suitable for streaming sensor data. However, despite its efficiency, high-quality H.265 streams can still require significant bandwidth [103], which may challenge real-time streaming to edge or cloud [144]. Bandwidth limitations in these scenarios can compromise both streaming quality and detection performance. Therefore, finding an optimal balance between the low latency and the compression advantages of H.265 and JPEG remains critical [51]. Further research is needed to achieve the best trade-off between transmission speed and detection accuracy.

5.2.3 Perception using C-V2X communication

Recent advancements in 5G technology [112], especially in edge and cloud computing [5], have created new possibilities for the adoption and experimentation of C-V2X technology [1, 23]. Despite its potential [133], studies indicate that the deployment of C-V2X remains largely in the experimental phases, with larger-scale applications anticipated as the 5G infrastructure evolves [1]. C-V2X is projected to support advanced applications [13], including cooperative perception [170], where vehicles share sensory data to improve awareness of surrounding objects [150, 81].

Research highlights that local processing of perception sensor data on computer systems on-board vehicles can be limited by the available computational resources [43], making it challenging to meet the demands of real-time perception [79]. To address these limitations, studies have explored V2X based cooperative and distributed perception, which enables vehicles to offload computationally intensive tasks to edge and cloud servers [84, 73]. Findings in recent studies suggest that MEC and cloud platforms are viable solutions for perception data processing in autonomous vehicles, providing the computational power necessary to support these applications [114, 79, 133, 73].

However, real-time perception demands low-latency processing [173], which depends significantly on where the data is processed onboard the vehicle, at the edge, or in the cloud [124]. Several studies have proposed different sensor data offload strategies [27, 101], ranging from raw data to partially or fully processed data [104], to reduce network load [81, 11, 104]. Although these approaches are designed to minimize transmission overhead [24], they often overlook the potential impact on perception accuracy [14]. For instance, while [166] demonstrates that offloading raw sensor data can improve detection accuracy [133], it can also substantially increase transmission costs [55]. These findings underscore the need for balanced strategies that account for both perception quality and transmission efficiency in V2X systems [68]. Transmitting compressed data can conserve network resources but may also lead to a reduction in detection accuracy [124]. To the best of our knowledge, there is a lack of comprehensive studies that explore the balance between detection quality and end-to-end processing time. This chapter aims to address this gap.

Platform	Scenario / Model	Hardware configuration
Local (≈20W)	YOLOv5 small 157 layers, 7M params 640x640 Resolution	NVIDIA Jetson Xavier NX SoC Volta GPU, 384 CUDA cores Carmel ARMv8.2 CPU@1.9GHz
Edge (≈100W)	YOLOv5 large 267 layers, 46M params 640x640 Resolution	Laptop with GeForce GTX 1650 Turing GPU, 896 CUDA cores Intel i9-9980HK @2.4GHz
Cloud (≈450W)	YOLOv5 large high-res 346 layers, 76M params 1280x1280 Resolution	HPC node with Tesla V100 Volta GPU, 5120 CUDA cores Intel Xeon G6132 @2.6 Ghz

TABLE 5.2: To meet the 50 ms constraint on inference time, we chose specific platforms for running three models of varying sizes. Figure 5.1 presents a comparison of the inference times across these models and platforms. **Source:** Reprinted from [59], with permission, ©2023 IEEE.

5.3 Methodology

This section outlines our preliminary approach to estimating the end-to-end delay in real-time object detection. Following prior research, our goal is to achieve object detection at a frequency of 20 Hz while preserving detection quality [1]. The end-to-end delay in this process is heavily impacted by the computation platform. When the vehicle offloads data to the edge or cloud, it must transmit raw data, which can exceed the bandwidth limitations of current V2X technologies [55].

5.3.1 Motivation & Hardware

Real-time object detection within autonomous vehicles presents significant challenges due to the limited computing power and energy constraints associated with onboard hardware. Given these limitations, processing tasks are offloaded to external platforms with greater computational resources, such as edge and cloud servers. However, while this approach alleviates the onboard vehicle processing burden, it introduces additional complexities in maintaining high detection quality and achieving minimal inference delays, both of which are crucial for responsive and accurate object detection [5].

To better understand and address these challenges, we designed and conducted a series of experiments utilizing a range of hardware setups. Table 5.2 outlines these configurations, distinguishing between local processing (performed on the onboard device) and processing on edge and cloud platforms. The onboard device, representing local processing, is constrained by limited processing power compared to the more capable edge and cloud infrastructure, where computational resources are more abundant. This setup aligns with prior research on distributed computing approaches in vehicular networks [115, 133].

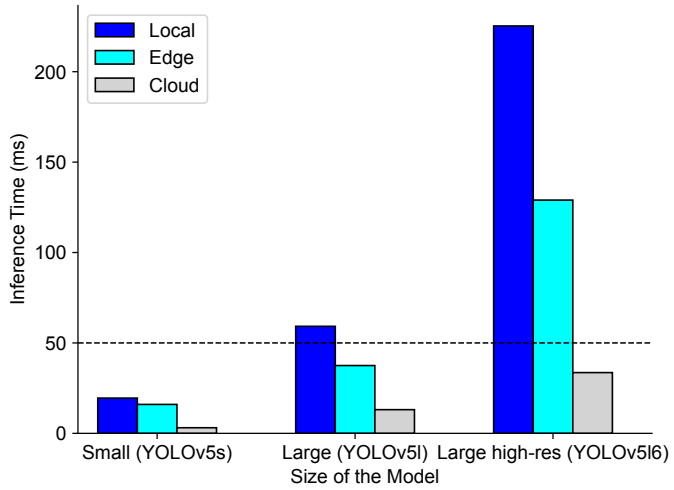


FIGURE 5.1: Comparison of inference times for different model sizes across various hardware platforms. We employ half-precision floating-point computation to enhance processing speed. The dashed line represents the target latency of 20Hz required for real-time object detection in autonomous driving scenarios. **Source:** Reprinted from [59] with permission, ©2023 IEEE.

In this chapter, we selected YOLOv5 [75] for object detection due to its excellent balance of accuracy and low-latency performance, as demonstrated on recent benchmarks [75, 172]. YOLOv5 offers models in multiple sizes, from smaller, faster models to larger, more accurate ones. While larger models generally yield better detection quality, they also have greater computational requirements, which can impact inference time and overall performance. Thus, we aim to identify the best-performing model for each platform that also meets our real-time constraints.

Figure 5.1 illustrates the inference times of various model sizes across the different platforms, allowing us to determine the optimal deployment for each model size. The large model, with an inference time exceeding 50ms, is unsuitable for the onboard hardware due to latency constraints. To achieve the target 20Hz detection rate, we selected the smaller YOLOv5 model for local hardware, while utilizing the larger model on the edge platform and the high-resolution large model on the cloud, where computational resources are sufficient to handle the added demand. Table 5.2 summarizes the specific YOLOv5 versions and input resolutions employed for each platform.

5.3.2 Networking Aspects

This section provides an in-depth overview of the 5G Radio Access Network (RAN) components and explains how we utilized network simulations to assess end-to-end delays in a real-time object detection model supported by MEC and cloud infrastructure. We employed Simu5G [114], a discrete-event network simulator built on OMNeT++ [153], to model the 5G data plane. Our simulations are based on the C-V2X Release 16 standard, which enables advanced V2X communication through 5G technology. The setup explores two main scenarios: offloading perception data to MEC and offloading to cloud infrastructure, both utilizing C-V2X for connectivity.

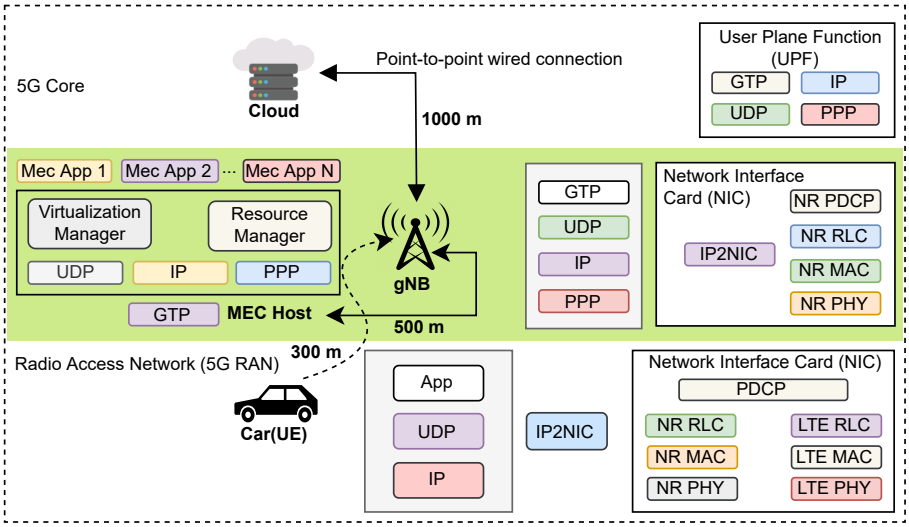


FIGURE 5.2: Network simulation architecture, illustrating key elements of the 5G RAN, including the MEC host-level components and the User UE. **Source:** Reprinted from [59] with permission, ©2023 IEEE.

Our simulated network includes both a 5G RAN and a 5G Core Network (CN). The RAN consists of a single 5G base station (gNB), which serves as the primary connectivity point for the vehicle or user equipment (UE) [35]. A MEC server, situated 500 meters from the gNB and linked by a high-speed wired connection, facilitates rapid data exchange, allowing for swift interaction between the MEC and gNB. The gNB also connects to a cloud server via the 5G CN, creating an end-to-end data pathway for offloading object detection tasks. The architecture of this setup is illustrated in Figure 5.2.

In our experiment, the vehicle was kept stationary to simplify latency measurements. Although this static setup does not capture the full range of real-world conditions where vehicles are in motion, it allows for a focused examination of network latency. We believe this controlled approach provides a stable baseline for understanding latency patterns, acknowledging that dynamic conditions may introduce further variability.

5G Core Network (CN): We configured the 5G core network as a standalone system to meet our simulation requirements [113]. This setup includes a User Plane Function (UPF) that links the RAN to the cloud via a wired interface. The gNB and cloud communicate using the GPRS Tunneling Protocol (GTP), which enables efficient routing of IP datagrams (UDP) across the network. A Point-to-Point (PPP) interface connects the gNB to the cloud, providing a dedicated wired pathway for streamlined data flow [162].

Multi-access Edge Computing (MEC): MEC plays an increasingly critical role in the 5G ecosystem, with ongoing standardization efforts to facilitate its integration into 5G networks [37]. Our work employs a simplified MEC host setup based on the ETSI reference architecture [23]. The MEC server is co-located with the gNB, as shown in Figure 5.2, and includes essential modules to ensure seamless MEC operations. Applications run in a virtualized environment on the MEC server, with a Resource Manager orchestrating their lifecycle. Additionally, a Virtualization Manager allocates and manages the computing, storage, and networking resources required by MEC applications.

Parameter Name	Value
Carrier frequency	3.6 GHz [39]
gNB Tx Power	46 dBm
Path loss model	[114], Urban Macro (UMa)
Fading + Shadowing model	Enable, Long-normal distribution
Number of repetitions	200
Path loss model	3GPP-TR 36.873 [120]
UDP Packet size	4096 B [92]
Throughput	113.94 Mbit/s (Avg.)
Numerology (μ)	3
Latency (Vehicle-to-Edge)	0.43 ms (Avg.)
Latency (Vehicle-to-Cloud)	0.45 ms (Avg.)
Packet loss ratio	0.0001

TABLE 5.3: Summary of key network parameters for a stationary test. Averages are calculated over 200 repetitions. **Source:** Reprinted from [57], with permission, ©2023 Elsevier.

The MEC host supports GTP, providing flexible deployment options within the network. Consistent with prior studies [114], the MEC server is positioned 500 meters from the gNB and connected via a high-capacity 100 Gbps PPP link [161].

5G Base Station (gNB): In this simulation, the gNB is configured with protocol support up to Layer 3 and equipped with dual network interfaces: one for PPP wired connectivity to the core network and another for wireless radio access. The PPP interface connects to the core network via GTP, aligning with the CN protocol structure. The radio access interface is layered for efficient data transmission. The Packet Data Convergence Protocol (PDCP) layer receives IP datagrams, encrypts them, and sends them to the Radio Link Control (RLC) layer. Data units are then temporarily stored in the RLC buffer until needed by the Media Access Control (MAC) layer for transmission. The MAC layer assembles data into transport blocks, adds a MAC header, and forwards the packet to the Physical (PHY) layer for over-the-air transmission. For further details, refer to the Simu5G documentation [114].

User Equipment (UE): As defined by ETSI [23] and 3GPP standards [7], UE refers to any device operated by the end user. The UE is represented by a connected vehicle equipped with C-V2X protocol stacks and linked to the gNB. C-V2X, a 3GPP-defined technology for connected mobility applications, is compatible with 5G New Radio (NR) and supports bidirectional V2I communication. This configuration is crucial for our focus on offloading perception data to edge and cloud platforms for real-time object detection. The UE is equipped with dual network interface cards (NICs) to support both LTE [79] and 5G NR [150] connectivity, as shown in Figure 5.2.

The network configuration described here is to thoroughly capture end-to-end latency in real-time object detection scenarios. By leveraging both edge and cloud resources within a 5G network, this setup enables an evaluation of network capability to support demanding, low-latency applications essential for autonomous vehicle functionality.

	Class		
	Pedestrians	**Traffic Lights**	**Vehicles**
Train	12916	43418	33351
Validation	2164	8272	11295
Test	1756	11115	7897
Total	16836	62805	52576

TABLE 5.4: Summary of dataset composition and instance counts for each class. **Source:** Reprinted from [57], with permission, ©2023 Elsevier.

5.3.3 Perception Aspects

Dataset Generation: The YOLOv5 models outlined in Table 5.2 were trained using a synthetic dataset that we generated with the CARLA simulator [32]. CARLA enables us to simulate a virtual environment where a camera-equipped vehicle navigates through a realistic town setting under various conditions. For the purposes of this study, we focused exclusively on clear daylight weather, providing consistent lighting conditions across the dataset. Additionally, CARLA offers ground truth annotations, specifically 3D bounding boxes, for three target object classes: vehicles, pedestrians, and traffic lights.

During the simulation, we captured 10,000 frames from the front-facing camera of the ego-vehicle, recording one frame per second (1 Hz). Alongside these images, we obtained ground truth 3D bounding boxes, which were projected onto the 2D image plane. This projection enabled the generation of 2D bounding boxes compatible with YOLOv5 training requirements, as it operates on 2D object detection.

To prepare the dataset for model training and evaluation, we split it into three subsets: 6,000 frames for training, 2,000 frames for validation, and 2,000 frames for testing. Table 5.4 summarizes the instance distribution across classes and dataset splits, with the first column indicating the intended use of each subset.

A notable observation is the imbalance in class distribution, with fewer pedestrian instances compared to vehicles and traffic lights across all subsets. This imbalance could affect the performance of YOLOv5 models trained on this dataset, potentially reducing detection accuracy for pedestrians. To address this, additional dataset balancing through either more data collection or data augmentation may be needed to improve the model proficiency across all classes, ensuring more reliable and representative performance.

5.3.4 Exploring Compression Settings

We investigate the impact of various compression levels on the data volume transmitted to the edge or cloud, aiming to balance file size with detection quality. By analyzing different compression settings, we aim to identify a resource efficient configuration that optimizes network usage and minimizes transmission latency. This approach aims to maintain high detection accuracy while efficiently managing bandwidth and reducing network load.

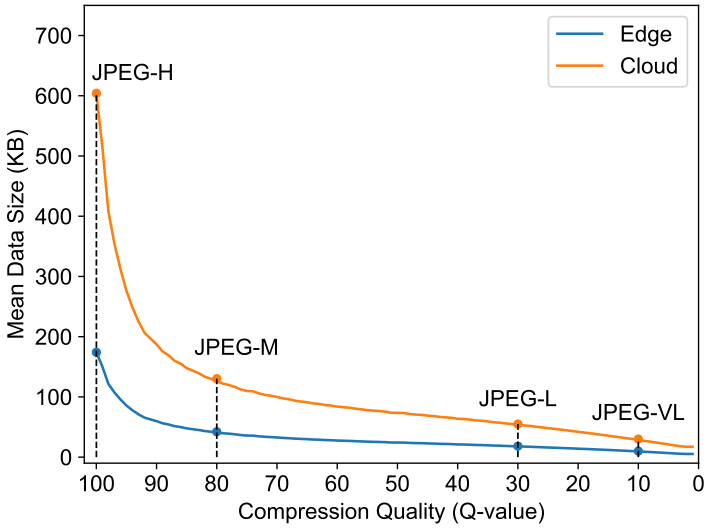


FIGURE 5.3: Average data sizes for JPEG compression levels high (JPEG-H), medium (JPEG-M), low (JPEG-L), and very low (JPEG-VL) for image resolutions of 640×640 (edge processing) and 1280×1280 (cloud processing).
Source: Reprinted from [57], with permission, ©2023 Elsevier.

JPEG Compression is a widely used algorithm valued for balancing image quality with data size reduction. A lossy compression method is employed, selectively discarding certain visual information to reduce file size. The level of compression is regulated by a quality parameter, referred to as the *Q-value*, which varies between 0 and 100. Higher *Q-values* preserve more image details but produce larger file sizes, whereas lower values sacrifice detail to achieve greater compression.

In this study, we evaluated JPEG compression across various quality levels, from high quality (JPEG-H) to very low quality (JPEG-VL), using default settings for other compression parameters. Figure 5.3 shows the average data sizes for each JPEG quality level. To structure our analysis, we defined four specific scenarios: JPEG-H (high quality) with a *Q-value* of 100, JPEG-M (medium quality) at 80, JPEG-L (low quality) at 30, and JPEG-VL (very low quality) at 10. This range of compression scenarios allows us to observe how quality affects data volume across different image resolutions: 640x640 for edge processing and 1280x1280 for cloud transmission. This analysis helps identify configurations that reduce data size while maintaining the visual quality.

H.265 Compression applying H.265 compression to stream camera sensor data requires adjusting several configurations and parameters, each impacting compression quality, latency, and overall performance. Fine-tuning these parameters is essential to balance image quality and transmission delay, which is particularly important in real-time applications like autonomous vehicle perception.

Key parameters include the Constant Rate Factor (CRF), encoding presets, and lookahead settings, each providing distinct control over the trade-offs in encoding. The following sections examine these parameters in detail, focusing on their impact on the efficiency and latency of H.265 compression.

- **H.265 Frames:** H.265 encoding utilizes three frame types to optimize compression efficiency: I-frames (intra-coded), P-frames (predictive), and B-frames (bidirectional predictive), each playing a distinct role. I-frames, or key frames, serve as reference points, enabling subsequent P-frames and B-frames to compress data by referencing them. I-frames are compressed by encoding minor variations between adjacent pixels, leveraging pixel similarity to reduce data size.

  P-frames are not self-contained, they encode only changes from the previous I-frame or P-frame, achieving high compression by representing object movement. B-frames improve efficiency further by referencing both preceding and succeeding frames, yielding smaller data sizes and higher quality. However, this added complexity increases encoding latency. In latency sensitive applications, such as real-time object detection, B-frames are often excluded to reduce latency. In these cases, only I-frames and P-frames are used, streamlining encoding for optimal performance.

- **Lookahead:** The lookahead function enables the encoder to analyze future frames before encoding the current frame, improving compression efficiency and data quality. However, this introduces latency, making it impractical for real-time applications like autonomous driving. Disabling lookahead allows the encoding process to rely only on I-frames and P-frames. Although less efficient than configurations with B-frames, this approach satisfies the latency constraints of real-time streaming.

- **Constant Rate Factor (CRF):** CRF is a quality control variable that adjusts visual quality based on frame complexity and motion in H.265 encoding. CRF values range from 0 to 51, with lower values indicating higher quality but larger data sizes. Adjusting CRF helps balance desired image quality with data size for transmission to edge or cloud processing nodes. Lower CRF values produce high quality images requiring higher bandwidth, while higher CRF values reduce data size by decreasing image quality, thereby optimizing bandwidth usage.

To identify optimal CRF values for our real-time object detection use case, we conducted a trade-off analysis. We disabled lookahead and excluded B-frames while keeping other encoding parameters at default settings, as implemented in FFmpeg[1].

By omitting B-frames and lookahead, we achieved a notable reduction in computational complexity and latency, which is beneficial for applications requiring immediate data transmission. Figure 5.4 presents the experimental results, illustrating the relationship between CRF values and resulting data sizes. Lower CRF values, indicating higher quality, correspond to larger data sizes, while higher CRF values reduce data size at the expense of quality. This correlation underpins the trade-off between quality and compression level, from high-quality (H.265-H) to very low quality (H.265-VL).

To facilitate further analysis, we defined four specific compression scenarios: H.265-H at CRF 0 for high quality, H.265-M at CRF 24 for medium quality, H.265-L at CRF 30 for low quality, and H.265-VL at CRF 51 for very low quality. These scenarios enable us to evaluate the impact of different CRF levels on data transmission sizes, using two image dimensions: 640x640 for edge processing and 1280x1280 for cloud processing.

[1] **Available:** https://github.com/FFmpeg/FFmpeg

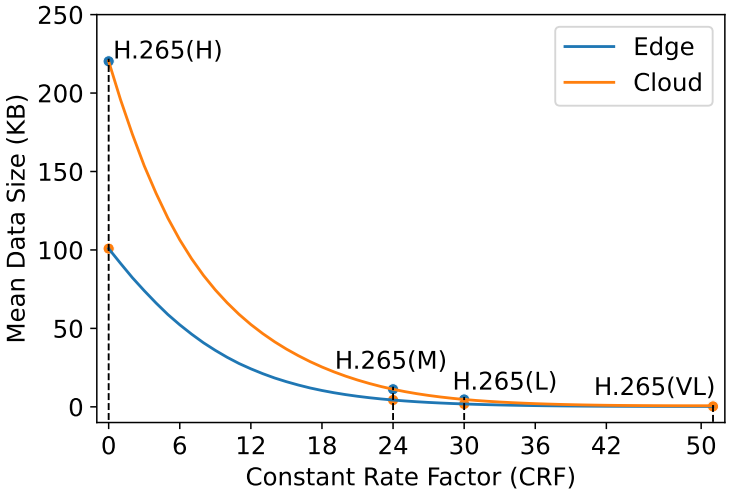


FIGURE 5.4: Mean data sizes across H.265 compression settings, including H.265-H (high quality), H.265-M (medium quality), H.265-L (low quality), and H.265-VL (very low quality). Evaluations were conducted at two image resolutions 640x640 for edge processing scenarios and 1280x1280 for cloud processing scenarios highlighting the impact of quality levels on data size.
Source: Reprinted from [57], with permission, ©2023 Elsevier.

5.4 Training Protocol

To train each model, we conducted 100 epochs using a single NVIDIA Tesla V100 GPU. We selected the Adam optimizer with an initial learning rate of 0.001 to facilitate efficient convergence. To maximize computational efficiency and expedite the training process, we adjusted the batch size for each experiment to the largest size that the GPU memory could support. This approach ensured full utilization of the GPU capabilities and maintained stable training performance across all configurations.

5.5 Results

This section presents the experimental evaluation of our proposed data offloading strategies. The experiments were conducted using the framework setup detailed in Section 5.3. Our analysis focuses on two critical performance indicators: end-to-end delay and detection quality. These metrics are essential for assessing the feasibility and efficiency of real-time object detection in autonomous driving environments, as they directly impact the vehicle ability to make timely and accurate decisions. Monitoring end-to-end delay enables us to identify latency-related bottlenecks, while assessing detection quality reveals potential compromises in object detection accuracy. Together, these insights help to clarify the strengths and limitations of various offloading strategies in supporting safe and responsive autonomous vehicle operations.

5.5.1 End-to-end delay evaluation

To evaluate the performance and latency trade-offs of the three scenarios outlined in Section 5.3 and Table 5.5, we begin by measuring their respective end-to-end delays.

In the local processing scenario, the end-to-end delay depends solely on the onboard hardware inference time. This measurement includes essential processing steps such as input preprocessing, non-maximum suppression (NMS), and the model forward pass. Incorporating these components, we observe a local end-to-end delay of 19.5 ms. This baseline latency serves as a reference point in our performance analysis, highlighting the efficiency of local data processing without data offloading.

In the second and third scenarios, we analyze the data offloading strategies between the vehicle and edge or cloud platforms. Network latency is assessed using the end-to-end simulation framework illustrated in Figure 5.2. Key network simulation parameters, including throughput and packet loss ratio, are detailed in Table 5.3. The end-to-end delay in these scenarios accounts for the processes of compression, transmission, decompression, and inference. While the time required to transmit detection results back to the vehicle is not explicitly measured, an additional latency of 0.43 ms is added for this transmission. This estimation is based on the premise that the raw detection results fit within a single packet. Table 5.3 further supports this assumption, indicating that a packet size of 4096 B requires an average of 0.43 ms to travel from the edge or cloud to the vehicle.

Transmitting raw, uncompressed frames to remote platforms introduces significant transmission delays due to the large data sizes involved. In our measurements, the vehicle-to-edge scenario exhibited an average end-to-end latency of 123.2 ms, which increased substantially to 521.7 ms in the vehicle-to-cloud scenario, as higher-quality frames were processed by the cloud model. These delays are impractical for most real-time perception applications. To address this challenge, we explored various compression strategies, as detailed in Section 5.3.4, to minimize the amount of camera sensor data transmitted over the network to the edge or cloud. Compression is performed locally on the vehicle, while decompression is executed on the edge or cloud platform, depending on the scenario. As shown in Table 5.5, these strategies significantly reduce frame sizes, enabling real-time remote object detection with the support of C-V2X communication.

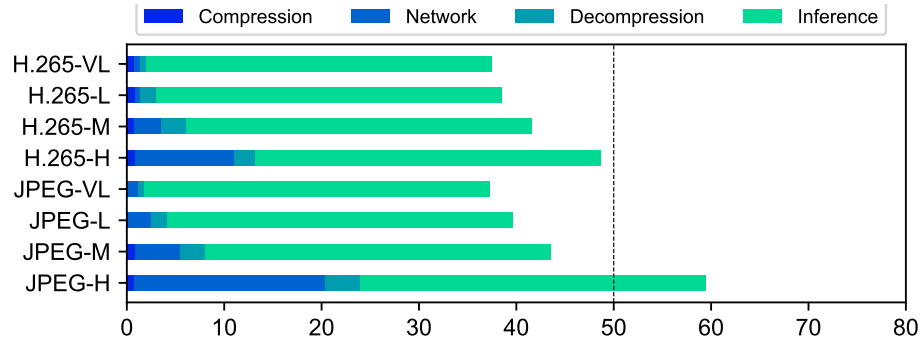
Table 5.5 provides a detailed analysis of data size and end-to-end delay for edge and cloud scenarios across different JPEG and H.265 compression quality levels. In the edge scenario, the uncompressed data size is 1.23 MB, resulting in an end-to-end delay of 123.20 ms. High-quality JPEG compression reduces the data size significantly to 174.12 KB (13.82% of the original size), lowering the delay to 59.48 ms. Further reductions in JPEG quality lead to smaller data sizes and shorter delays, with JPEG very low compression bringing the size down to 9.48 KB (0.75% of the original size) and reducing the delay to 37.27 ms. Similarly, H.265 compression shows remarkable efficiency by leveraging temporal relationships between frames. At its very low compression setting, the data size is minimized to just 0.26 KB (0.01% of the original size), achieving an end-to-end delay of 37.47 ms. These results highlight the potential of compression techniques to drastically reduce data transmission sizes and enable real-time perception.

Platform	JPEG Quality	Avg. data size (% of original)	End-to-end delay (ms)
Edge 640×640	No compression	1.23 MB (100%)	123.20
	JPEG-H	174.12 KB (13.82%)	59.48
	JPEG-M	40.78 KB (3.24%)	43.59
	JPEG-L	17.86 KB (1.42%)	39.62
	JPEG-VL	9.48 KB (0.75%)	37.27
	H.265-H	100 KB (2.00%)	48.65
	H.265-M	4.20 KB (0.09%)	41.61
	H.265-L	1.80 KB (0.04%)	38.51
	H.265-VL	0.26 KB (0.01%)	37.47
Cloud 1280×1280	No compression	4.92 MB (100%)	521.7
	JPEG-H	604.38 KB (12.00%)	74.50
	JPEG-M	125.51 KB (2.50%)	40.71
	JPEG-L	53.78 KB (1.13%)	32.93
	JPEG-VL	28.60 KB (0.6%)	29.42
	H.265-H	220 KB (4.37%)	46.93
	H.265-M	11.20 KB (0.22%)	30.21
	H.265-L	4.69 KB (0.09%)	28.72
	H.265-VL	0.69 KB (0.01%)	27.78

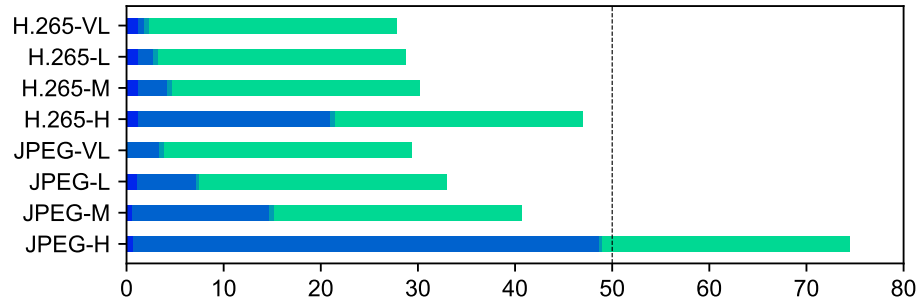
TABLE 5.5: Summary of data sizes and end-to-end delays for edge and cloud scenarios using various compression qualities. For a detailed breakdown of these delays, refer to Figure 5.5. **Source:** Reprinted from [57], with permission, ©2023 Elsevier.

In the cloud scenario, the uncompressed data size is significantly larger, starting at 4.92 MB and resulting in an end-to-end delay of 521.7 ms. However, as in the edge scenario, applying JPEG and H.265 compression effectively reduces both data size and delay. With very low JPEG compression, the data size decreases to 28.60 KB (0.6% of the original), reducing the delay to 29.42 ms. Similarly, H.265 compression at its very low setting reduces the size to just 0.69 KB (0.01% of the original), achieving a delay of 27.78 ms. Table 5.5 highlights the effectiveness of compression techniques in minimizing data size and latency for both vehicle-to-edge and vehicle-to-cloud scenarios, Figure 5.5 provides a detailed breakdown of end-to-end delays. Compression and decompression have a negligible impact on overall delay across all scenarios. Network transmission times increase with larger data sizes. Inference latency remains constant for a given platform since the same model and input resolution are used. Notably, all compression qualities, except for JPEG-H, enable real-time operation at 20 Hz on both platforms.

The breakdown of average end-to-end delay (ms) includes three key components: compression overhead, network transfer time, and inference time. The compression overhead represents the time required to compress data on the local device before transmission, while network transfer time accounts for the latency introduced during data transmission between the vehicle and edge or cloud platforms. Finally, inference time reflects the time taken by the remote platform to process the data and generate detection results. A dashed line in Figure 5.5 indicates the 20 Hz latency constraint, corresponding to a maximum allowable delay of 50 ms for real-time object detection. This line serves as a benchmark to evaluate whether different compression and offloading strategies can meet the stringent timing requirements for real-time perception. The following section explores the effects of compression on detection quality in detail.



(A) Car-to-Edge (large model, 640 × 640 input).



(B) Car-to-Cloud (large model, 1280 × 1280 input).

FIGURE 5.5: Average end-to-end delay breakdown: compression, transfer, and inference times, with a 20Hz latency constraint (dashed line).
Source: Reprinted from [57], with permission, ©2023 Elsevier.

5.5.2 Analyzing detection quality

To comprehensively evaluate detection quality, relying only on Precision and Recall metrics is insufficient. Instead, following standard object detection practices, we compute the Average Precision (AP), which represents the area under the Precision-Recall curve. A detection is considered a true positive if its Intersection-over-Union (IoU) with the ground truth bounding box exceeds 50%. To provide an overall metric across all classes, the AP values for individual classes are averaged, yielding the Mean Average Precision (mAP).

As discussed earlier, applying various compression techniques can significantly reduce end-to-end latency. However, excessive compression may negatively impact detection quality, resulting in a lower mAP. This section aims to find the optimal balance between detection accuracy and latency. The trade-offs are illustrated in Figure 5.6.

As expected, local processing achieves the shortest latency at 19.5 milliseconds with an mAP of 64%. Figure 5.6 also shows that the cloud platform consistently outperforms the edge platform in balancing mAP and end-to-end latency across comparable compression techniques. For instance, on the cloud platform using H.265-H compression, the mAP reaches 82% with an end-to-end delay of 46.93 milliseconds. Notably, with H.265-M compression, the mAP remains at 82%, but the delay is significantly reduced to 29.21 milliseconds. Despite increasing the CRF to 24 in H.265-M for higher compression, the stable mAP indicates that object detection performance was not adversely affected.

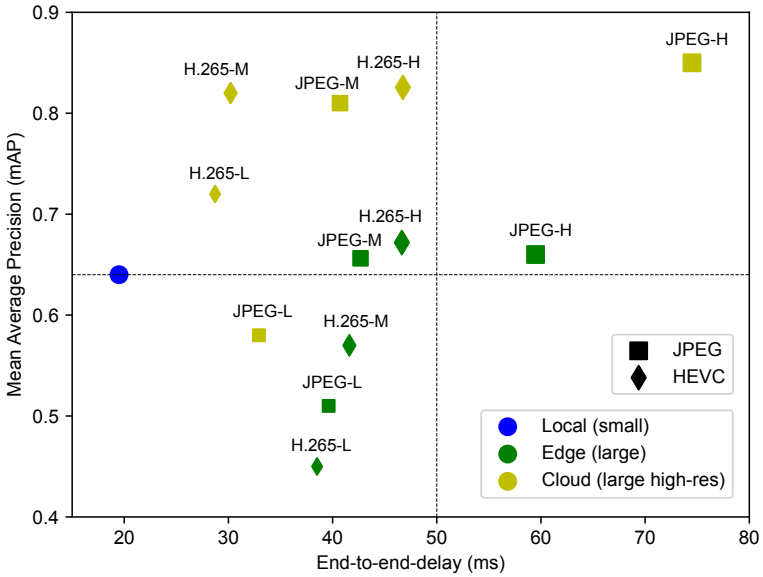


FIGURE 5.6: Trade-off between mean average precision (mAP) and end-to-end delay across various platforms and compression levels. The total delay accounts for the combined durations of compression, network transmission, decompression, and object detection inference. Extremely low-quality settings for H.265-VL and JPEG-VL are excluded from consideration, as their mAP values fall below 10%, which is insufficient for acceptable detection performance, even with reduced delays. **Source:** Reprinted from [57], with permission, ©2023 Elsevier.

The key insight is the substantial reduction in end-to-end delay to just 29.21 ms. The advantages observed are attributed to the core functionality of the Constant Rate Factor (CRF) in the H.265 codec, which skillfully balances video quality with file size. Even at elevated compression levels, the codec preserves critical features and details essential for accurate object detection. This decrease in file size enhances data transmission efficiency, substantially reducing the end-to-end delay.

In summary, the H.265-M compression emerges as a compelling choice for real-time object detection on cloud platforms. It maintains consistent detection accuracy while significantly improving responsiveness by lowering the delay. Additionally, further reducing the compression quality to H.265-L leads to a slight drop in mean average precision (mAP) to 72%, along with an additional reduction in delay to 28.72 milliseconds. Figure 5.6 illustrates the progression of mAP using JPEG compression techniques on the cloud platform. The mAP increases from 58% with JPEG-L to 81% with JPEG-M, reaching up to 85% with JPEG-H. At the same time, the end-to-end delay rises from 32.93 milliseconds (JPEG-L) to 40.71 milliseconds (JPEG-M), peaking at 74.50 milliseconds with JPEG-H.

Although JPEG-H provides the highest detection quality with an mAP of 85%, its increased latency of 74.5 milliseconds renders it unsuitable for real-time object detection at 20 Hz. However, it remains a viable option for applications where a 10 Hz latency is acceptable. Tasks like long-range route planning and trajectory optimization, which are less sensitive to latency constraints, can benefit from the enhanced detection quality without significantly affecting overall system performance. This analysis highlights the potential to strategically balance detection accuracy and latency by leveraging Edge and Cloud platforms in real-world applications.

(A) H.265-H (CRF=0); Detected: 1 pedestrian, 7 vehicles, 8 traffic light

(B) H.265-M (CRF=24); Detected: 1 pedestrian, 7 vehicles, 7 traffic light

(C) H.265-L (CRF=30); Detected: 0 pedestrian, 7 vehicles, 6 traffic light

(D) H.265-VL (CRF=51); Detected: 0 pedestrian, 0 vehicles, 0 traffic light

FIGURE 5.7: Visualisation of the number of detected pedestrians, vehicles and traffic light in the cloud platform on different H.265 compression settings. A detection is considered only if its IoU with a ground truth bounding box exceeds 50%. Ground truth: 8 traffic lights, 8 vehicles and 2 pedestrians. **Source:** Reprinted from [57], with permission, ©2023 Elsevier.

In the edge scenario, applying high-quality H.265 compression (H.265-H) results in a mean average precision (mAP) of 67% with an end-to-end delay of 48.65 milliseconds. When the compression quality is reduced to medium (H.265-M) and low (H.265-L), the mAP decreases to 57% and 45%, respectively, while the delay shortens to 41.61 milliseconds and 38.51 milliseconds.

This trend is particularly noteworthy when contrasted with the cloud scenario. However, it aligns with findings from earlier studies [148, 100, 50], which indicate that H.265 compression tends to perform better with high-resolution input images. For JPEG compression on the edge platform, the mAP improves from 51% using low quality (JPEG-L) to 66% with medium quality (JPEG-M), and reaches 67% at high quality (JPEG-H). Correspondingly, the end-to-end delay increases from 39.62 milliseconds (JPEG-L) to 43.59 milliseconds (JPEG-M), peaking at 59.48 milliseconds (JPEG-H). These results suggest that the edge platform does not currently offer a significant advantage over cloud offloading. However, the performance and competitiveness of edge computing could potentially be enhanced by employing specialized hardware designed for efficient inference, rather than relying on conventional consumer GPUs.

In contrast, the cloud platform delivers notably better results, achieving higher detection performance with both JPEG and H.265 compression methods. Specifically, it attains an mAP of 81% using JPEG-M and 82% with H.265-H, all while maintaining end-to-end delays below 50 ms. This ensures compliance with the 20 Hz real-time processing requirement. While compression is essential for real time functionality on both edge and cloud platforms, it can adversely affect detection quality.

Compression	Platform	Pedestrian (+% of Local)	Vehicle (+% of Local)	Traffic light (+% of Local)
No compression	Local	0.30 (0%)	0.79 (0%)	0.80 (0%)
	Edge	0.36 (+20%)	0.81 (+2%)	0.82 (+2%)
	Cloud	0.81 (+170%)	0.89 (+12%)	0.86 (+7%)
JPEG-H	Edge	0.36 (+20%)	0.80 (+1%)	0.82 (+2%)
	Cloud	0.80 (+166%)	0.89 (+12%)	0.86 (+7%)
JPEG-M	Edge	**0.41 (+36%)**	**0.83 (+5%)**	**0.78 (-2%)**
	Cloud	**0.65 (+116%)**	**0.88 (+11%)**	**0.85 (+6%)**
JPEG-L	Edge	0.35 (+16%)	0.77 (-2%)	0.74 (-7%)
	Cloud	0.43 (+43%)	0.84 (+6%)	0.81 (+1%)
JPEG-VL	Edge	0.23 (-23%)	0.73 (-7%)	0.55 (-31%)
	Cloud	0.24 (-20.00%)	0.78 (-1%)	0.62 (-22%)
H.265-H	Edge	**0.36 (+20%)**	**0.80 (+1%)**	**0.83 (+3%)**
	Cloud	**0.68 (+126%)**	**0.83 (+5%)**	**0.83 (+3%)**
H.265-M	Edge	0.25 (-16%)	0.78 (-1%)	0.82 (+2%)
	Cloud	**0.69 (+130%)**	**0.84 (+6%)**	**0.82 (+2%)**
H.265-L	Edge	0.04 (-86%)	0.59 (-25%)	0.72 (-10%)
	Cloud	**0.51 (+70%)**	**0.82 (+3%)**	**0.82 (+2%)**
H.265-VL	Edge	0.01 (-96%)	0.02 (-97%)	0.03 (-96%)
	Cloud	0.03 (-90%)	0.23 (-70%)	0.04 (-95%)

TABLE 5.6: We present per-class Average Precision (AP) scores across various compression qualities and computational platforms. The input resolutions are set to 640×640 pixels for both local and edge platforms, and 1280×1280 pixels for the cloud platform. The baseline performance of the local model is shaded in gray for reference.
Source: Reprinted from [57], with permission, ©2023 Elsevier.

Excessive compression levels may cause the mAP for remote detection to drop below that of local processing, even as they introduce longer delays making the offloading process counterproductive. For instance, when using very low-quality settings like H.265-VL and JPEG-VL in both cloud and edge scenarios, the mAP dropped below 10%. Such a low detection rate is unacceptable despite the reduced end-to-end latency, as it compromises the overall effectiveness of the object detection system. We conducted a comprehensive evaluation of detection quality across three primary object classes: pedestrians, vehicles, and traffic lights. This assessment is visually depicted in Figure 5.7, illustrating how each compression method uniquely affects these classes. The results, summarized in Table 5.6, indicate that compression unevenly affects Average Precision (AP), with smaller classes being more affected.

To benchmark performance, we compared various compression settings on different platforms against a baseline scenario the local platform operating without any compression. In this baseline, the local platform achieved an AP of 30% for pedestrian detection and higher APs of 79% and 80% for vehicles and traffic lights, respectively. As shown in Table 5.6, switching to the cloud platform resulted in substantial improvements in pedestrian detection performance compared to local and edge platforms. Specifically, with medium compression settings like JPEG-M and H.265-M, there were increases in AP of 116% and 130%, respectively. This demonstrates that the cloud platform can significantly enhance detection quality for certain classes when appropriate compression are applied.

5.6 Conclusion and Future Work

In this study, we investigated the feasibility of performing real-time object detection remotely. While larger neural network models typically offer better detection performance, they also require substantial computational power. Due to cost and energy constraints inherent in autonomous vehicles, running the most advanced models locally and in real time is often impractical. To overcome this limitation, we proposed several strategies to offload object detection tasks to edge or cloud devices using Cellular Vehicle-to-Everything (C-V2X) communication. We evaluated these offloading strategies by comparing their detection accuracy and their ability to meet end-to-end latency requirements. To facilitate this assessment, we generated a synthetic dataset and trained various versions of the YOLOv5 architecture. Utilizing a comprehensive end-to-end 5G network simulation framework, we measured the network latency associated with transmitting camera frames for processing on edge and cloud platforms. Additionally, we analyzed how employing heavy compression techniques specifically JPEG and H.265 codecs can reduce frame sizes by up to 98%, thereby enabling real-time remote processing.

Our findings demonstrate that with appropriate compression, models can run in real time on edge or cloud platforms while surpassing the performance of local detection. The experimental results indicate that H.265 video compression provides a superior balance between detection quality and end-to-end latency compared to JPEG image compression, especially in cloud-based scenarios. Nonetheless, JPEG compression remains sufficient in certain contexts, such as applications where a 10 Hz latency is acceptable. Importantly, our experiments revealed that excessive compression can degrade detection quality relative to processing raw frames, particularly for detecting pedestrians. This highlights the necessity of carefully balancing compression levels to maintain high detection accuracy while satisfying real-time latency constraints in autonomous driving applications.

Chapter 6

Lightweight Feature Sharing for Real-Time Perception

As discussed in the previous chapter, most object detection models face a trade-off between high accuracy and fast inference speeds, creating challenges in achieving real-time performance. While JPEG and H.265 compression can aid in streaming sensor data for edge or cloud-based processing, another promising approach is network layer partitioning. This approach enables computationally intensive layers to be processed in the cloud while retaining lighter layers on the vehicle, effectively balancing workload distribution and potentially enhancing overall performance [22].

This chapter presents the development and evaluation of a model partitioning strategy designed for cooperative perception in autonomous driving. Our approach addresses the inherent trade-off between high detection accuracy and low latency, both of which are crucial for real-time performance. By partitioning the model such that computationally heavy layers are processed in the cloud while lighter layers remain on the vehicle, we aim to optimize data transmission and processing efficiency.

This chapter builds upon the following publications:

Faisal Hawlader, François Robinet, and Raphaël Frank. "Lightweight Features Sharing for Real-Time Object Detection in Cooperative Driving". In: *2023 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2023. DOI: 10.1109/VNC57357.2023.10136339

The chapter is organized as follows: Section 6.1 introduces the motivation for this chapter, concluding with a summary of contributions. Section 6.2 reviews related work on lightweight feature vector sharing for real-time object detection. Section 6.3 outlines the methodology and technical details of our approach. Section 6.4 presents and discusses our findings, demonstrating the effectiveness of feature sharing between vehicles and the cloud, along with an evaluation of bandwidth requirements. Finally, Section 6.5 summarizes our findings and outlines directions for future work.

6.1 Introduction

Real-time object detection is essential for enhancing the safety and operational efficiency of autonomous vehicles. However, the computational demands of advanced object detection algorithms often exceed the processing capacity of the limited hardware available in many vehicles. Most object detectors must balance a trade-off between accuracy and speed, making it difficult to achieve high levels of real-time performance without significant sacrifices in either domain. A promising solution to this issue is model partitioning, where parts of the network layers are processed on the vehicle while more computationally demanding layers are offloaded to the cloud for processing [22].

The core concept of model partitioning involves assigning computationally heavy layers to the cloud while retaining lightweight layers on the vehicle. This approach aims to maintain high detection accuracy by distributing computational tasks according to the processing capabilities of each platform. However, this partitioning comes with an inherent challenge: the latency introduced by transmitting intermediate data from the vehicle to the cloud. Such latency can pose a significant problem for real-time applications, as strict timing constraints are often necessary to ensure safe and responsive object detection.

To address these latency issues, previous studies have explored post-training quantization and compression of layer activations as practical solutions [22]. Quantization and compression techniques reduce the memory footprint and network bandwidth requirements, which, in turn, lowers transmission delays and computational costs. After applying quantization, additional data compression often using lossless encoding methods can further minimize the size of the feature tensors being transmitted, optimizing the data flow between vehicle and cloud.

This work specifically investigates the latency implications of lightweight feature sharing over a cellular network. Our study examines critical factors such as the optimal choice of partitioning layers, the achievable accuracy levels, and the overall impact on latency, offering insights into how these aspects influence real-time performance.

This chapter contributes to the research with the following:

Contributions

- Designed a model partitioning strategy that optimizes data transmission by balancing delay and accuracy through early network splitting.
- Analyzed the effects of quantization and clipping on detection accuracy and latency, providing guidelines to balance mean Average Precision (mAP) and end-to-end delay for cooperative perception applications.
- Evaluated lightweight feature sharing over a simulated 5G network, emphasizing trade-offs between compression levels and transmission latency.

6.2 Related Work

Real-time object detection for autonomous vehicles demands efficient handling of computational resources and network bandwidth [57]. Previous studies have explored model partitioning as a method to distribute processing tasks between a vehicle onboard systems and remote cloud infrastructure [22, 104, 89]. This approach can alleviate the processing load on vehicles by offloading computationally intensive layers to the cloud, while lighter layers are retained locally [50]. However, model partitioning often introduces latency due to data transmission requirements, which is especially challenging for applications requiring low-latency responses [51].

To mitigate these delays, prior research has focused on quantization and compression techniques to reduce the size of transmitted data while preserving detection accuracy [89, 124]. Quantization, by lowering the bit precision of model activations, minimizes the data that needs to be shared, thus reducing transmission time and bandwidth demands [44]. Compression methods, such as zlib or other lossless algorithms, have been applied in tandem with quantization to further decrease data size without sacrificing critical information [137]. Recent work has examined the balance between quantization levels and detection performance, emphasizing the importance of maintaining mean Average Precision (mAP) while achieving lower transmission delays [16].

Cooperative perception frameworks for autonomous vehicles leverage V2X (vehicle-to-everything) communication to expand sensory coverage by sharing data across multiple vehicles. Through V2X, vehicles gain a broader view of their surroundings, which improves object detection and tracking capabilities in dynamic environments [94]. Despite progress in this area, open-source simulation tools that integrate realistic perception data with robust communication models remain limited. Extensions to simulators like CARLA provide a testbed for cooperative perception, allowing for complex, photorealistic experiments that would be impractical to conduct on real roads [170]. Simulating network conditions is also critical to evaluating latency and bandwidth effects on cooperative perception. Simu5G [114], based on OMNeT++ [151], has been widely used to model 5G networks [39], assessing the end-to-end delays associated with packet transmission under various network constraints [51, 106]. By providing insights into the latency characteristics of 5G networks, prior work with Simu5G has helped assess the feasibility of offloading perception tasks to the cloud, particularly under the stringent latency requirements typical of autonomous driving applications [133, 105].

While these studies have laid essential groundwork in model partitioning, quantization, cooperative and distributed perception, and network simulation, the approach in this chapter integrates these elements into a cohesive framework designed for real-time object detection in cloud-augmented autonomous driving. By investigating optimal layer partitioning, quantization settings, and compression techniques within a 5G network simulation, this work addresses the balance between detection accuracy, latency, and computational efficiency. Additionally, the CARLA extension developed here offers a flexible simulation environment for sensor data sharing among vehicles, supporting further advances in efficient data transmission strategies for scalable autonomous driving systems.

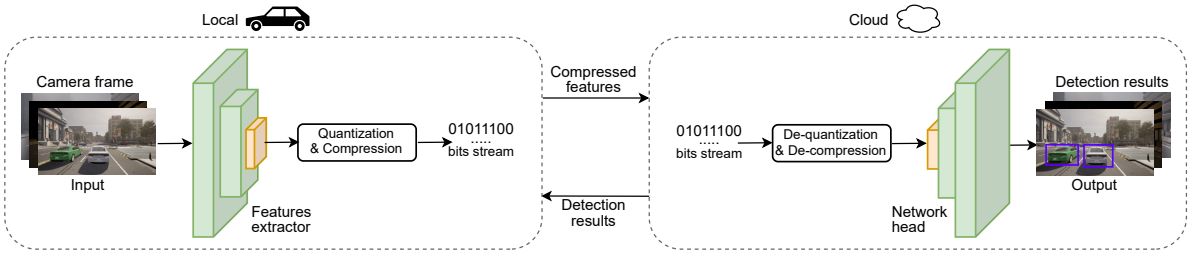


FIGURE 6.1: System overview of lightweight feature sharing for real-time object detection in cooperative driving. **Source:** Reprinted from [58] with permission, ©2023 IEEE.

6.3 Methodology

This section details the methodology used to implement and evaluate a model partitioning strategy for real-time object detection in autonomous driving. The proposed approach combines model partitioning, quantization, and compression techniques to manage the computational and bandwidth limitations of vehicle-based systems while leveraging cloud resources for complex processing tasks. We begin by defining a model architecture in which lightweight feature extraction is performed locally on the vehicle, followed by data transmission to the cloud. To optimize data transfer and reduce latency, we apply post-training quantization and lossless compression to the feature maps before transmission. A simulated 5G network environment is used to assess the impact of different quantization and compression settings on latency, providing insights into how these configurations can support cooperative perception in cloud supported perception systems. The methodology also outlines the experimental setup, hardware configurations, and dataset design used to validate the proposed framework, as shown in Figure 6.1.

6.3.1 Lightweight Feature Sharing

Building on the experimental results and findings on different YOLO variants from the previous chapter, this chapter focuses specifically on the larger variant for further investigation. This model demonstrates strong potential for real-time perception in distributed computing environments utilizing cloud resources. Configured with an input resolution of 640×640 pixels and consisting of 43 million learnable parameters [75], the larger YOLO model is structured for efficient feature sharing. The initial layers, which handle lower-level feature extraction, are executed locally on the vehicle to process raw sensor input, generating intermediate activation maps that encode spatial and structural information essential for object detection. As illustrated in Figure 6.1, our method extracts activation maps from the local vehicle processing unit, specifically from the initial model layers that function as a feature extractor. These early layers capture key features from the input, generating intermediate activation maps that are subsequently transferred to the cloud for further processing. Due to the substantial size of these activation maps, direct transmission demands high bandwidth, as discussed in Chapter 3. To mitigate this, we applied quantization and compression to reduce the size of the data prior to transmission.

Upon receiving the compressed feature maps, the cloud performs decompression and dequantization. The remaining model layers, dedicated to object detection, are then executed in the cloud, completing the inference process. This division of tasks between local and cloud processing balances computational demands by maximizing the vehicle limited resources and leveraging cloud processing power for more intensive computations.

For this initial setup, we apply straightforward quantization and compression techniques. The feature values are clipped to a specified range, $[0, c_{max}]$, which reduces the variance of the feature map values and simplifies quantization. The clipped feature values are then uniformly quantized to L discrete levels using Equation 6.1:

$$Q = \text{round}\left(\frac{\text{clip}(x)}{c_{max}}(L-1)\right) \tag{6.1}$$

In this equation, Q represents the quantized value, $\text{clip}(x)$ confines the feature values to the defined range, and c_{max} and L are parameters that define the upper range and the number of quantization levels, respectively. By reducing the numerical precision, we lower the data size, thereby decreasing transmission requirements. After quantization, we apply zlib, a lossless compression technique, to further reduce the size of the feature maps before transmission [44]. This additional compression step helps achieve significant bandwidth savings without sacrificing data loss, as zlib retains the original information in the compressed data.

To estimate the end-to-end latency of this distributed object detection process, we consider the cumulative time required for each stage. This includes the time for local feature extraction, quantization, compression, data transmission over the network, and cloud-based inference using the remaining model layers. This approach allows us to comprehensively analyze how each component affects latency and overall detection performance, particularly in scenarios that demand real-time responses.

6.3.2 Network Simulation

To simulate a 5G core network and assess end-to-end transmission latency, we employ the Simu5G framework [114], built on OMNeT++ [151]. Simu5G enables us to emulate realistic network conditions that autonomous vehicles may face when transmitting data to cloud servers. In this setup, we focus on evaluating the latency of data packet transmission, specifically for sending feature maps for remote processing.

All major simulation parameters for end-to-end network transmission latency remain consistent with those detailed in Chapter 4. Our findings indicate that transmitting a single 4096-byte UDP packet results in an average latency of 0.45 ms. Since raw detection results can be encapsulated within a single packet, this adds 0.45 ms to the overall latency each time processed detection results are sent back to the vehicle. For further details on the parameters and configurations used in this network simulation, refer to [59, 57].

Hardware: To assess the feasibility of the system under real-world hardware constraints, local computations are executed on an NVIDIA Jetson NX SoC, which has a thermal design power (TDP) of approximately 20W, simulating the limited processing capacity typical in vehicle based systems. Cloud side processing utilizes a high performance setup with an NVIDIA Tesla V100 GPU and an Intel Xeon G6132 CPU, together reaching a TDP of around 450W. This setup supports handling computationally intensive tasks, enabling efficient processing of data offloaded from the vehicle.

Dataset: To train the object detection model, we utilized the synthetic dataset described in the previous chapter and released as open-source with the publications [57, 59]. This dataset is specifically tailored to meet the study requirements and is divided into three subsets: 6000 frames for training, 2000 frames for validation, and 2000 frames for testing. Each subset is curated to cover a range of scenarios likely encountered in real-world environments. Further details on the dataset generation process and specific training protocols can be found in [57] and in Chapter 3.

6.4 Results

Given the structure of the YOLOv5 architecture and the limitations of local hardware, splitting the network in the early layers is most effective [106]. Splitting at later layers, particularly after the introduction of skip connections [108], would require transferring feature maps from multiple layers, increasing data transfer complexity and latency. However, strided convolutions in the deeper layers progressively reduce the spatial dimensions of feature maps, minimizing data transfer size, a result that aligns with our preliminary analysis and a previous study [22].

Balancing these factors, we selected layer 4 as the optimal split point. This choice allows for manageable data transfer, avoids complexities from skip connections, and benefits from the spatial reduction provided by early strided convolutions.

6.4.1 Optimal split-layer and clipping

We begin by analyzing the effects of varying c_{max} and the number of quantization levels on detection quality. As expected, the choice of quantization levels influences the optimal value of c_{max} due to its role in controlling the range and distribution of feature values, which in turn affects how information is preserved during quantization.

As shown in Figure 6.2, for moderate quantization levels, specifically $L = 8$ and $L = 16$, detection quality remains fairly robust across a range of c_{max} values from 5 to 15. This robustness suggests that with these quantization levels, the model can effectively maintain the critical features necessary for accurate object detection within this range. This stability is beneficial for practical implementations, as it allows some flexibility in choosing c_{max} without significantly sacrificing mAP.

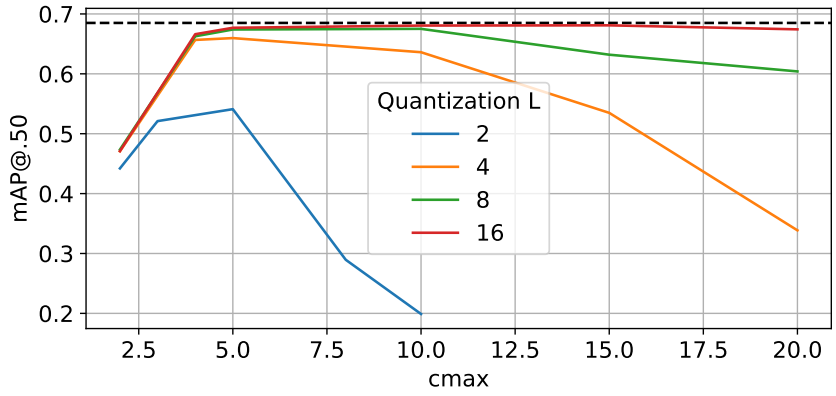


FIGURE 6.2: Effect of varying c_{max} values and quantization levels on detection quality. The dashed line indicates the mAP of the model without quantization. **Source:** Reprinted from [58] with permission, ©2023 IEEE.

However, for more extreme quantization (e.g., lower L values), detection quality becomes increasingly sensitive to the choice of c_{max}. In these cases, a smaller c_{max}, specifically $c_{max} = 5$, yields the best results. This outcome likely reflects the need to focus the quantization range more tightly around smaller values, ensuring that essential features are not excessively distorted or clipped. In extreme quantization, an inappropriately high c_{max} risks introducing greater quantization noise and losing subtle but important variations in feature maps, thereby degrading mAP. These findings highlight the importance of selecting appropriate c_{max} values in conjunction with quantization levels to maintain detection performance. The observed sensitivity at extreme quantization levels underscores the need for careful parameter tuning in applications that require high compression but cannot compromise detection accuracy. This trade-off between quantization aggressiveness and detection quality is a crucial consideration for deploying object detection models in resource-constrained environments, such as edge devices.

6.4.2 Impact of quantization and compression levels

Figure 6.3 provides insights into the trade-off between detection quality, measured by mAP, and end-to-end latency as we vary the quantization and compression levels. Based on the findings from the previous section, we select $c_{max} = 5$ for configurations with $L \leq 4$, as this setting minimized mAP degradation in lower quantization scenarios, while $c_{max} = 10$ is chosen for configurations with higher quantization levels to preserve detection quality more effectively. As quantization levels decrease, the total amount of data to be transmitted between the vehicle and the cloud is significantly reduced, resulting in lower latency. This effect is particularly noticeable with extreme quantization (e.g., level-2 and level-4), where we observe substantial latency improvements. However, these lower quantization levels come with a clear trade-off, as the mAP declines notably under level-2 and level-4 quantization. This result indicates that while aggressive quantization may reduce transmission time, it compromises detection accuracy by constraining the range and granularity of feature representations.

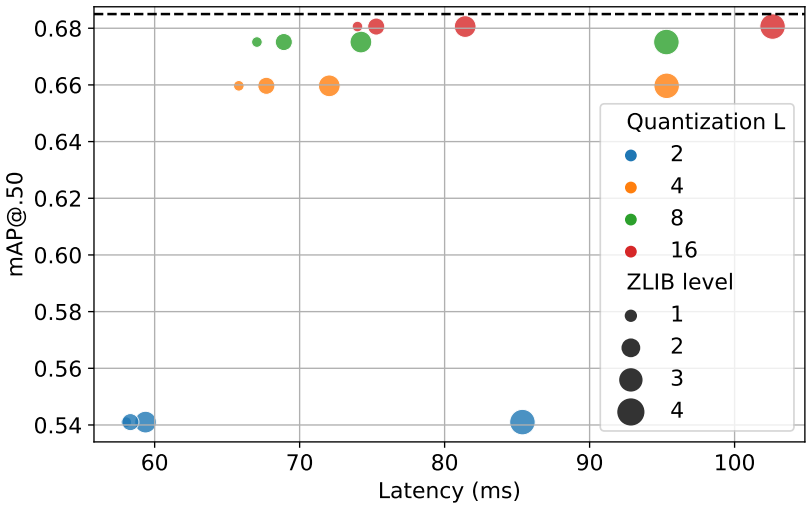


FIGURE 6.3: Trade-off between mean average precision (mAP) and end-to-end latency for different quantization levels and compression factors. The mAP of the unquantized model is represented by the dashed line. **Source:** Reprinted from [58] with permission, ©2023 IEEE.

In practical terms, this trade-off implies that achieving very low latency may come at the cost of reduced object detection precision, which could impact real-time decision making in autonomous driving applications. The influence of zlib compression on latency is also evident in Figure 6.3. As zlib compression is lossless, it does not impact mAP directly. However, as we increase the compression factor, the time required for compression on the local hardware rises. This additional compression time directly contributes to end-to-end latency, particularly at higher compression factors where the compression becomes more computationally intensive. The effect is cumulative with lower quantization levels, where the increased compression factor further extends latency.

For moderate quantization levels (e.g., level-8 or level-16), the trade-off is more balanced. Here, the mAP remains relatively stable while latency is reduced due to more compact data representation. These quantization settings appear to provide an optimal balance, preserving high detection accuracy while offering latency reductions that are manageable within the constraints of real-time processing.

In summary, the results in Figures 6.2 and 6.3 underscore the need for careful tuning of both quantization and compression factors. While low quantization and high compression factors can provide latency benefits, these come at the cost of either mAP degradation or increased processing times on local hardware. Selecting an optimal configuration thus requires balancing these parameters to achieve low latency without substantially sacrificing detection quality, especially for applications with strict real-time requirements. This analysis highlights potential strategies for effectively balancing data transmission needs and computational overhead, which are critical considerations in designing efficient, high-performance cooperative perception systems for autonomous vehicles.

6.5 Conclusion and Future work

In this chapter, we demonstrate the feasibility of sharing lightweight features for real-time object detection in cooperative driving environments. Our study examined the balance between detection quality and latency, with an emphasis on determining effective quantization and compression levels for data sharing and distributed processing. The results indicate that detection accuracy is particularly sensitive to the selection of c_{max}. If c_{max} is not optimized, aggressive quantization can significantly degrade detection accuracy, which impacts the reliability of real-time detection.

Additionally, while compression helps reduce transmission delays, excessive compression on local hardware can be computationally demanding. This added load leads to increased end-to-end latency. Therefore, a balanced approach is essential, where both quantization and compression levels are carefully tuned to minimize latency without sacrificing detection quality.

Part IV

Field Trials and Real-World Deployment

Chapter 7

Experimental Validation of Cooperative Perception

This chapter introduces a validation framework and a field trial for real-time perception, utilizing both local and cloud computing resources. Previous chapters examined the offloading of camera sensor data processing to a cloud server, which demonstrated significant promise, particularly in minimizing end-to-end latency and achieving high perception accuracy. However, these findings were mainly based on results from vehicular simulation. Although simulations provide controlled environments that are valuable for initial testing, they often lack the complexity of real-world conditions. Real-world environments are inherently dynamic and present scenarios that are challenging to replicate in simulations, especially in the context of CAV systems and cooperative driving.

This chapter presents a real-world field trial validating our offloading approach under diverse, uncontrolled conditions. By combining onboard processing with cloud offloading, we evaluate performance across key metrics: end-to-end delay, detection accuracy, and resilience to environmental variability, with a focus on network measurements. The trial also benchmarks H.265 encoding for data compression to optimize V2X transmission, balancing bandwidth usage with perception quality.

This chapter builds upon the following publication:

> Faisal Hawlader, François Robinet, and Raphaël Frank. "Cooperative Perception Using V2X Communications: An Experimental Study". In: *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*. IEEE. 2024. DOI: 10.1109/VTC2024-Fall63153.2024.10757448

The remainder of this paper is structured as follows. In Section 7.2, we review the related literature. Section 7.3 describes experimental setup, including details on the hardware and software configuration, the vehicle instrumentation, and data collection during field trials. Section 7.4 presents the experimental results and discusses data processing trade-offs in terms of end-to-end delay and detection quality. Finally, in Section 7.5, we conclude this work with a summary of our findings.

7.1 Introduction

Simulations are valuable for initial testing of perception frameworks [55, 53], providing controlled conditions for preliminary evaluations [61]. They lack the dynamic complexity of real world environments [54], which is essential to validate performance under genuine and unpredictable conditions [8]. Previous studies indicate that cloud offloading of perception tasks can reduce end-to-end delay and improve accuracy [51, 57]. However, these results are often based on simplified simulations [53]. For example, network latency in simulations may remain stable, whereas real-world latency varies widely due to factors such as location, interference, and congestion, all of which can significantly affect end-to-end delay [165]. Real-world validation, therefore, is crucial to understanding how perception frameworks perform outside controlled environments [9].

This chapter addresses this gap by presenting a field trial of a hybrid perception framework that combines on-board and cloud processing. The trial evaluates end-to-end latency, detection accuracy, including network fluctuations [118, 117], and environmental variability. Furthermore, we benchmark the H.265 encoding to assess its effectiveness in balancing bandwidth use and perception quality in practical scenarios [158, 126].

Insights from this field trial deepen our understanding of perception system limitations. They reveal opportunities to optimize these systems under real-world conditions. This knowledge is essential to advance autonomous driving systems beyond the capabilities of simulation-based studies alone.

In this chapter, we make the following key contributions, including the release of a real-world driving dataset[1]:

Contributions

- We present a validation framework for assessing real-time perception in connected driving environments. This framework employs field trials and rigorous validation methods to address challenges in sensor data processing, including latency management and communication constraints.
- We introduce a real-world driving dataset captured with a front-facing rooftop camera, annotated with pedestrian, vehicle, and traffic light classes. This dataset supports the training and evaluation of object detection models within a perception pipeline tailored for cooperative driving scenarios.
- We evaluate both onboard and cloud processing approaches to enhance real-time object detection and minimize latency under real-world conditions. For cloud offloading, we apply H.265 video encoding, and examine the trade-off between end-to-end delay and detection accuracy to balance bandwidth requirements with perception quality.

[1] **Dataset available:** https://zenodo.org/records/14523854

7.2 Related Work

This section reviews the existing literature on real-time perception for connected and automated driving, focusing on key areas directly relevant to our study [56].

Prior research has established the pivotal role of V2X communication in extending perception range and advancing connected driving systems [34, 140, 55]. Recent studies have examined various facets of V2X and CPM, focusing on communication protocols and message formats [91, 132]. Protocols such as ITS-G5 [140, 138] and C-V2X [48, 107] are frequently investigated for their ability to support low-latency, reliable data channels between vehicles [114, 130]. Additionally, research efforts have aimed at optimizing CPM message formats and establishing generation rules [41] to facilitate relevant perception data exchange while minimizing communication overhead [30, 143].

Despite these advancements [142, 96], onboard perception systems continue to face substantial challenges [121]. Vehicles equipped with limited computing resources often struggle to meet the high computational demands required for real-time perception and decision-making [49, 51]. This issue becomes particularly pronounced in dynamic environments [66], where processing delays can compromise both perception accuracy and the system ability to react quickly [21]. Consequently, there exists a critical trade-off between accuracy and processing speed, where optimizing one frequently impacts the other [11, 111]. This trade-off has been widely documented in recent surveys [165], underscoring the need for more efficient, adaptive perception frameworks to enhance system performance without overburdening onboard hardware.

Integrating cloud technology with C-V2X has emerged as a promising approach [34], as it enables vehicles to transmit sensor data to remote cloud servers with ample computational resources [93, 78]. This setup allows for more scalable and computationally intensive processing, fostering the development of robust driving ecosystems [51, 38]. Studies have shown that cloud-based C-V2X processing enhances scalability and supports more complex perception tasks [93, 2]. However, challenges such as data transmission latency and limited bandwidth remain significant barriers to fully realizing this potential [4, 141].

Addressing these limitations requires sustained research and investment to optimize the integration of cloud and connected vehicle technologies [90, 25]. Testing and validating such integrated systems in real time is essential for ensuring they meet latency and other application-specific requirements [40, 169]. Nonetheless, implementing these systems in vehicles introduces further complexities. Equipping vehicles with necessary sensors, communication hardware, and software that seamlessly integrates with cloud servers can be costly and may face constraints from regulatory and safety considerations. These challenges often limit the scope of real-world tests, leading many researchers to rely on simulation-based evaluations, which lack direct real-world applicability [135, 23].

7.3 Methodology

In this section, we detail the methodology used to design and setup the testbed framework aimed at evaluating distributed vehicular perception.

7.3.1 Tested Scenarios

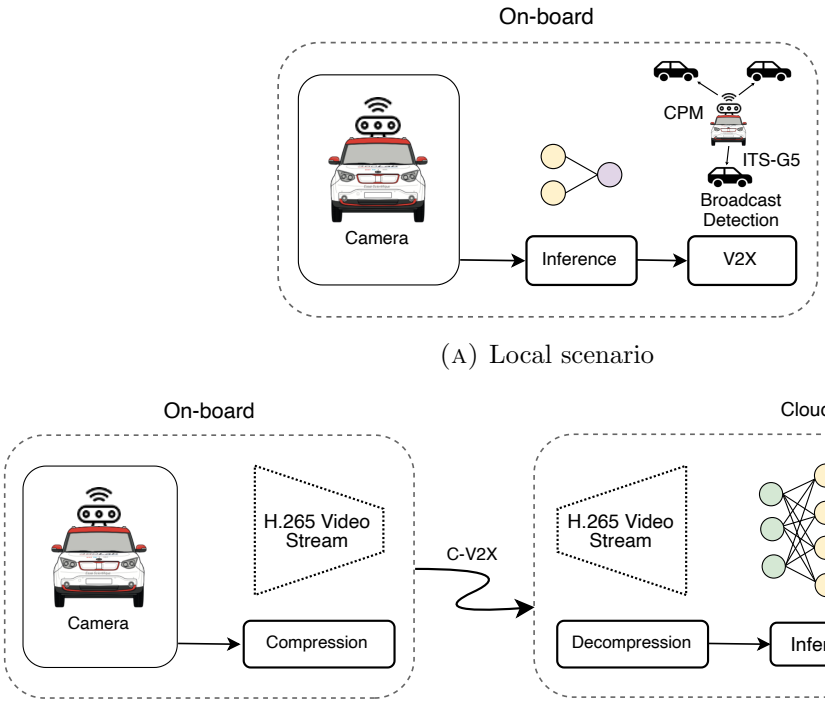
We evaluate the performance of cooperative object detection through two distinct scenarios, depicted on Fig. 7.1.

- **Local scenario:** Camera images are processed on-board upon capture. To cope with on-board hardware constraint, a lightweight detector is used (see Section 7.3.6 for details). The detection results are then encoded to generate CPM, which are broadcast to nearby vehicles or roadside units (RSU) using the ITS-5G communication standardized by the ETSI [146].
- **Cloud scenario:** Camera images are compressed on-board upon capture using the H.265 codec. The compressed bitstream is then transmitted to the cloud using the cellular mode of C-V2X. Once on the cloud, the encoded frames undergo decoding, followed by an inference step to detect objects. Since hardware constraints are relaxed on the cloud compared to on-board, a more computationally demanding model is used than in the Local scenario. Finally, the detection results are transmitted back to the originating vehicle as well as to nearby vehicles using C-V2X.

The Local scenario focuses on on-board processing within the vehicle, while the Cloud scenario explores the potential benefits of cloud-based processing. We aim to evaluate and compare the performance of local and cloud processing approaches in terms of end-to-end delay, detection quality, and overall effectiveness in real-world driving scenarios.

7.3.2 Vehicle Instrumentation

We conduct experiments using an experimental vehicle used for CAVs research. The vehicle, a 2018 KIA Soul EV, is equipped with a suite of sensors, including a LiDAR, cameras, and a GNSS receiver. These sensors are crucial for capturing environmental data required for perception tasks. However, for the purposes of this study, we focus exclusively on processing data from a single front-facing camera sensor. Communications between vehicles and with infrastructure are facilitated by the YoGoKo Y-Box communication module, which features ITS-G5 and C-V2X communication technologies. For further information on the test vehicle, equipped sensors, hardware, and driving software stack, we refer the reader to [155].



(A) Local scenario

(B) Cloud scenario

FIGURE 7.1: Experimental scenarios: in the Local scenario, inference is performed on-board using a lightweight detection model, and the results are transmitted as CPMs to nearby vehicles and/or infrastructures over ITS-G5. In the Cloud scenario, a compressed video feed is sent to a Cloud server using C-V2X, where a more computationally demanding detector is used. The detection results are then transmitted to vehicles in the vicinity, enabling cooperative perception.
Source: Reprinted from [56] with permission, ©2024 IEEE.

7.3.3 Test Route & Network Measurements

Our driving tests were conducted on public roads in the Kirchberg area of Luxembourg City, illustrated in Figure 7.2. This area offers diverse road features and traffic conditions, allowing to evaluate cooperative perception scenarios with a high degree of realism.

In the Local scenario, the experimental setup for the CPM measurements involved conducting V2X communication using the ITS-G5 protocol within a dynamic environment. The experimental setup involved a stationary receiver located at specific coordinates (lon 6.161993, lat 49.626478) while a transmitter vehicle moved at speeds ranging from 40 to 50 km/h. This stationary receiver position was chosen to ensure consistency in measurement conditions, providing a fixed reference point for assessing the quality of communication between the transmitter vehicle and the receiver. The experimental route spans over a distance of approximately 1.5km^2. Within this experimental setup, we evaluated the transmission of CPM messages to assess the reliability and performance of V2X communication under public traffic conditions.

[2]**Local test route:**http://g-o.lu/3/GsHC

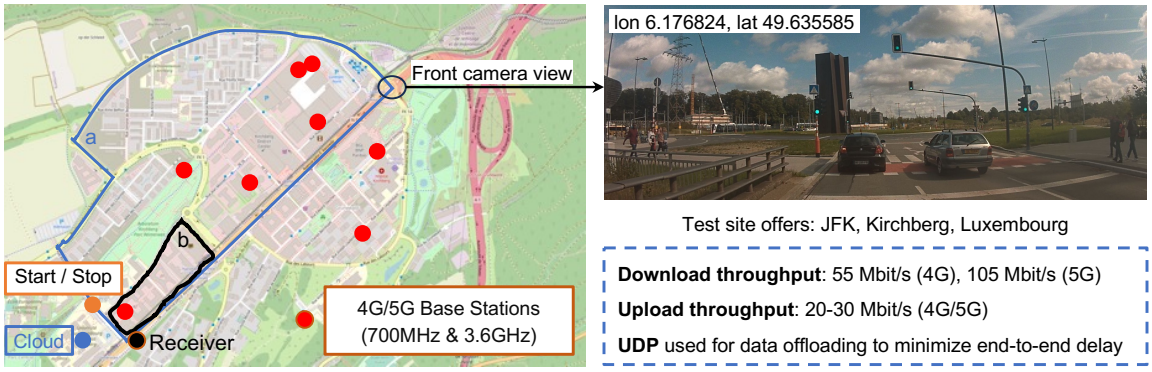


FIGURE 7.2: Test site in Luxembourg City, illustrating area and route for local (black) and cloud (blue) processing scenarios.

In the Cloud scenario, the driving route spans approximately 4km[3] and we use the cellular mode of the C-V2X technology to communicate with a cloud server located at the premises of the University of Luxembourg in the same area. Twelve commercial base stations are located in proximity of the test route, including 4G and 5G (non-standalone) cellular sites, using Low- (700 MHz) and Mid-band (3.6 GHz) frequencies. The area features an average download throughput of 55 Mbit/s for 4G and 105 Mbit/s for 5G, whereas the average upload throughput oscillates between 20 and 30 Mbit/s for both technologies. We use UDP for data offloading to the cloud, as UDP has shown potential for streaming sensor data to a remote cloud for processing with the lowest end-to-end delay [57].

7.3.4 Data Collection & Annotation

Data collection is a crucial aspect in the development and evaluation of perception systems for autonomous vehicles. To obtain more accurate detection results on the previously introduced test routes, we first collect and annotate a dataset, and then perform transfer learning on the pre-trained YOLO model. To do so, we set up a front-facing camera and a GNSS sensor mounted in the front of the roof rack of the vehicle. We collected 5000 frames at 1Hz with synchronized time and GNSS measurements along the Cloud test route described in Section 7.3.3. To generate ground truth data for training and evaluation purposes, we manually annotated the collected frames and divided them into training (70%), validation (15%), and test (15%) subsets. The annotation process involved meticulously labeling objects of interest in the captured images, focusing on three object classes: pedestrians, vehicles, and traffic lights. We use the specialized Roboflow [76] annotation tool to streamline this process and ensure accurate labeling. The dataset comprises a total of 59574 annotated instances across all classes, with an average of 15 objects per image.

[3]**Cloud test route:**http://g-o.lu/3/96TS

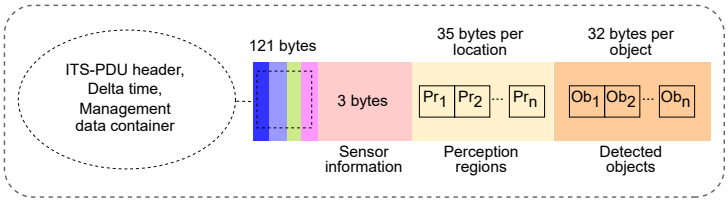


FIGURE 7.3: Overview of the different containers and their roles in the CPM format, as specified by the ETSI standard [146]. **Source:** Reprinted from [56] with permission, ©2024 IEEE.

7.3.5 CPM Encoding

In the Local scenario, the results of the object detection performed on-board are packaged into CPMs and broadcast to nearby vehicles over the ITS-G5 communication protocol. The list of detected objects is encoded into CPM following the ETSI standard [146]. A basic overview of different containers included in the CPM message format is displayed on Fig. 7.3. The ITS-PDU header, management data, and sensor information containers include, among others, information such as protocol versions, message type, reference position, and sensor ID. Containers of detected objects include information about the detected objects, such as the class of the object and the confidence level assigned by the perception system. According to the ETSI definition, the CPM should also include perception regions, which shall list the location information of the detected objects calibrated with the sensor location. Obtaining the location of the object would require an additional localization technique to be employed for each detected object, which is beyond the scope of this work. In order to maintain the standard format of the CPM payload, the ego vehicle location is used for the perception region.

7.3.6 Hardware Configuration & Detection Models

The limited processing capabilities and energy constraints inherent to vehicles pose significant challenges to real time applications such as cooperative perception. As a result, computational tasks are often offloaded to external resources, where computing resources are more abundant.

Platform	**Hardware configuration**
Local (≈100W)	GeForce GTX 1650 GPU 896 CUDA Cores, 5.7 TFLOPS (FP16) Intel i9-9980HK @2.4GHz
Cloud (≈450W)	NVIDIA GeForce RTX 4090 GPU 16384 CUDA Cores, 82.6 TFLOPS (FP16) Intel i9-13900K @2.8GHz

TABLE 7.1: Hardware setups for the Local and Cloud scenarios described in Section 7.3.1. **Source:** Reprinted from [56], with permission, ©2024 IEEE.

YOLOv8 variant	**Params(M)**	**FLOPS(B)**	**Inference(ms)** Local	Cloud	**mAP**
nano	3.2	8.7	19.2	1.2	0.65
small	11.2	28.6	**29.4**	1.9	0.69
medium	25.9	78.9	192.3	2.7	0.74
large	43.7	165.2	361.1	3.2	0.85
xlarge	68.3	257.8	521.4	**3.9**	0.89

TABLE 7.2: Comparison of YOLOv8 variants in terms inference time, and mAP on the test dataset presented in Section 7.3.4. **Source:** Reprinted from [56], with permission, ©2024 IEEE.

However, using this paradigm, achieving high detection quality and low end-to-end delay requires a well-designed architecture, high speed networks and high-performance computing [174]. To gain deeper insight into this issue and devise effective solutions, we use a real vehicle [155] equipped with sensors and other necessary software and hardware components in our setup. The hardware configurations used in our experiments are detailed in Table 7.1. The Local setup refers to the on-board device, for which we select hardware with appropriately limited power consumption. In contrast, Cloud offers more robust compute capabilities, at higher financial and electrical power costs.

To perform object detection, we use YOLOv8 [74], for its state-of-the-art performance in terms of accuracy and speed [75]. YOLOv8 offers various models ranging from small to extra-large. Table 7.2 presents a comparison of YOLOv8 variants in terms of model complexity, inference speed (Local and Cloud), and detection accuracy. The comparison highlights significant differences in inference times between Local and Cloud processing. While Local inference times range from 19.2 to 521.4 ms depending on the model variant, cloud-based inference is consistently faster, ranging from 1.2 to 3.9 ms. These results underscore the advantage of offloading computational tasks to a cloud where power constraints are lifted and resources are abundant, allowing the use of better models while significantly reducing inference time. Since our study focuses on real-time detection, we select the *small* variant as the lightweight detector running on-board in our Local scenario. For the Cloud scenario where power constraints are relaxed, we use the much more demanding *xlarge* variant.

7.3.7 Model Training

We trained the *small* and *xlarge* YOLOv8 variants, for the Local and Cloud respectively. We split the dataset described in Section 7.3.4 into a training set 3500 frames, a validation set of 750 frames, and a test set 750 frames. To avoid class imbalance, we include the same proportion of each class into each dataset split. All models are trained for up to a 100 epochs with early stopping and using the Adam optimizer with an initial learning rate of 0.001. Rather than learning from scratch, we leverage transfer learning by initializing models with pre-trained weights obtained by training on the COCO dataset [88]. To accelerate training, we set the batch size to the maximal value that fits in GPU memory.

7.4 Results

In this section, we present the results of our experiments and provide a detailed discussion of the findings. The experiments were conducted under typical day weather conditions featuring partly cloudy skies, on a predefined route as detailed in Section 7.3.3. To ensure the collection of statistically valid results, each experiment consisted of 10 repetitions.

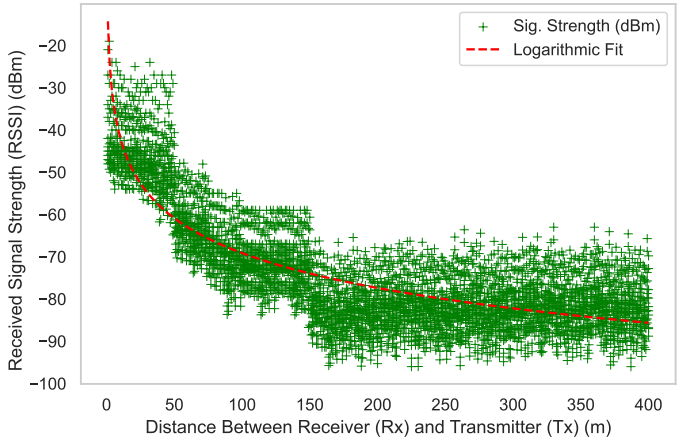
7.4.1 ITS-G5 CPM Transmission (Local scenario only)

The CPM transmission latency is influenced by various factors, such as network conditions, signal quality, and hardware capabilities, including processing time at the receiving end. These factors require a detailed investigation to understand their collective impact on the performance of CPM transmission. To ensure reliable transmission, we begin by measuring RSSI while transmitting CPM over ITS-G5. Our findings, as shown in Fig. 7.4a, reveal a consistent pattern in which RSSI values exhibit an inverse correlation with distance. As expected, with increasing distance between the sender and receiver, the RSSI decreases. However, in particular, beyond a distance of 150 m, we observed an increase in path loss. This phenomenon suggests additional factors influencing the dynamics of signal propagation beyond a distance of 150 m. Environmental variables such as buildings, terrain topology and signal interference may contribute to this observed increase in path loss [136]. Despite this deviation, the results demonstrate reliable transmission range of 150 m, and a maximum range of up to 400 m. The network parameters with transmission power and radio configuration details are summarized in Table 7.3.

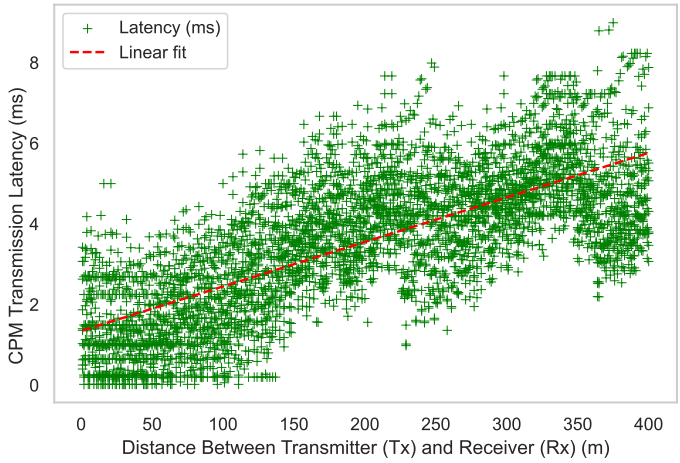
In a second experiment, we measured end-to-end CPM transmission delay between the mobile sender and static receiver. The results are shown in Fig. 7.4b. It shows that the CPM transmission latency remains consistently low, with values ranging from 1.5 ($\pm$ 1.0) ms to 6 ms ($\pm$2.5) ms, for a communication range of up to 400 m. These findings align with previous studies on ITS-G5 technologies, which have demonstrated low latency and reliable transmission of CPMs in a simulated connected driving scenarios [77].

Parameter Name	**Value**
Transmission Power (Tx)	23 dBm
Energy threshold	-85 dBm
Channel bandwidth / carrier frequency	10 MHz / 5.9 GHz
Radio Configuration	Single Channel (CCH)
Data rate	6 Mbit/s
Number of CPM Transmitted / loss ratio	4000 / 0.07

Table 7.3: ITS-G5 network parameters.
Source: Reprinted from [56], with permission, ©2024 IEEE.



(A) RSSI vs. distance for V2X communication



(B) Latency vs. distance for V2X communication

FIGURE 7.4: Demonstration of (a) RSSI measurements and (b) latency measurements, plotted against distance between receiver and transmitter during CPMs transmission using the ITS-G5. **Source:** Reprinted from [56] with permission, ©2024 IEEE.

7.4.2 Video Streaming Latency (Cloud scenario only)

To stream camera images for cloud detection, we consider two different streaming resolutions: 1920x1080 (FHD) and 1280x720 (HD). Encoding always occurs in the on-board hardware, while decoding is performed on the cloud. Depending on the compression quality, the latency for encoding and decoding ranges from 0.5 to 3 ms, which is relatively low compared to inference and transmission times. The choice of compression value of H.265, controlled by Constant Rate Factor (CRF) while keeping other parameters to defaults [139], results in streaming latencies ranging from 3 to 457.4 ms for FHD and 2.5 to 341.4 ms for HD, as illustrated in Fig. 7.5. The CRF values range from 0 to 51, where 0 corresponds to lossless compression and 51 represents the highest possible compression with the greatest loss of data. To further investigate the end-to-end delay of the cloud scenario, we define distinct settings: FHD-H (High quality) at CRF 0, FHD-M (Medium quality) at CRF 24, FHD-L (Low quality) at CRF 30, and FHD-VL (Very Low quality) at CRF 51. Similarly, we assign the same CRF values for HD, defining the following settings: HD-H, HD-M, HD-L and HD-VL.

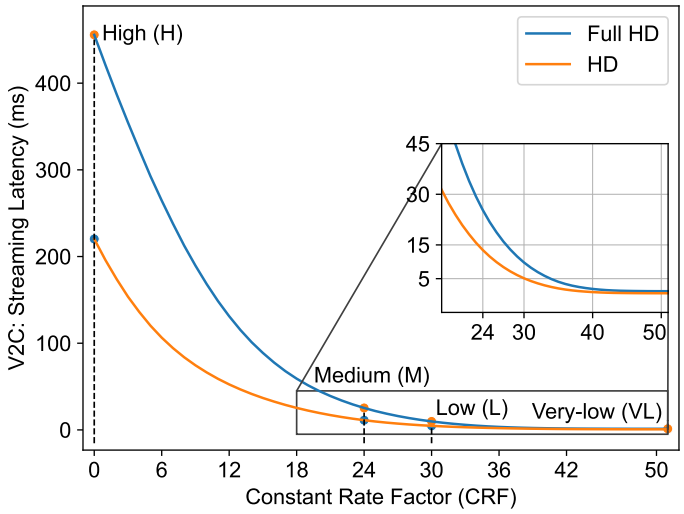


FIGURE 7.5: For cloud processing, we measured the mean latency for HD and FHD input resolutions at various H.265 compression qualities: High (H), Medium (M), Low (L) and Very Low (VL). The results focus solely on vehicle-to-cloud transmission using UDP, excluding encoding and decoding latencies. **Source:** Reprinted from [56] with permission, ©2024 IEEE.

7.4.3 End-to-End Delay

In this section, we examine the end-to-end delay of both the Local and the Cloud scenario. A breakdown of the different components of end-to-end delays for the different experiments is displayed in Fig. 7.6.

In the case of the Local scenario, the end-to-end delay primarily depends on the onboard inference time of 29.4 ms (see Table 7.2) and the V2X latency incurred by CPM transmission over ITS-G5 (see Fig. 7.4b). With an average CPM transmission latency of 7 ms for distances up to 150 meters, the total local end-to-end delay is 36.4 ms. These delays illustrate the low-latency capabilities of ITS-G5 for localized cooperative perception.

In the Cloud processing scenario, the end-to-end delay is dependent on a series of interconnected factors. These include compressing images using H.265, latency incurred during vehicle-to-cloud (V2C) communication over C-V2X in cellular mode, decoding executed within the cloud infrastructure, inference time for object detection, and transmission of detection results back to one or more vehicles (C2V) in the vicinity.

Conversely, in the Cloud processing scenario, considering FHD-M, the delay can increase to 40.8 ms, while considering HD-M, the delay decreases to 30.4 ms. For FHD-VL and HD-VL streaming, the delay decreases to 20.8 and 15.26 ms, respectively. In terms of end-to-end delay, all compression factors are viable for real-time operation, except for high compression qualities (FHD-H and HD-H). In the cases of FHD-H and HD-H, not displayed on Fig. 7.6 for scaling reasons, delay respectively reaches 457.4 and 341.4 ms. Note that inference time remains constant at 3.9 ms for all Cloud experiments since the same detection model is used.

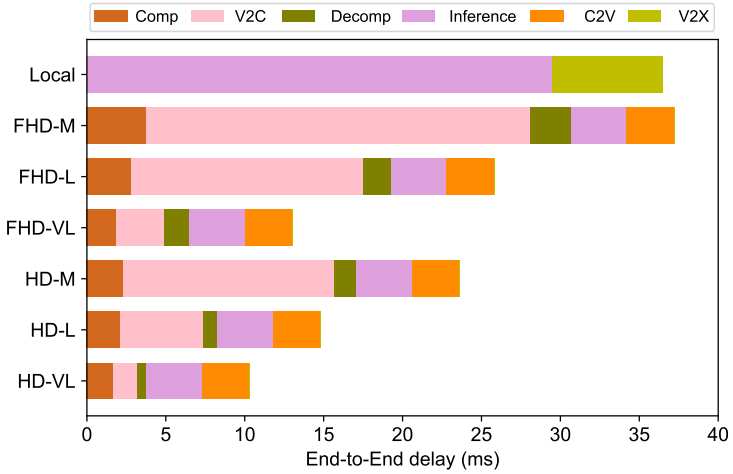


FIGURE 7.6: Breakdown of mean end-to-end delay (ms) for Local and Cloud processing scenarios. In the Local scenario, end-to-end delay depends solely on inference time and V2X (CPM) communication. For the Cloud processing, end-to-end delay depends on compression, vehicle-to-cloud (V2C) streaming, decompression, inference in the cloud, and the Cloud-to-vehicle (C2V) communication latency. **Source:** Reprinted from [56] with permission, ©2024 IEEE.

7.4.4 Detection Quality vs. End-to-End Delay Trade-off

To gain a comprehensive understanding of the proposed distributed and cooperative perception strategies, it is essential to carefully consider the trade-off between detection quality and end-to-end delay. This trade-off plays a crucial role in designing systems that effectively balance real-time responsiveness and reliable object detection. By exploring the trade-off, our objective is to identify the optimal balance between quality and end-to-end delay. Fig. 7.7 synthesizes our results by presenting different trade-offs between end-to-end delay and mean average precision (mAP), for Local and Cloud processing with different input resolutions and compression qualities.

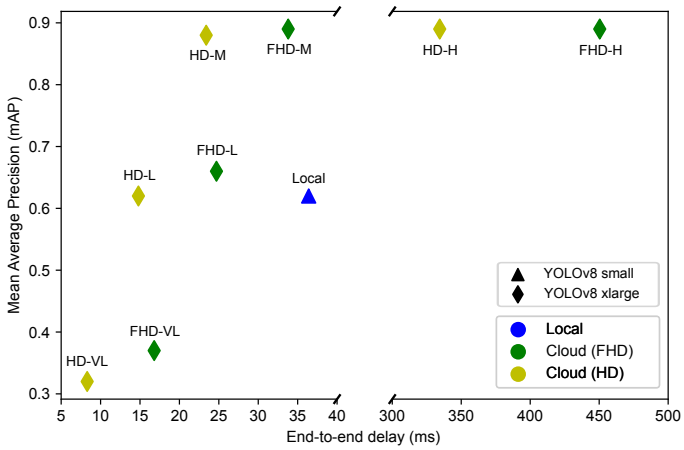
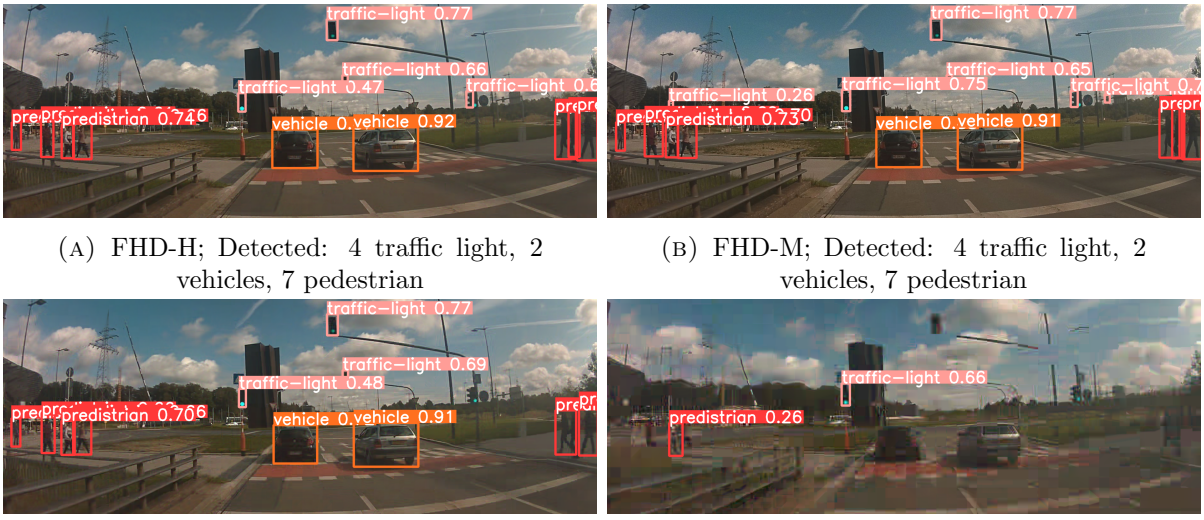


FIGURE 7.7: Comparison of mAP against End-to-end Delay for local and cloud-based processing, for two distinct input resolutions (HD and FHD), and four compression qualities ranging from High (H) to Very Low (VL). **Source:** Reprinted from [56] with permission, ©2024 IEEE.



(A) FHD-H; Detected: 4 traffic light, 2 vehicles, 7 pedestrian

(B) FHD-M; Detected: 4 traffic light, 2 vehicles, 7 pedestrian

(C) FHD-L; Detected: 2 traffic light, 2 vehicles, 6 pedestrian

(D) FHD-VL; Detected: 1 traffic light, 0 vehicles, 0 pedestrian

FIGURE 7.8: Cloud-based detection at various compression qualities. Detections are considered valid when they have 50% IoU with the ground truth. **Ground truth:** 4 traffic lights, 2 vehicles and 7 pedestrians. **Source:** Reprinted from [56] with permission, ©2024 IEEE.

In Cloud processing scenarios, a higher mAP score of 0.89 is achieved in FHD-H with an end-to-end delay of 450.4 ms. For high-quality compression scenarios (FHD-H and HD-H), the use of CRF 0 entails lossless compression and results in larger amount of data streamed to the cloud. This inevitably increases end-to-end delay in both the FHD-H (450.4ms) and HD-H (334.5ms) cases. With optimal choices of compression qualities, end-to-end delays can be lowered below 50 ms without disproportionately sacrificing mAP.

Specifically, the FHD-M and HD-M scenarios maintain a high accuracy compared to their high compression quality counterparts (0.89 mAP from FHD-H to FHD-M, 0.88 mAP from HD-H to HD-M), while drastically reducing end-to-end delay (33.8 ms from FHD-H to FHD-M, 23.4 ms from HD-H to HD-M). A further reduction in data size is achievable by streaming Very Low quality images (HD-VL and FHD-VL). However, this comes at the cost of significantly degraded mAP, even falling below the performance of the lightweight Local model. Since the HD-L model offers a comparable end-to-end delay with noticeably higher mAP, the use of Very Low compression provides no tangible benefit. These findings demonstrate that combining H.265 compression with UDP streaming of FHD data is an effective strategy for cloud-based processing. This approach reduces the computational load on onboard hardware. It also improves detection accuracy while keeping end-to-end delay low. Such a solution provides a practical complement to local processing for perception tasks.

The visual impact of lowering compression quality and its effect on detection results are illustrated on Fig. 7.8. In particular, in FHD-M settings, Cloud detection rates closely resemble those of object detection, in contrast to FHD-H settings. However, a decrease in detection instances begins with the FHD-L scenario. In FHD-VL, only a traffic light is detected, leaving other objects undetected, thereby compromising overall detection quality due to the low-quality stream.

7.5 Conclusion and Future Work

In this study, we analyze real-time object detection strategies, which include both Local and Cloud-based methodologies, and employ field trials for validation. We demonstrate the feasibility of ITS-G5 for robust transmission of CPM in local environments. Furthermore, we explore the potential of C-V2X communication to utilize cloud hardware, facilitating real-time object detection. Our findings indicate significantly improved detection quality compared to processing solely on-board hardware. To gain a comprehensive understanding of the proposed cooperative perception strategies, we investigate the trade-off between detection quality and end-to-end delay. In addition, we also create and release a real-world driving dataset annotating instances of pedestrian, vehicle, and traffic light classes providing a valuable resource for researchers. Future research will explore adaptive techniques and hybrid optimization strategies, including dynamically switching between local and cloud processing based on bandwidth availability and situational complexity, such as high-accuracy demands in areas like intersections.

Chapter 8

Hybrid Computing for 360-Degree 3D Perception

This chapter provides an in-depth exploration of perception techniques for CAV, focusing on achieving comprehensive 360-degree 3D perception. Unlike Chapter 6, which examined the offloading of data from a single front-facing camera to the cloud for 2D perception, this chapter utilizes a multi-camera setup. Using an existing dataset with six strategically positioned cameras around the vehicle, we captured a full 360-degree view in 3D space.

To optimize cloud offloading, we introduce techniques such as dynamic feature clipping, compression, and precision adjustments, collectively reducing offload latency while preserving detection accuracy. These optimizations contribute to a robust cloud-assisted perception framework that adapts to varying network conditions.

We evaluated the system in real world scenarios with integration V2X, assessing performance across key metrics such as end-to-end delay and detection reliability. The results indicate that our approach achieves low latency, making it viable for time-sensitive perception tasks in real-world driving.

This chapter builds upon the following publication:

> Faisal Hawlader, François Robinet, Elghazaly Gamal, and Raphaël Frank. "Cloud-Assisted 360-Degree 3D Perception for Autonomous Vehicles Using V2X Communication and Hybrid Computing". In: *20th Wireless On-demand Network systems and Services Conference (WONS)*. IEEE. 2025

The remainder of this chapter is structured as follows: Section 8.2 provides a review of relevant literature. Section 8.3 outlines our proposed approach, detailing the onboard and cloud components, as well as the test route and communication technologies utilized. Section 8.4 presents our experimental results, emphasizing the trade-offs between latency and accuracy. Finally, Section 8.5 concludes with a summary of our findings and potential directions for future research.

8.1 Introduction

Industry leaders, such as Tesla [102], BMW, and Mercedes-Benz, face considerable challenges in processing extensive sensor data (e.g., cameras, radar, and LiDAR) required for accurate 3D object detection [165, 160]. Recent research has aimed to meet the strict latency and accuracy requirements associated with autonomous perception tasks [57]. Models like BEVFormer [86] have shown high detection accuracy [164], yet their computational demands frequently exceed the capabilities of onboard hardware [69], leading to increased latency and energy consumption [51]. For example, an industrial report by Ford Motor indicates that future vehicles may need to allocate up to 47% of their energy for onboard computing [43], highlighting the need for more efficient processing strategies.

To address the limitations of onboard computing, researchers have proposed partitioning perception models [108] and offloading intensive layers to the cloud [22, 21]. While this approach alleviates onboard processing demands, it introduces feature transmission latency [145], which poses a challenge for real-time detection with stringent latency requirements [108]. To address the efficient transmission of large feature vectors, techniques such as post-training quantization [97] and clipping [89] can be applied to reduce bandwidth and transmission latency, enabling real-time processing in hybrid environments [89]. Further compression can decrease feature vector size [59], enhancing overall efficiency. However, quantization, clipping, and compression may reduce detection quality due to data loss [89], making it essential to balance end-to-end delay with detection quality.

In this chapter, we propose a BEVFormer-based hybrid computing strategy that integrates cooperative perception for 360-degree 3D detection. By exchanging CPM containing object detection data, this method extends perception beyond onboard sensors [146]. Our approach involves lightweight feature extraction on the vehicle, with intensive computations offloaded to the cloud, thereby combining local and cloud processing to enhance real-time performance. Our experimental results show a reduction in end-to-end delay by over 79.2% compared to onboard-only computing, underscoring the effectiveness of distributing perception tasks between the vehicle and the cloud.

In this chapter, we make the following key contributions:

Contributions

- **360-degree 3D Perception**: We benchmark onboard computing, where tasks are processed locally, and hybrid computing, where tasks offloaded to the cloud. Detection results are encoded into CPM in both cases.
- **Hybrid Computing**: We introduce dynamic feature clipping, compression, and precision adjustments to reduce offloading latency while preserving detection quality.
- **Real-world Testing**: The system is tested in real-world scenarios with V2X integration, achieving low end-to-end delay, making it suitable for real-time perception.

8.2 Related Work

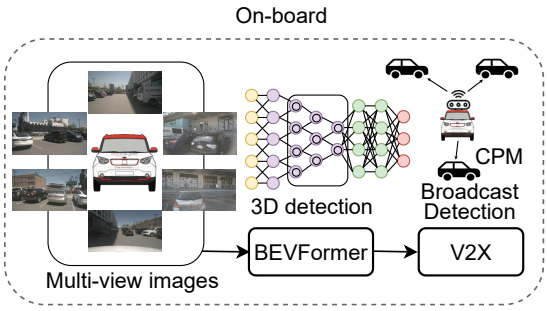
Cooperative perception [61] has gained significant attention as a method to enhance the situational awareness of autonomous vehicles [137]. By enabling vehicles to share sensor data and computational resources [46], these systems can significantly improve object detection [160] and prediction in complex environments [36]. Several recent studies have explored vehicle-to-cloud (V2C) communication to offload perception tasks to remote servers [59, 57, 165]. This approach leverages the higher computational power of the cloud to complement onboard processing [156, 43]. Early works in the field, such as [103, 58], focused on transmitting raw data to the cloud. However, these approaches suffered from bandwidth limitations and high transmission latency [45], making them unsuitable for real-time applications [124]. To address these challenges, [145, 89] introduced feature-level offloading [83], where intermediate feature vectors are transmitted instead of raw data [104], significantly reducing bandwidth usage [126]. However, the size of these feature vectors can still be prohibitively large [20, 108], especially when generated by deep neural networks like ResNet101 [62] or BEVFormer [86].

Recent studies, such as [108], have proposed various methods for compressing feature vectors [22], including quantization and lossy compression [22, 93]. Although these methods reduce transmission data size, they often result in a degradation of detection accuracy [51]. Our work builds on these efforts by introducing a dynamic clipping mechanism [89] that minimizes unnecessary feature data while retaining key information for accurate 3D object detection [159]. We also evaluate the effectiveness of lossless compression in combination with different floating-point precisions to achieve a balance between latency, bandwidth, and accuracy.

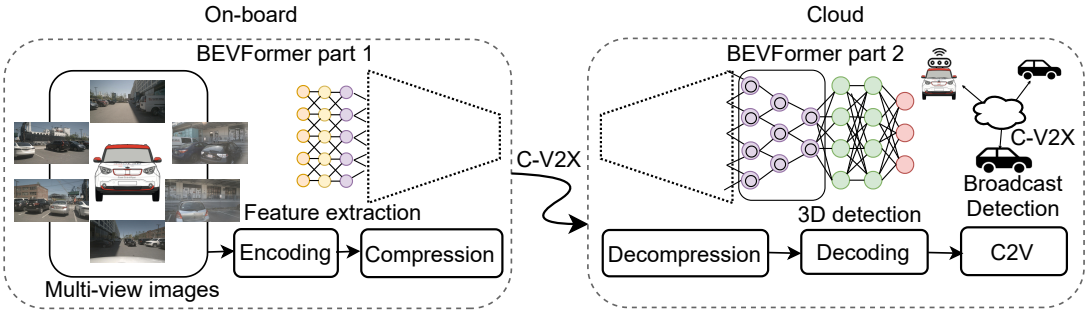
Additionally, while previous work has focused primarily on static environments or simulations, our system is tested in real-world driving scenarios, utilizing V2X communication. This enables us to account for network jitter and variability, providing a more robust evaluation of real-time capabilities and performance of the CPS.

8.3 Methodology

In this section, we outline the methodology for evaluating the proposed lightweight 360-degree 3D perception system. Multi-view (6x) camera images are processed using BEVFormer [86], a 3D object detection framework for autonomous driving. The model outputs 3D bounding boxes with object positions, orientations, and sizes in a BEV space, making it well suited for perception. Results are then encoded into CPM and broadcast to nearby vehicles or infrastructure via V2X, standardized by ETSI [146].



(A) Onboard computing scenario



(B) Hybrid computing scenario

FIGURE 8.1: Experimental scenarios: In the Onboard Computing scenario, the BEVFormer model runs locally, transmitting detection results as CPMs over ITS-G5. In the Hybrid Computing scenario, a compressed feature vector is sent via C-V2X to the cloud for intensive BEVFormer processing, with detection results broadcast to nearby vehicles. **Source:** Reprinted from [60] with permission, ©2025 IEEE.

8.3.1 Test Scenario & Route

The tests were carried out on public roads in the Kirchberg area of Luxembourg, which offers various road layouts and traffic conditions. This environment allowed us to evaluate perception scenarios in highly realistic settings. Communications between vehicles and infrastructure are handled by the YoGoKo Y-Box module, which supports the ITS-G5 and C-V2X technologies. For more information on the test vehicle, sensors, hardware, and software stack, refer to [142]. For this work, we setup two distinct scenarios, as shown in Figure 8.1:

In the **Onboard computing scenario**, multi-view images are fed into the BEVFormer model, with all perception tasks performed locally. The detection results are then encoded into CPM and transmitted to nearby vehicle or infrastructure via ITS-G5. The experimental route spanned a distance of approximately 1.5 km[1]. Within this setup, we evaluated the transmission of CPM to assess the reliability and performance of V2X communication in real world traffic. To ensure consistent measurements, we placed a stationary receiver at specific coordinates. This provided a fixed reference point for evaluating V2X communication quality as the transmitting vehicle moved along the designed test route.

[1]**Local test route:** http://g-o.lu/3/GsHC

Platform	Hardware Configuration
Local (≈30W)	NVIDIA Jetson Orin 2048 CUDA Cores, 131.4 TOPS (INT8) 8-core ARM Cortex-A78AE
Cloud (≈3000W)	2x Intel Xeon Skylake CPUs (56 cores total) 4x NVIDIA Tesla V100 GPUs (16 GB each) 20480 CUDA Cores, 500 TFLOPS (FP16)

TABLE 8.1: Hardware setups for the onboard and cloud computing platforms described in Section 8.3.1. **Source:** Reprinted from [60], with permission, ©2025 IEEE.

In the **Hybrid computing scenario**, BEVFormer is split into two parts: Part-1 handles initial feature extraction with lightweight computations onboard, while Part-2 completes perception tasks in the cloud. Multi-view images are processed by BEVFormer Part-1 to extract feature vectors, which are then clipped, compressed, and sent to the cloud via C-V2X. In the cloud, BEVFormer Part-2 completes the remaining perception tasks, including 3D object detection. The detection results are encoded into CPM and broadcast to nearby vehicles via C-V2X, enabling cooperative perception. The test route spans approximately 4km[2]. We use the cellular mode of C-V2X to communicate with a cloud server located at the University of Luxembourg in the same area. Twelve commercial base stations, including 4G and 5G (non-standalone) sites, operate along the route on Low- (700 MHz) and Mid-band (3.6 GHz) frequencies. The area provides an average download throughput of 57 Mbit/s for 4G and 115 Mbit/s for 5G, with upload speeds ranging from 25 to 35 Mbit/s for both. We use UDP for data offloading to the cloud, as it offers the lowest end-to-end delay for offloading sensor data [57].

8.3.2 Hardware Configuration & Detection Model

The hardware configurations used in this study are outlined in Table 8.1. The onboard setup utilizes a Jetson Orin, selected for its low power consumption and processing capabilities in embedded perception tasks. The cloud platform employs multiple Tesla V100 GPU nodes, designed to handle computationally intensive tasks. For more details on GPU node configurations, we refer the reader to [156].

We use the BEVFormer with ResNet101 backbone [62], initialized from the FCOS3D checkpoint [164]. BEVFormer performs multi-view 3D object detection in several stages. It begins with a ResNet101 backbone that extracts features from multi-view camera inputs. These features are transformed into a BEV representation using a view transformer that fuses spatial information from different perspectives. The BEV representation is refined by a BEV encoder, which applies self-attention to improve the detection accuracy. Finally, the detection head outputs 3D bounding boxes, predicting object positions, orientations, and sizes.

[2]**Cloud test route:** http://g-o.lu/3/96TS

For more details on the BEVFormer, we refer the reader to [86]. In our hybrid setup, we split BEVFormer after the initial backbone layers, performing feature extraction onboard. The remaining backbone layers, view transformation, BEV encoding, and 3D detection are offloaded to the cloud for efficient processing.

8.3.3 Dataset and Evaluation

For this study, we use the **nuScenes** dataset [15], a large-scale dataset specifically created for autonomous driving research. The dataset consists of images resolution of 1600x900 from six cameras, five radars, and one LiDAR, providing full 360-degree coverage, perfectly aligning with our objective of achieving comprehensive 3D detection. The dataset also includes detailed annotations for 3D object detection, tracking, and segmentation across various classes such as vehicles, pedestrians, and cyclists. In this work, we use a BEVFormer model pre-trained on the NuScenes dataset without performing any additional training. We use the nuScenes dataset solely to evaluate the performance of the model after post-quantization, clipping, and compression. The evaluation aims to assess potential detection accuracy loss resulting from different quantization levels, as well as the effects of clipping and compression. To evaluate performance across all classes, we use the **NuScenes Detection Score (NDS)**, which offers a comprehensive assessment of detection tasks. The NDS is calculated using the following formula [86]:

$$\text{NDS} = \frac{1}{10}\left(5\text{mAP} + \sum_{\text{mTP}\in\mathbb{TP}} (1 - \min(1, \text{mTP}))\right) \tag{8.1}$$

This score integrates various aspects of model performance, including mean average precision (mAP) and mean true positive (mTP), providing a robust evaluation.

8.3.4 Lightweight Features Offloading: Hybrid Computing

In hybrid computing, multi-view images captured around the vehicle are first processed by the initial backbone layers onboard to extract features. These features are then dynamically clipped and compressed to reduce their size and optimize bandwidth before transmission to the cloud. In the cloud, the remaining backbone processing, view transformation, BEV encoding, and 3D object detection are completed. This division reduces the computational load on the vehicle, while resource intensive tasks are handled in the cloud. The complete processing workflow is detailed in Algorithm 2.

Feature Vector Extraction: The backbone network B on the vehicle processes the input images X and extracts feature vectors $F(x) \in \mathbb{R}^{C\times H\times W}$ at the split layer L_{split}, where C is the number of channels, and H and W represent the spatial dimensions. The selection of L_{split} balances onboard processing and reduces data offloaded to the cloud.

Algorithm 2 Lightweight Features Offloading: Hybrid. **Source:** Reprinted from [60], with permission, ©2025 IEEE.

Given:
X: Input image, B: Backbone network, L_{split}: Split layer,
$p_{\text{lower}}, p_{\text{upper}}$: Clipping percentiles, Δt: frequency

Onboard Hardware:
Initialize input X, backbone network B, and layers L
while vehicle is driving **do**
  for each layer L in B **do**
    if $L = L_{\text{split}}$ **then**
      Compute clipping thresholds $p_{\text{lower}}, p_{\text{upper}}$:
        $L_p \leftarrow \text{Percentile}(F(x), p_{\text{lower}})$
        $U_p \leftarrow \text{Percentile}(F(x), p_{\text{upper}})$
      $F_{\text{clip}}(x) \leftarrow \max\Big(L_p,\ \min\big(F(x), U_p\big)\Big)$
      $F_{\text{comp}}(x) \leftarrow \text{Compress}(F_{\text{clip}}(x))$
      Transmit $F_{\text{comp}}(x)$ to the Cloud
  Wait Δt
On Cloud:
while receiving data from vehicle **do**
  $F_{\text{recv}}(x) \leftarrow \text{Decompress}(F_{\text{comp}}(x))$
  DetResults $\leftarrow$ BEVFormer.head($F_{\text{recv}}(x)$)
  CPM $\leftarrow$ Encode(DetResults)
  Broadcast CPM
Return: CPM containing 3D object detection results

Feature Vector Clipping & Compression: The extracted features $F(x)$ at the split layer L_{split} undergo a clipping process with lower bound L_p and upper bound U_p.

$$F_{clip}(x) = \max\left(L_p, \min\left(F(x), U_p\right)\right)$$

The system dynamically chooses the bounds (L_p, U_p) in order to clip activations to their p_{lower} and p_{upper} percentiles. Based on our experimental analysis, we set p_{lower} and p_{upper} to 10th and 90th percentiles [71], respectively, to balance bandwidth reduction and detection accuracy. By clipping values outside this range, the system removes outliers that do not contribute significantly to detection. This clipping function lowers the entropy of the activation distribution, making compression more efficient. The clipped features $F_{\text{clip}}(x)$ are compressed using a lossless zlib compression [58], further minimizing the data size for efficient transmission to the cloud.

Offloading to Cloud: After clipping and compression, the features are offloaded to the cloud via C-V2X while the vehicle follows the test route. Compressed $F_{\text{comp}}(x)$ are transmitted at regular intervals, defined by the offload frequency Δt, which adjusts according to network conditions to minimize latency [85]. The system monitors latency between transmission and cloud acknowledgment. If latency increases, indicating network congestion or slower speeds, the frequency is adjusted to prevent delays [72]. In the cloud, activations are decompressed and processed using a multi-GPU setup (Section 8.3.1), accelerating detection compared to on-board hardware. The results are encoded into CPM and transmitted to nearby vehicles and infrastructure, enhancing situational awareness through cooperative perception.

Quantization	Inference (ms)	NDS	CPM (ms)	End-to-end delay (ms)
FP32	486	0.52	5.9 (±1.8)	491.9 (±2.7)
FP16	257	0.52	5.7 (±1.7)	262.7 (±2.3)
FP8	194	0.51	4.9 (±1.6)	198.9 (±2.3)

TABLE 8.2: Performance metrics for the BEVFormer model with ResNet101 backbone, evaluated using TensorRT optimization at different quantization levels (FP32, FP16, FP8) on the onboard vehicle platform, as detailed in Section 8.3.2. The table includes inference time and CPM transmission latency, which together form the end-to-end delay. Standard deviations (±) are provided to reflect variability. **Source:** Reprinted from [60], with permission, ©2025 IEEE.

8.3.5 CPM Encoding

The CPM encoding process packages detected objects and environmental data into a standardized format defined by ETSI [146], ensuring interoperability in CPS. As illustrated in Fig. 7.3, the message structure includes several containers such as the ITS-PDU header, management, and sensor information containers, which store the reference position, sensor ID, and metadata. The encoding process is managed by the YoGoKo Y-Box module [142], which supports both ITS-G5 and C-V2X technologies, enabling seamless V2X communication.

8.4 Results

In this section, we evaluate the performance of our proposed perception system designed for autonomous driving. The system generates and transmits CPM while integrating 3D object detection to enhance environmental awareness. By offloading sensor data processing to the cloud and enabling communication with nearby vehicles, the system supports cooperative perception, allowing more accurate and timely decision making in complex driving scenarios. Repeating each experiment five times for statistical robustness, we evaluated its effectiveness using the scenarios detailed in Section 8.3.1.

8.4.1 Onboard computing and CPM transmission

To establish a baseline, we performed inference tests on the onboard platform, as detailed in Section 8.3.2, using the NuScenes dataset described in 8.3.3. These tests yielded an average inference time of 673 ms for the default model prior to optimization. This far exceeds the typical latency threshold for real-time perception in autonomous driving (e.g., less than 100 ms) [57]. Although the model achieved an NDS of 0.52, onboard processing consumed over 65% (±4) of the hardware resources, as monitored through the nvidia-smi GPU tracking. These results highlight the limitations of onboard computing, particularly regarding resource usage and time constraints.

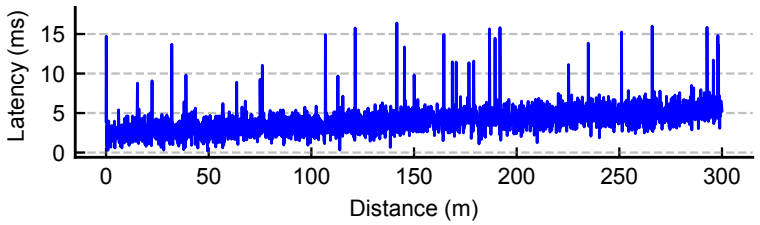


FIGURE 8.2: CPM transmission latency versus distance between a moving vehicle (25 km/h Avg.) and a stationary receiver at fixed coordinates (longitude: 6.161993, latitude: 49.626478). **Source:** Reprinted from [60] with permission, ©2025 IEEE.

Model optimization using TensorRT: To address resource usage and time constraints, we employ TensorRT for model optimization [23]. TensorRT enhances performance by applying precision calibration (FP32, FP16, or FP8) and optimizing the computational complexity of the model. The experimental results in Table 8.2 show how TensorRT optimization significantly reduces inference times without major degradation in detection. By reducing the precision from FP32 to FP16 and FP8, we observe substantial improvements in inference.

For instance, inference time drops from 486 ms (FP32) to 194 ms (FP8), with only a marginal decrease in NDS, from 0.52 to 0.51, indicating that the performance in terms of detection is largely preserved even with reduced precision. These results align with previous studies, which demonstrated that quantization effectively maintains high detection accuracy while significantly reducing inference time [109, 36].

CPM Transmission & V2X Communication: Upon detecting surrounding objects, the 3D detection results are encoded into a CPM and broadcast to nearby vehicles via V2X direct communication. The objective of this CPM transmission test was to measure the end-to-end latency in a cooperative perception scenario. Key parameters for this evaluation are summarized in Table 8.3.

A static receiver node was placed at fixed coordinates, as described in the caption of Figure 8.2. The transmitting node, located in the vehicle, as detailed in Section 8.3.1. The results of these tests, shown in Figure 8.2, illustrate how CPM transmission latency varies with the distance between the two communicating nodes. The results indicate a slight linear increase in transmission latency as the distance grows, likely due to propagation delays and intermittent network congestion. The average transmission latency was 4.10

Parameter Name	**Value**
Transmission Power (Tx)	23 dBm
Energy threshold	-85 dBm
Channel bandwidth / carrier frequency	10 MHz / 5.9 GHz
Radio Configuration	Single Channel (CCH)
Data rate	7 Mbit/s
Number of CPM Transmitted / loss ratio	6000 / 0.09

TABLE 8.3: Important network parameters for V2X. **Source:** Reprinted from [60], with permission, ©2025 IEEE.

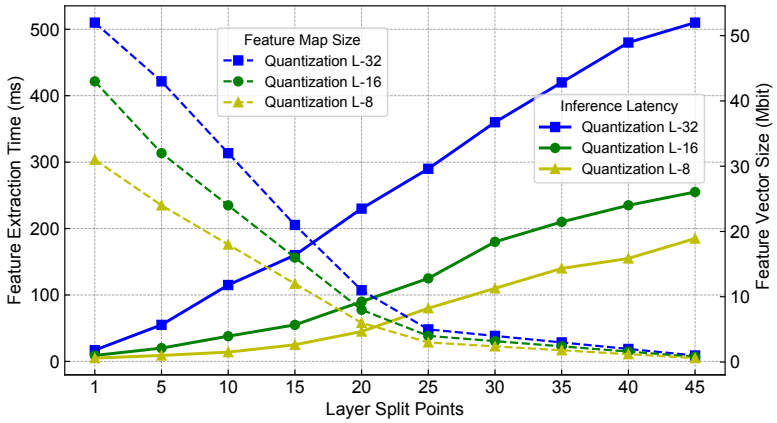


FIGURE 8.3: Feature vector size and extraction time vs. split depth. Solid lines show feature extraction time, while dashed lines indicate size. Lower quantization (FP16, FP8) reduce both extraction time and feature size. **Source:** Reprinted from [60] with permission, ©2025 IEEE.

ms, with a maximum of 18.41 ms and a standard deviation of 1.61 ms. Notably, packet loss increased significantly when the distance exceeded 300 m, highlighting sensitivity to longer distances. These observations are consistent with previous simulation-based research [147].

Although TensorRT optimizations and model quantization have significantly reduced end-to-end delay, as shown in Table 8.2, the system still falls short of the 100 ms target required for real-time perception. For example, quantization to FP8 results in an end-to-end delay of 198.9ms, which is almost double the desired threshold. Consequently, the end-to-end delay (onboard inference plus transmission) restricts CPM transmission rates to below 5Hz, highlighting the need for more efficient data processing to achieve real-time perception.

8.4.2 Cloud Computing and Lightweight Features Sharing

Offloading intensive perception tasks to the cloud, while keeping lighter tasks onboard, reduces local computation but adds transmission latency. Techniques like post-training quantization, compression, and clipping help minimize bandwidth usage and transmission time. This section explores the feasibility of lightweight feature sharing over networks, focusing on split layer selection, accuracy retention, and end-to-end delay.

Layer Partitioning and Feature Extraction: Determining the optimal partition layer in cloud processing is crucial, as it affects both the onboard feature extraction time and the size of transmitted features. In BEVFormer, partitioning earlier backbone layers (e.g., layer 1) minimizes onboard computation but requires transmitting larger feature vectors to the cloud. Conversely, deeper partitioning reduces feature vector size but increases onboard inference time. Figure 8.3 illustrates the trade-off between inference latency and feature vector size between split points and quantization levels.

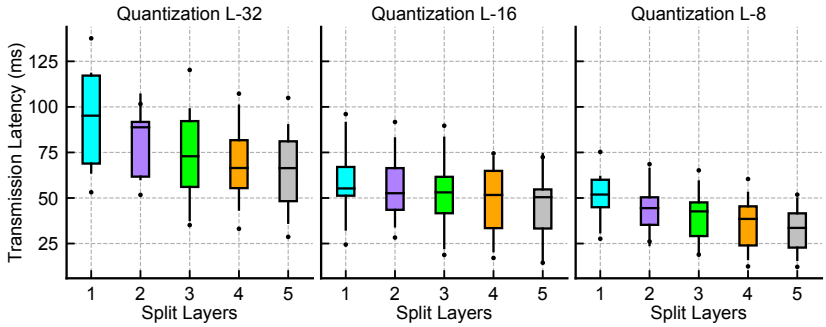


FIGURE 8.4: Transmission latency of feature vectors from vehicle to cloud across five split layers for FP32, FP16, and FP8 over a 5G network using C-V2X. FP8 demonstrates the lowest and most stable latency. FP32 exhibits the highest latency and variability, especially at earlier split layers due to larger feature size. **Source:** Reprinted from [60] with permission, ©2025 IEEE.

- **Feature extraction time**: As more backbone layers are executed locally, feature extraction time increases, significantly impacting the real time feasibility. A split at Layer 5 in FP32 yields a latency of 55 ms, acceptable for real-time use, but deeper splits quickly exceed the 100 ms threshold (e.g., 115 ms at Layer 10). Lower precisions like FP16 (20 ms at Layer 5) and FP8 (9 ms at Layer 5) reduce latency, but the benefits diminish with deeper layers, where even FP8 exceeds 140 ms by Layer 30. This highlights the need for optimal split points and further optimizations to meet real-time constraints.
- **Feature Vector Size:** Shallow splits generate large feature vectors, making real-time transmission challenging. For example, FP32 at Layer 1 produces 52 Mbit, requiring a throughput of 520 Mbit/s at 10 Hz, far exceeding typical V2X bandwidth. Even FP16 (43 Mbit, 430 Mbit/s) and FP8 (31 Mbit, 310 Mbit/s) remain too large. Deeper splits, however, reduce vector sizes: FP32 at Layer 25 requires 5 Mbit (50 Mbit/s), and FP8 only 3 Mbit (30 Mbit/s), which are more suitable for real-time transmission. Nevertheless, deeper splits increase onboard processing time, requiring a trade-off between transmission latency and feature size.

To reduce activation sizes beyond what quantization can offer, we also rely on dynamic clipping and compression.

Feature Transmission and Compression: Dynamic clipping and zlib compression were applied to reduce feature vector sizes and transmission latency across quantization levels. These techniques reduced feature sizes by approximately 97% for FP32, 90% for FP16, and 80% for FP8, significantly improving transmission efficiency across all layers. The larger reductions for FP32 are expected, as higher precision data contains more entropy, making it more compressible compared to FP16 and FP8. We focus on layers 1 to 5 because deeper splits yielded minimal improvements in transmission latency while increasing feature extraction time. As shown in Figure 8.4, FP8 exhibited the lowest latency, with medians of 52 ms at Layer 1 and 35 ms at Layer 5, meeting the threshold required for real-time perception systems.

FP16 provided a balanced latency, with medians of 67 ms at Layer 1 and 45 ms at Layer

Q-L	Split Layer	Onboard Processing Time		Transmission latency		Cloud Processing Time		End-to-end Delay (ms)	NDS	Bandwidth Usage (Mbit/s)
		Backbone	Compression	V2C	C2V	Decompression	Head			
32	1	17.2 (±2.10)	10.7 (±1.85)	65.8 (±4.00)	11.6 (±1.15)	2.6 (±0.60)	20.8 (±1.35)	128.7 (±4.20)	0.52	10.5
	2	22.3 (±1.75)	8.6 (±1.55)	58.0 (±3.90)	9.8 (±0.95)	2.4 (±0.55)	18.4 (±1.25)	119.6 (±3.90)	0.50	8.4
	3	30.5 (±1.90)	7.3 (±1.50)	48.9 (±3.75)	8.4 (±0.85)	2.5 (±0.50)	15.9 (±1.30)	113.5 (±3.80)	0.48	6.8
	4	39.8 (±1.65)	6.4 (±1.30)	54.5 (±3.50)	7.0 (±0.75)	2.3 (±0.45)	14.7 (±1.10)	124.7 (±3.60)	0.47	5.9
	5	55.4 (±1.45)	5.1 (±1.10)	56.3 (±3.20)	5.8 (±0.70)	2.5 (±0.40)	12.6 (±0.95)	137.7 (±3.40)	0.46	5.4
16	1	9.3 (±1.70)	9.1 (±1.50)	57.6 (±3.10)	12.7 (±0.95)	2.1 (±0.50)	18.2 (±1.30)	109.0 (±3.90)	0.51	9.0
	2	11.7 (±1.50)	7.3 (±1.30)	39.3 (±2.95)	7.6 (±0.85)	2.2 (±0.45)	16.6 (±1.20)	84.7 (±3.70)	0.49	6.6
	3	15.3 (±1.35)	6.2 (±1.20)	44.1 (±2.80)	6.6 (±0.80)	2.1 (±0.40)	14.3 (±1.05)	88.7 (±3.50)	0.47	5.6
	4	18.5 (±1.25)	5.2 (±1.05)	42.3 (±2.65)	8.6 (±0.75)	2.0 (±0.35)	13.4 (±0.95)	90.0 (±3.25)	0.46	4.6
	5	20.4 (±1.10)	4.3 (±0.95)	31.2 (±2.50)	7.1 (±0.70)	2.0 (±0.30)	12.2 (±0.85)	77.2 (±3.05)	0.45	4.3
8	1	5.1 (±1.45)	8.2 (±1.45)	33.6 (±2.80)	9.8 (±0.90)	1.6 (±0.50)	15.5 (±1.25)	73.8 (±3.90)	0.47	8.4
	2	6.2 (±1.25)	6.7 (±1.30)	40.4 (±2.60)	8.1 (±0.85)	1.6 (±0.45)	14.1 (±1.15)	77.1 (±3.60)	0.46	6.9
	3	7.3 (±1.10)	5.7 (±1.10)	44.3 (±2.40)	6.3 (±0.80)	1.5 (±0.40)	12.6 (±1.00)	77.7 (±3.50)	0.44	5.5
	4	8.4 (±1.05)	4.7 (±1.00)	33.4 (±2.20)	5.6 (±0.75)	1.5 (±0.35)	11.9 (±0.90)	65.5 (±3.25)	0.43	4.7
	5	9.1 (±0.90)	3.6 (±0.90)	29.3 (±2.00)	7.0 (±0.70)	1.4 (±0.30)	11.0 (±0.85)	61.9 (±3.00)	0.43	4.1

TABLE 8.4: Performance metric for various split layers and quantization levels, including onboard processing time, V2C and C2V transmission latency, cloud processing time, and total end-to-end delay. The last column shows bandwidth utilization for offloading feature vectors from vehicle to cloud. **Source:** Reprinted from [60], with permission, ©2025 IEEE.

5, making it suitable for applications that can tolerate moderate delays. In contrast, FP32 exhibited the highest latency, reaching 90 ms at Layer 1 and 70 ms at Layer 5, and was more sensitive to network jitter, as indicated by larger outliers. Despite providing higher precision, the elevated latency of FP32 renders it impractical for time-sensitive decisions without further optimization. FP8, with its minimal jitter and consistently low latency, emerges as the most viable option for cloud-based 3D object detection. As network demands increase with the adoption of 5G and beyond, the scalability of this approach, particularly with FP8, position it well for future real-time perception systems. While trade-offs between quantization levels exist, FP8 is ideal for latency critical environments, whereas FP16 and FP32 are more suitable for scenarios that prioritize higher data quality.

End-to-end Delay: The results in Table 8.4 summarize the end-to-end delay and the impact of split layers and quantization levels. For FP32 quantization at split Layer 1, the total end-to-end delay is 128.7 ms, with local processing time (backbone and compression) contributing 27.9 ms, and transmission latency (V2C and C2V) adding 77.4 ms. In contrast, FP8 at the same split layer shows a significantly reduced total delay of 73.8 ms, mainly due to the lower local processing time of 13.3 ms and a reduced transmission latency of 43.4 ms. Lower quantization levels, such as FP8, reduce both computational burden and transmission latency, though they introduce a slight trade-off in accuracy, with the NDS for FP8 at split Layer 1 being 0.50 compared to 0.52 for FP32. As the split Layer deepens (e.g., Layer 5), onboard processing time increases due to the more complex feature extraction. For FP32, the total delay at split Layer 5 increases to 137.7 ms, with 60.5 ms for local processing time and 62.1 ms for transmission latency. In comparison, FP8 achieves a lower end-to-end delay of 61.9 ms at the same split Layer, primarily due to its reduced local processing time of 12.7 ms and transmission latency of 36.3 ms. This reduction in delay for FP8 comes with only a slight decrease in accuracy, as the NDS drops marginally to 0.45. Cloud processing times (decompression and head) remain low across all quantization levels due to the powerful cloud hardware, with decompression times ranging from 1.4 to 2.6 ms, depending on the quantization level. This consistent cloud performance ensures that most of the delay comes from local processing time and transmission latency, highlighting the importance of selecting the optimal split point and quantization level for real-time systems. Transmission latencies exhibit greater variability

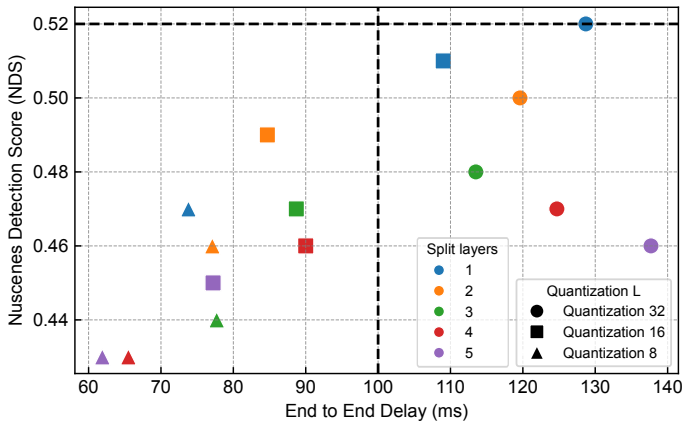


FIGURE 8.5: End-to-end delay vs. detection accuracy (NDS) across split layers and quantization levels. Earlier layers result in higher accuracy but increased delay, while intermediate layers provide a balance results. **Source:** Reprinted from [60] with permission, ©2025 IEEE.

compared to local or cloud latencies, with the largest deviations observed at split layer 1 for FP32 (±4.00 ms for V2C) and FP8 (±2.80 ms for V2C), reflecting the impact of fluctuating network conditions on the transmission process. Although lower quantization levels like FP8 reduce total delay, they introduce a slight degradation in accuracy. Optimizing split layer, quantization, compression, and clipping based on real-time constraints and network bandwidth enhances performance and reliability in cloud-based cooperative perception.

End-to-end Delay vs. Accuracy Trade-off: Figure 8.5 illustrates the trade-off between end-to-end delay and NDS across different split layers and quantization levels. This suggests that while FP32 maintains accuracy, it is highly sensitive to the increased computational load of deeper layers, making it less practical for real-time systems with strict latency constraints. Conversely, FP16 and FP8 provide more favorable trade-offs. FP16 at Layer 5 reduces latency to 77.2 ms, with a modest NDS of 0.45. FP8 offers the lowest delay, achieving 61.9 ms at Layer 5 while maintaining an acceptable NDS of 0.43. Smaller feature vectors also reduce the risk of network-induced latency and packet loss, enhancing robustness in real-world deployments. Intermediate splits, such as Layer 3 with FP16, provide a balanced trade-off between end-to-end delay and accuracy. Layer 3 with FP16 achieves a delay of 88.7 ms and an NDS of 0.47, making it a practical solution for perception systems that require both timely responses and reasonably accurate detection, a visual demonstration of the result is shown in Figure 8.6. In scenarios where accuracy is the primary concern, FP32 at shallower layers remains the best choice, although it comes at the expense of higher delay.

8.5 Conclusion and Future Work

This study evaluates cloud-based 3D object detection using a BEVFormer based model integrated with V2X communication. We propose a hybrid computing strategy that

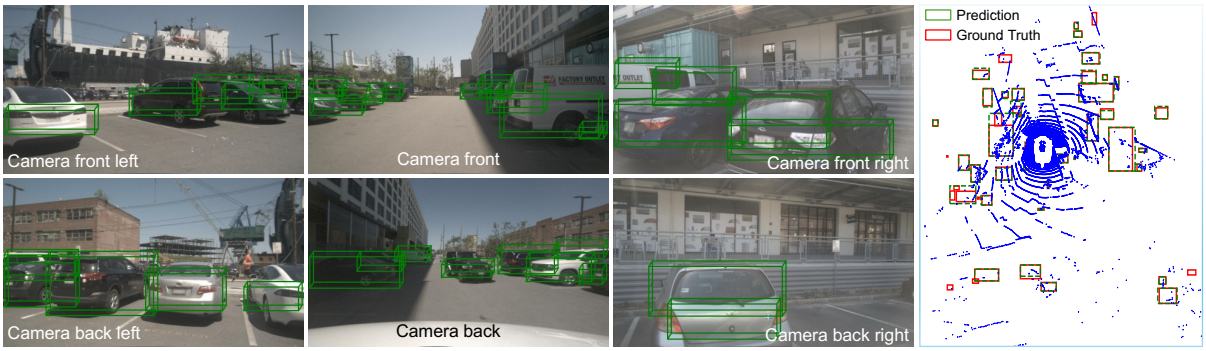


FIGURE 8.6: Qualitative results at split layer 3 with FP16 on the nuScenes validation set (SceneID: n008-2018-05-21-11-06-59-0400), an NDS score of 0.47. We show 3D bounding box predictions in multi-view (6x) camera images and the 360-degree BEV. **Source:** Reprinted from [60] with permission, ©2025 IEEE.

leverages cooperative perception for 360-degree detection. The approach offloads intensive computations to the cloud while maintaining lightweight feature extraction onboard, enabling real-time perception. Experimental results demonstrate that dynamic clipping, compression, and 5G-enabled C-V2X transmission significantly optimize latency and bandwidth utilization.

For instance, offloading FP32 feature vectors at 10Hz from Layer 1 reduced bandwidth usage by over 95%, from 520 Mbit/s to 10.5 Mbit/s. We also investigate the trade-offs between end-to-end delay and detection quality across various split layers and quantization levels. Additionally, TensorRT optimizations were applied to further enhance inference speed. Our results demonstrate that while FP32 offers the highest accuracy, its substantial end-to-end delay renders it impractical for real-time applications. In contrast, FP8 achieves significantly lower latency with reasonable accuracy, making it suitable for latency-sensitive scenarios. FP16 provides a balanced trade-off between accuracy and latency, fitting applications that require both timely responses and adequate detection performance. This study underscores the importance of selecting appropriate split layers and quantization levels based on operational requirements. Shallower splits with FP32 are optimal for accuracy-focused tasks, whereas deeper splits with FP8 cater to applications with strict latency constraints.

Part V

Challenges, Conclusion, and Future Directions

Chapter 9

Open Challenges and Lessons Learned

This chapter explores the challenges and insights from developing and evaluating perception systems for CAVs. It focuses on open issues, including network reliability, scalability in large-scale deployments, and the balance between resource demands and real-time processing requirements. The discussion highlights the potential of V2X network technologies, particularly in enhancing scalability and enabling real-time object detection.

Finally, the chapter concludes with a summary of lessons learned and a forward-looking roadmap for advancing real-time perception technologies. Key directions include adaptive computing strategies, optimized mixed-precision data handling, and integrating emerging technologies like 6G to enhance efficiency and reliability.

9.1 Open Challenges

Implementing cooperative perception systems for CAVs involves several critical challenges that significantly affect performance. These include managing distributed processing across heterogeneous platforms, ensuring reliable data exchange over dynamic vehicular networks, and meeting strict real-time latency constraints. Addressing these challenges requires efficient task allocation between local and cloud resources, as well as strategies to mitigate packet loss and network delays for reliable deployment, as shown in Figure 9.1.

9.1.1 Distributed Processing

While distributed processing enhances perception accuracy under low-latency constraints, it also presents challenges. Key issues include task allocation between vehicles and servers, maintaining synchronization across distributed platforms, and addressing variations in computational capabilities. Furthermore, when multiple vehicles offload data from the same scene to the cloud, redundant processing can strain computing and network resources, increasing overall latency.

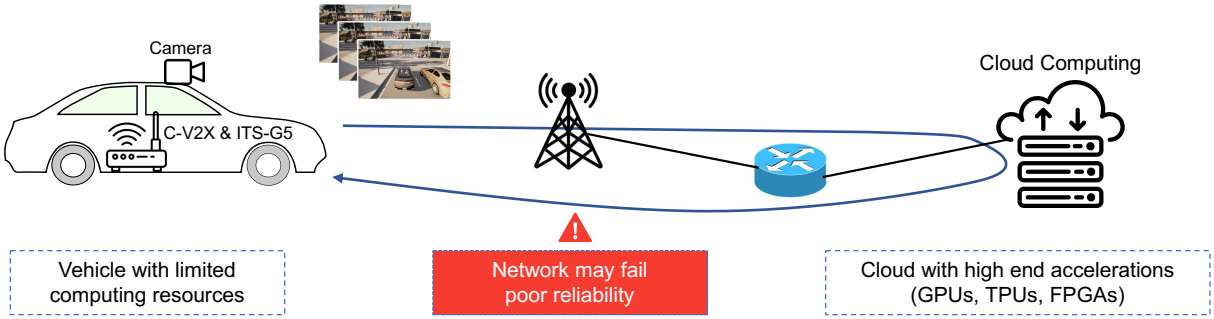


FIGURE 9.1: Streaming high-resolution sensor data significantly increases bandwidth requirements. In networks with variable throughput and latency, this can lead to transmission failures.

Effectively addressing these issues requires advanced resource allocation strategies capable of dynamically balancing workloads and minimizing redundancy. While this challenge is not fully explored in this dissertation, it highlights an important direction for future research to improve scalability and efficiency in distributed processing for cooperative perception systems.

9.1.2 Network Reliability

Our experiments demonstrated the significant advantages of V2X communication in enhancing cooperative perception. However, numerous challenges persist, particularly in dynamic vehicular network environments. Reliable data exchange is critical for cooperative perception systems, as it directly affects the quality of perception. Ensuring this reliability remains challenging due to factors such as high mobility, signal interference, and network congestion, which frequently lead to packet loss, delays, and bandwidth instability. These issues are further exacerbated by poor connectivity and external interference.

In high-speed scenarios, frequent handovers between communication nodes can disrupt connectivity, while urban areas often experience increased interference and jitter due to dense traffic and infrastructure. Existing protocols, such as C-V2X and ITS-G5, address some of these challenges through features like direct V2V communication and message prioritization. However, further advancements are required to achieve the desired levels of performance and reliability. Techniques such as adaptive bandwidth allocation, advanced error correction methods, and multi-link communication strategies hold potential for improving system reliability. Emerging technologies, including 6G and satellite-based communication systems, also show promise in addressing these challenges, particularly in remote or rural areas. These aspects, while promising, are beyond the scope of this study and remain an open area for future research.

9.1.3 Real-Time Performance

Future CAVs are expected to integrate high-resolution sensors, such as LiDAR and multi-camera systems, to enhance perception capabilities and situational awareness. However, this study focuses specifically on camera data, leaving the complexities of sensor fusion, such as combining LiDAR point clouds with camera outputs for future research. Even with a single sensor modality, achieving real-time performance remains a significant challenge.

The perception pipeline, which encompasses data collection, feature extraction, compression, transmission, and inference, faces critical bottlenecks when processing high-resolution camera inputs. These data streams place significant demands on both computational and network resources. Offloading to the cloud alleviates local processing burdens but introduces additional latency, particularly under variable network conditions. Incorporating additional sensors such as LiDAR would further increase the data volume and fusion complexity, underscoring the need for continued research in this area.

9.1.4 Scaling and Adoption

Scaling cooperative and distributed perception systems is essential to enable widespread deployment and realize their full potential in CAVs. However, this goal presents both technical and practical challenges. From a technical standpoint, ensuring seamless interoperability across diverse vehicle platforms and sensor technologies is a critical hurdle. This challenge is compounded by the need to integrate communication standards such as C-V2X and ITS-G5. Additionally, large-scale deployments generate substantial data volumes and significantly increase network traffic, which can strain communication infrastructures and adversely affect system performance. While technical challenges focus on system interoperability and performance, practical challenges primarily revolve around infrastructure investments and regulatory considerations. Scaling these systems requires considerable investment in infrastructure, including roadside units, cloud computing resources, and reliable communication networks. These investments must also align with regulatory compliance and support ongoing standardization efforts.

Overcoming these challenges will require developing cost-effective solutions and advanced data-sharing protocols for large-scale operations. Next-generation technologies, such as 6G networks and satellite communication systems, offer significant potential to enhance scalability and reliability. For example, 6G networks promise ultra-low latency and high bandwidth, which are critical for supporting real-time data sharing across large-scale vehicular networks. Achieving these advancements will also necessitate close collaboration among industry stakeholders, researchers, and policymakers to establish standardized frameworks and drive widespread adoption across real-world environments. Collaboration is essential to harmonize technical standards, align regulations across regions, and encourage investments from private and public sectors.

Addressing these challenges is vital to unlock the potential of cooperative perception systems and pave the way for smarter, safer transportation systems globally.

9.2 Broader Reflections

This section explores the broader implications of our work, highlighting its potential to advance technologies for CAVs. By situating the research within societal, industrial, and academic contexts, it reflects on the impact, interdisciplinary nature, and future potential of the contributions. These reflections also provide a foundation for addressing current limitations and driving future progress.

9.2.1 Contributions and Impact

This dissertation presents several key contributions to the field of cooperative perception, including the creation of a versatile simulation framework that integrates various platforms. The framework supports the evaluation of cooperative perception strategies by simulating realistic traffic scenarios, environmental dynamics, and detailed communication channels essential for sensor data sharing and network analysis. By addressing the limitations of existing tools, it provides a robust solution for accurately modeling sensor interactions and multi-vehicle communication.

Another significant contribution is the development of hybrid computing strategies that delegate resource intensive computations to the cloud while maintaining lightweight feature extraction on the vehicle. This method effectively minimizes latency and bandwidth usage, facilitating real-time perception with little impact on detection accuracy. By rigorously evaluating techniques such as clipping, quantization, and compression, this work provides practical approaches for optimizing resource efficiency. The significance of these contributions extends beyond academic research, offering practical value for real-world implementation. By demonstrating scalable and reliable cooperative perception methods, this work establishes a strong foundation for deploying such systems in real-world scenarios. Furthermore, it emphasizes the trade-offs between detection quality, latency, and bandwidth, providing critical insights for designing advanced connected driving systems and shaping the future of intelligent transportation.

9.2.2 Interdisciplinary Impact

The research presented in this dissertation spans multiple disciplines, emphasizing the integration of edge and cloud computing, communication technologies, and perception systems for autonomous driving. By integrating V2X communication with advanced deep learning models like BEVFormer, it highlights the convergence of computer vison and networking. Through the adoption of lightweight feature sharing techniques, this work tackles the challenges of bandwidth constrained environments, effectively bridging the gap between machine learning and wireless communication technologies.

The simulation frameworks and methodologies developed in this research also intersect with fields such as transportation engineering and urban planning. They provide valuable tools for analyzing the effects of cooperative perception on traffic dynamics, reducing congestion, and enhancing safety in complex urban environments. Furthermore, this research establishes a foundation for future collaborations between academia and industry, driving innovation at the crossroads of automotive engineering, data science, and telecommunications. The interdisciplinary scope of this thesis advances autonomous vehicle technology while opening new avenues for exploring cooperative systems across related domains.

9.2.3 Societal Impact

The strategies presented in this work hold significant value for both society and industry. Cooperative perception facilitates real-time exchange of sensor data, tackling critical issues in road safety and traffic management. By improving the situational awareness of individual vehicles, this approach reduces the dangers posed by blind spots and occlusions, paving the way for safer and more efficient mobility solutions.

From an industrial standpoint, the methodologies and frameworks developed in this research are well-suited for integration into smart city initiatives, enabling more effective traffic management and autonomous vehicle operations. Technology providers can utilize these findings to refine their designs for V2X-enabled CAVs, lowering development costs while improving overall performance. Additionally, the adoption of hybrid computing strategies aligns with the industry increasing emphasis on cloud, edge architectures, providing scalable solutions for real-time perception. The societal benefits also encompass environmental sustainability, with better traffic management and fewer accidents leading to reduced emissions and minimized resource waste. This research contributes to the broader goal of creating safer, more sustainable, and efficient transportation systems.

9.2.4 Contributions to the Research Community

This work makes a meaningful contribution to the research community by offering publicly available datasets generated from both simulation based and real world experiments. These datasets include detailed annotations of sensor outputs, serving as essential benchmarks for the advancement of cooperative perception research. By providing these resources, the study encourages collaboration and supports innovation, enabling other researchers to validate the findings and expand upon the methodologies proposed.

By showcasing the integration of realistic driving scenarios with robust communication models, this simulation framework provides a pathway for testing advanced algorithms and analyzing trade-offs in hybrid computing systems. It offers a valuable example for the field, enabling researchers to evaluate complex V2X interactions and system performance without relying on costly real-world setups.

Furthermore, the insights gained from this thesis open new avenues for research, including adaptive compression methods, mobility aware perception frameworks, and the adoption of emerging technologies like 6G. By bridging theoretical concepts with practical implementations, this work establishes a solid foundation for advancing the capabilities of autonomous driving systems.

9.3 Lessons Learned

The design, testing, and evaluation of cooperative perception systems have provided valuable lessons, highlighting key technical and practical considerations. These insights serve as a roadmap for refining system designs and addressing existing challenges.

9.3.1 Data Compression

One of the most critical insights gained was the inherent trade-offs between detection quality and end-to-end delay. Effective data compression is essential for minimizing communication overhead without compromising perception accuracy. Techniques such as mixed-precision formats (e.g., FP16 and FP8) and entropy clipping demonstrated significant reductions in bandwidth usage while maintaining system performance. However, the choice of compression strategy must align with real time requirements, balancing compression efficiency and data transmission latency constraints. This highlights the need for adaptive compression mechanisms that can adjust to dynamic operating conditions.

9.3.2 Evaluation Framework

Simulation frameworks, such as the extended CARLA-SUMO co-simulation platform, are essential tools for testing cooperative perception strategies. These frameworks allow for the exchange of realistic sensor data between simulated vehicles. They also enable the modeling of diverse traffic conditions and environmental scenarios, including varying vehicle densities and challenging road layouts. Simulations offer a scalable and cost-effective alternative to field trials by reducing the need for expensive hardware and controlled testing environments.

However, simulations have limitations. They cannot fully replicate real-world complexities like unpredictable network interference, sensor noise, or extreme weather conditions. This gap highlights their role as a complement, not a replacement, to real-world validation. For example, while simulations can evaluate data-sharing protocols under ideal conditions, real-world tests are necessary to assess system performance under dynamic and uncertain environments.

A key takeaway is the need for continuous refinement of simulation frameworks. Enhancements like more accurate sensor models, realistic communication channel emulations, and dynamic environmental interactions can improve their reliability. Combining these improvements with targeted real-world tests ensures robust evaluation of cooperative perception systems.

9.3.3 Computing Resources

Efficient transmission of sensor data and intermediate features emerged as a major challenge, particularly in bandwidth constrained environments. Techniques like H.265 compression and feature quantization reduced transmission sizes by up to 98%, enabling real-time perception. However, excessive compression degraded detection quality for smaller objects like pedestrians, underlining the need for adaptive optimization strategies. Future systems must prioritize methods that dynamically balance resource utilization and detection performance in response to real-time network conditions.

9.3.4 Practical Constraints

Deploying cooperative perception systems in real-world environments revealed several non-technical challenges, including regulatory compliance, privacy concerns, and logistical complexities associated with large-scale field trials. Addressing these challenges requires collaboration among researchers, policymakers, and industry stakeholders. Clear guidelines on data sharing and ownership, coupled with privacy-preserving protocols, are necessary to build public trust and streamline regulatory approval. Incorporating these considerations during the design phase can help mitigate delays in deployment.

9.4 Conclusion

This chapter provided a detailed overview of the challenges, reflections, and lessons learned during the development of cooperative perception systems for CAVs. Key issues, including scalability, security, network variability, and resource management, were identified as central to the successful deployment of these systems. The analysis underscored the importance of adaptive compression techniques, hybrid evaluation frameworks, and dynamic task allocation in overcoming technical barriers.

Moreover, addressing practical constraints, such as regulatory compliance and interoperability, is critical to enabling real-world adoption. These insights emphasize the need for interdisciplinary collaboration and innovative approaches to advance cooperative perception systems. By building on the findings presented in this study, future research can pave the way for scalable, reliable, and real-time systems that accelerate the integration of autonomous driving technologies into modern transportation networks.

Chapter 10

Conclusion

In this concluding chapter, we synthesize the core contributions of this dissertation and situate them within the broader context of cooperative perception systems and their role in advancing autonomous driving technologies. The research has proposed a hybrid computing framework that strategically partitions processing tasks between onboard and cloud systems, enabling real-time perception for CAVs. Rigorous real-world evaluations have provided valuable empirical benchmarks, addressing key metrics such as latency, detection quality, and scalability. Additionally, the work explores how cooperative perception can leverage emerging V2X communication standards and advanced network technologies to enhance system reliability and adaptability.

Beyond technical advancements, this dissertation reflects on the broader implications of cooperative perception, including its potential to reshape transportation systems through improved safety, efficiency, and environmental sustainability. The chapter concludes by outlining future directions that can build on this foundation, addressing unresolved challenges, and unlocking further opportunities for innovation.

10.1 Summary and Contributions

This dissertation addresses the challenge of achieving scalable, reliable, and real-time perception in CAVs by introducing a hybrid computing strategies that optimizes latency and detection accuracy through lightweight data techniques, such as clipping, quantization, and H.265 compression. Real-world testing established benchmarks for critical metrics like end-to-end delay and detection quality, revealing trade-offs between latency and accuracy. Key contributions include the enhancement of the CARLA-SUMO co-simulation framework for realistic sensor data sharing and the release of annotated datasets.

This research extends beyond technical contributions by tackling critical challenges such as bandwidth optimization, scalability, and multi-vehicle collaboration. It establishes a framework for deploying cooperative perception systems in practical scenarios. The broader societal benefits include increased road safety, reduced traffic congestion, and enhanced energy efficiency, highlighting cooperative perception as a sustainable and adaptive mobility solution.

This dissertation investigates six key research questions, outlined in Section 1.2.2. These questions focus on designing evaluation frameworks and methodologies to enable efficient, scalable, and reliable sensor data processing. The research addresses the challenge of achieving real-time perception in CAVs through cooperative strategies augmented by infrastructure support. The answers to these questions are discussed in detail throughout the dissertation and summarized as follows:

RQ1: How can existing vehicular simulation frameworks be leveraged to effectively evaluate cooperative perception solutions?
To address RQ1, we investigated realistic modeling of perception in vehicular simulations using two well established frameworks, SUMO and CARLA. The objective was to develop an evaluation framework for cooperative perception systems. We compared two distinct approaches grid-based and vision-based for modeling perception sensors within simulation environments. Our analysis examined the impact of vehicle layout, connection ratios, weather conditions, and distance on perception accuracy. These contributions are detailed in Chapter 3 and our publications [53, 54].

RQ2: Can raw sensor data be efficiently shared for cooperative perception?
To address RQ2, we investigated the feasibility of sharing raw sensor data for cooperative perception in real-time scenarios. The objective was to analyze bandwidth requirements and explore strategies for efficient data transmission. Our findings revealed that raw sensor data imposed excessive bandwidth demands, rendering direct sharing impractical for real-time applications. These contributions are detailed in Chapter 4 and our publications [55].

RQ3: What can be the best offloading strategies for real-time perception?
Building on RQ2, we proposed a framework and offloading strategies to tackle transmission latency and network resource limitations. These strategies leverage H.265 and JPEG streaming for real-time perception. We evaluated camera frame transfer and processing on edge and cloud platforms to identify the optimal trade-off between accuracy and end-to-end delay. The results, detailed in Chapter 5 and our papers [57, 59], demonstrated that H.265 and JPEG compression significantly reduced bandwidth requirements. These methods achieved real-time constraints of 20 Hz while maintaining detection accuracy.

RQ4: How can model partitioning and lightweight data-sharing techniques improve the efficiency of cloud-based real-time perception?
To address RQ4, we investigated strategies to optimize cloud-based perception systems for cooperative driving applications. The objective was to minimize bandwidth requirements while balancing end-to-end delay and detection accuracy. Our analysis focused on a model partitioning strategy that divides computational tasks between onboard systems and the cloud, reducing transmission latency through early network splitting. We also evaluated the effects of quantization and clipping on detection accuracy and latency, offering practical guidelines for cooperative perception. These contributions are detailed in Chapter 6 and our publications [58].

RQ5: How can cooperative perception systems be validated under real-world conditions to ensure reliable real-time performance?

To address RQ5, building on RQ1–4, we validated cooperative perception systems in real-world driving environments. The objective was to assess the performance of offloading strategies under practical conditions, focusing on challenges such as latency, communication constraints, and accuracy trade-offs. We developed a validation framework through field trials to evaluate real-time perception. This framework incorporates a real-world driving dataset captured with a front-facing rooftop camera and annotated with pedestrian, vehicle, and traffic light classes. The dataset supports the training and evaluation of object detection models for cooperative driving scenarios. We evaluated onboard and cloud processing approaches to enhance real-time object detection and reduce latency. For cloud offloading, we applied H.265 video encoding and analyzed the trade-off between end-to-end delay and detection accuracy, balancing bandwidth requirements. These contributions are detailed in Chapter 7 and our publications [56].

RQ6: How can real-time cooperative perception systems be advanced to achieve robust 360-degree 3D detection in real-world V2X environments?
To address RQ6, we extended the development of cooperative perception systems to tackle real-world challenges in 360-degree 3D detection within V2X environments. We benchmarked onboard computing, where tasks are processed locally, against hybrid computing, where tasks are offloaded to the cloud. To enhance the hybrid approach, we introduced dynamic feature clipping, compression, and precision adjustments, significantly reducing offloading latency while maintaining detection quality. Real-world testing with V2X integration confirmed the system suitability, achieving low latency and robust 360-degree 3D perception. These contributions are detailed in Chapter 8 and our publications [60].

10.2 Future Research Directions

While this dissertation has made significant progress in advancing cooperative perception systems for CAVs, several areas remain open for further investigation. One key area is improving the resilience of the system in challenging real-world conditions. Adverse environments, such as extreme weather, occlusions from large vehicles, and complex traffic dynamics, often undermine the accuracy of the detection. Future work should explore the development of more robust perception algorithms and noise-resilient techniques to address these challenges. Furthermore, simulation frameworks are needed to incorporate these complexities more realistically, enabling rigorous validation of cooperative perception systems in controlled but representative settings.

Another promising avenue lies in improving the compression and feature sharing methodologies. While this dissertation has shown the utility of approaches such as clipping, quantization, and H.265 compression, future research could focus on adaptive techniques that respond dynamically to varying network conditions and operational demands. For instance, entropy or loss-aware encoding schemes could improve the balance between bandwidth usage and detection performance, particularly for smaller object classes that are more sensitive to data degradation. Such advancements are critical to ensuring reliable system performance, even in bandwidth constrained environments.

Emerging communication technologies provide significant opportunities to advance the scalability and reliability of cooperative perception systems. The adoption of 6G networks, satellite-based communication, and other next-generation technologies could enable ultra-reliable low-latency communication (URLLC) and expand the operational scope of these systems to underserved and remote areas. Future research should investigate how these technologies can overcome existing challenges related to network variability and scalability, improving the robustness of cooperative perception in diverse settings.

Beyond technical enhancements, there is significant potential for cooperative perception systems to address broader societal challenges. One such application is predictive maintenance, where shared perception data could be utilized to anticipate and mitigate vehicle or infrastructure problems, minimizing downtime and maintenance costs. Similarly, integrating cooperative perception into smart city initiatives could improve urban mobility, with applications such as pedestrian safety, autonomous public transport, and dynamic traffic flow management. These developments would play a pivotal role in creating safer, more efficient, and more adaptive transportation systems.

Finally, the widespread adoption of cooperative perception systems will require standardized protocols for V2X communication. Establishing global interoperability standards is essential to ensure seamless integration across various platforms, manufacturers, and jurisdictions. Such efforts will also foster collaboration among stakeholders, accelerating the deployment of these systems on a scale. Future research should focus on developing and refining these standards to support the growth and widespread acceptance of cooperative perception technologies. Although this research represents significant progress, several challenges and opportunities remain for future exploration. These include advancing technology, refining methodologies, and expanding applications.

10.3 Closing Reflections

This dissertation has highlighted the significant role of cooperative perception in shaping the future of autonomous driving systems. Through the integration of hybrid computing frameworks, advanced simulation tools, and extensive real-world evaluations, this work bridges the divide between conceptual research and practical deployment. The findings emphasize the necessity of interdisciplinary collaboration, addressing technical challenges, to develop equitable and sustainable solutions. As we look to the future, cooperative perception represents a transformative approach to intelligent transportation, offering new opportunities to enhance safety, efficiency, and adaptability in mobility networks. Continued innovation and collaborative efforts will be crucial in refining these systems, ensuring they meet the demands of real-world applications while fostering societal acceptance. Contributing to this rapidly evolving field has been an enriching experience, and we remain optimistic about its potential to revolutionize transportation in the years ahead.

References

[1] Shimaa A Abdel Hakeem, Anar A Hady, and HyungWon Kim. "5G-V2X: Standardization, architecture, use cases, network-slicing, and edge-computing". In: *Wireless Networks* 26.8 (2020), pp. 6015–6041. DOI: 10.1007/s11276-020-02419-8.

[2] Ana Aguiar, Afonso Azevedo, and Bernardo Carrilho. "POSTER: Multipath Transport for Video Streams in Heterogeneous Wireless Environments". In: *2023 IEEE 31st International Conference on Network Protocols (ICNP)*. 2023, pp. 1–2. DOI: 10.1109/ICNP59255.2023.10355572.

[3] M Nadeem Ahangar, Qasim Z Ahmed, Fahd A Khan, and Maryam Hafeez. "A survey of autonomous vehicles: Enabling communication technologies and challenges". In: *Sensors* 21.3 (2021), p. 706. DOI: 10.3390/s21030706.

[4] Manzoor Ahmed, Haseeb Mirza, Fang Xu, Wali Ullah Khan, Qing Lin, and Zhu Han. "Vehicular communication network enabled CAV data offloading: A review". In: *IEEE Transactions on Intelligent Transportation Systems* (2023). DOI: 10.1109/TITS.2023.3263643.

[5] Manzoor Ahmed, Salman Raza, Muhammad Ayzed Mirza, Abdul Aziz, Manzoor Ahmed Khan, Wali Ullah Khan, Jianbo Li, and Zhu Han. "A Survey on Vehicular Task Offloading: Classification, Issues, and Challenges". In: *Journal of King Saud University-Computer and Information Sciences* (2022). DOI: 10.1016/j.jksuci.2022.05.016.

[6] Nesma Alabyad, Ziad Hany, Abdelrahman Mostafa, Reem Eldaby, Ibrahim Ayman Tagen, and Amal Mehanna. "From Vision to Precision: The Dynamic Transformation of Object Detection in Autonomous Systems". In: *2024 6th International Conference on Computing and Informatics (ICCI)*. IEEE. 2024, p. 332.

[7] Zoraze Ali, Sandra Lagén, Lorenza Giupponi, and Richard Rouil. "3GPP NR V2X mode 2: overview, models and system-level evaluation". In: *IEEE Access* (2021). DOI: 10.1109/ACCESS.2021.3090855.

[8] Shunsuke Aoki, Takamasa Higuchi, and Onur Altintas. "Cooperative perception with deep reinforcement learning for connected vehicles". In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2020, pp. 328–334. DOI: 10.1109/IV47402.2020.9304570.

[9] Leyre Artal-Villa, Ahmed Hussein, and Cristina Olaverri-Monreal. "Extension of the 3DCoAutoSim to Simulate Vehicle and Pedestrian Interaction based on SUMO and Unity 3D". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2019, pp. 885–890. DOI: 10.1109/IVS.2019.8814253.

[10] Zhengwei Bai, Jacqueline Garrido Escobar, Guoyuan Wu, and Matthew J. Barth. "Object Perception Framework for Connected and Automated Vehicles: A Case Study". In: *2023 IEEE Transportation Electrification Conference and Expo (ITEC)*. 2023, pp. 1–5. DOI: 10.1109/ITEC55900.2023.10186931.

[11] Andrey Belogaev, Alexey Elokhin, Artem Krasilov, Evgeny Khorov, and Ian F Akyildiz. "Cost-effective V2X task offloading in MEC-assisted intelligent transportation systems". In: *IEEE access* 8 (2020). DOI: 10.1109/ACCESS.2020.3023263.

[12] Venkata Bhardwaj. "AI-Enabled Autonomous Driving: Enhancing Safety and Efficiency through Predictive Analytics". In: *International Journal of Scientific Research and Management (IJSRM)* 12.02 (2024), p. 1076.

[13] Lucas Bréhon–Grataloup, Rahim Kacimi, and André-Luc Beylot. "Mobile edge computing for V2X architectures and applications: A survey". In: *Computer Networks* 206 (2022). DOI: 10.1016/j.comnet.2022.108797.

[14] Fabio Busacca, Carla Cirino, Giuseppe Faraci, Christian Grasso, Sergio Palazzo, and Giovanni Schembra. "Multi-Layer Offloading at the Edge for Vehicular Networks". In: *2020 Mediterranean Communication and Computer Networking Conference (MedComNet)*. 2020, pp. 1–8. DOI: 10.1109/MedComNet49392.2020.9191474.

[15] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. "nuScenes: A multimodal dataset for autonomous driving". In: *arXiv* (2019). DOI: 10.48550/arXiv.1903.11027.

[16] Antoine Caillot, Safa Ouerghi, Pascal Vasseur, Rémi Boutteau, and Yohan Dupuis. "Survey on Cooperative Perception in an Automotive Context". In: *IEEE Transactions on Intelligent Transportation Systems* (2022). DOI: 10.1109/TITS.2022.3153815.

[17] Devendra K Chaturvedi. *Modeling and simulation of systems using MATLAB® and Simulink®*. CRC press, 2017.

[18] Qi Chen, Xu Ma, Sihai Tang, Jingda Guo, Qing Yang, and Song Fu. "F-cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds". In: *Proceedings of the 4th ACM/IEEE Symposium on Edge Computing*. 2019, pp. 88–100. DOI: 10.1145/3318216.3363300.

[19] Gong Cheng, Xiang Yuan, Xiwen Yao, Kebing Yan, Qinghua Zeng, Xingxing Xie, and Junwei Han. "Towards large-scale small object detection: Survey and benchmarks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2023).

[20] Hyomin Choi and Ivan V Bajić. "Deep feature compression for collaborative object detection". In: *2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE. 2018, pp. 3743–3747. DOI: 10.1109/ICIP.2018.8451100.

[21] Hyomin Choi and Ivan V Bajić. "Near-lossless deep feature compression for collaborative intelligence". In: *2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE. 2018, pp. 1–6. DOI: 10.1109/MMSP.2018.8547134.

[22] Robert A Cohen, Hyomin Choi, and Ivan V Bajić. "Lightweight compression of neural network feature tensors for collaborative intelligence". In: *2020 IEEE International Conference on Multimedia and Expo*. IEEE. 2020. DOI: 10.1109/ICME46284.2020.9102797.

[23] Multi-access Edge Computing. *Framework and Reference Architecture, ETSI GS MEC 003, Rev. 2.2. 1, Dec. 2020*.

[24] Christopher Crick, Graylin Jay, Sarah Osentoski, Benjamin Pitzer, and Odest Chadwicke Jenkins. "Rosbridge: Ros for non-ros users". In: *Robotics Research*. Springer, 2017.

[25] Jiaxun Cui, Hang Qiu, Dian Chen, Peter Stone, and Yuke Zhu. "Coopernaut: End-to-end driving with cooperative perception for networked vehicles". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 17252–1726. DOI: 10.1109/CVPR52688.2022.01674.

[26] N. Dalal and B. Triggs. "Histograms of oriented gradients for human detection". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. 2005, 886–893 vol. 1. DOI: 10.1109/CVPR.2005.177.

[27] Alisson Barbosa De Souza, Paulo AL Rego, Tiago Carneiro, Jardel Das C Rodrigues, Pedro Pedrosa Reboucas Filho, Jose Neuman De Souza, Vinay Chamola, Victor Hugo C De Albuquerque, and Biplab Sikdar. "Computation offloading for vehicular environments: A survey". In: *IEEE Access* 8 (2020), pp. 198214–198243. DOI: 10.1109/ACCESS.2020.3033828.

[28] Benjamin Deguerre, Clément Chatelain, and Gilles Gasso. "Fast object detection in compressed jpeg images". In: *2019 IEEE intelligent transportation systems conference (ITSC)*. IEEE. 2019, pp. 333–338. DOI: 10.1109/ITSC.2019.8916937.

[29] P. Deutsch and J.-L. Gailly. "RFC1950: ZLIB Compressed Data Format Specification version 3.3". In: *RFC Editor* (1996). DOI: https://dl.acm.org/doi/10.17487/rfc1950.

[30] Tiago Dias, Emanuel Silva, João Almeida, and Joaquim Ferreira. "An ITS-G5 V2X solution in C-ROADS Portugal". In: *Transportation Research Procedia* (2023).

[31] Samuel Dodge and Lina Karam. "Understanding how image quality affects deep neural networks". In: *2016 Eighth International Conference on Quality of Multimedia Experience (QoMEX)*. 2016. DOI: 10.1109/QoMEX.2016.7498955.

[32] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[33] Max Ehrlich, Larry Davis, Ser-Nam Lim, and Abhinav Shrivastava. "Analyzing and Mitigating JPEG Compression Defects in Deep Learning". In: *2021 International Conference on Computer Vision Workshops (ICCVW)*. IEEE, 2021, pp. 922–930. DOI: 10.1109/ICCVW54120.2021.00267.

[34] TR ETSI. "103 562 V2. 1.1 (2019-12) Intelligent Transport Systems (ITS)". In: *Vehicular Communications* (2019).

[35] Madiha Farasat, Dushmantha N Thalakotuna, Zhonghao Hu, and Yang Yang. "A review on 5G sub-6 GHz base station antenna design challenges". In: *Electronics* 10.4 (2021), p. 405. DOI: 10.3390/electronics10040405.

[36] Weiyang Feng, Siyu Lin, Ning Zhang, Gongpu Wang, Bo Ai, and Lin Cai. "C-V2X based Offloading Strategy in Multi-Tier Vehicular Edge Computing System". In: *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*.

[37] Abderrahime Filali, Amine Abouaomar, Soumaya Cherkaoui, Abdellatif Kobbane, and Mohsen Guizani. "Multi-access edge computing: A survey". In: *IEEE Access* 8 (2020). DOI: 10.1109/ACCESS.2020.3034136.

[38] Bryse Flowers, Yu-Jen Ku, Sabur Baidya, and Sujit Dey. "Utilizing Reinforcement Learning for Adaptive Sensor Data Sharing over C-V2X Communications". In: *IEEE Transactions on Vehicular Technology* (2023).

[39] Raphaël Frank and Faisal Hawlader. "Poster: Commercial 5G Performance: A V2X Experiment". In: *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE. 2021, pp. 129–130. DOI: 10.1109/VNC52810.2021.9644666.

[40] Mario H. Castañeda Garcia, Alejandro Molina-Galan, Mate Boban, Javier Gozalvez, Baldomero Coll-Perales, Taylan Şahin, and Apostolos Kousaridas. "A Tutorial on 5G NR V2X Communications". In: *IEEE Communications Surveys & Tutorials* 23.3 (2021), pp. 1972–2026. DOI: 10.1109/COMST.2021.3057017.

[41] Keno Garlichs, Hendrik-Jörn Günther, and Lars C. Wolf. "Generation Rules for the Collective Perception Service". In: *2019 IEEE Vehicular Networking Conference (VNC)*. 2019, pp. 1–8. DOI: 10.1109/VNC48660.2019.9062827.

[42] Keno Garlichs, Martin Wegner, and Lars C Wolf. "Realizing collective perception in the artery simulation framework". In: *2018 IEEE Vehicular Networking Conference (VNC)*. IEEE. 2018, pp. 1–4.

[43] James H Gawron, Gregory A Keoleian, Robert D De Kleine, Timothy J Wallington, and Hyung Chul Kim. "Life cycle assessment of connected and automated vehicles: sensing and computing subsystem and vehicle level effects". In: *Environmental science & technology* 52.5 (2018), p. 3249.

[44] Georgios Georgiadis. "Accelerating convolutional neural networks via activation map compression". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 7085–7095. DOI: 10.1109/CVPR.2019.00725.

[45] Ross Girshick. "Fast R-CNN". In: *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015. DOI: 10.1109/ICCV.2015.169.

[46] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. 2014. DOI: 10.1109/CVPR.2014.81.

[47] Daniel Grimm, Marc Schindewolf, David Kraus, and Eric Sax. "Co-simulate no more: The CARLA V2X Sensor". In: *2024 IEEE Intelligent Vehicles Symposium (IV)*. IEEE. 2024, pp. 2429–2436.

[48] Peter Große, Carsten Andrich, Wim Kotterman, Alexander Ihlow, and Giovanni Del Galdo. "Measuring ETSI ITS-G5 Communications Latencies with Commercial off-the-shelf Wi-Fi Hardware". In: *2018 IEEE MTT-S International Conference on Microwaves for Intelligent Mobility (ICMIM)*. 2018, pp. 1–4. DOI: 10.1109/ICMIM.2018.8443355.

[49] Hendrik-jorn Gunther, Oliver Trauer, and Lars Wolf. "The potential of collective perception in vehicular ad-hoc networks". In: *2015 14th International Conference on ITS Telecommunications (ITST)*. IEEE. 2015, pp. 1–5.

[50] Juan Gutierrez-Aguado, Raúl Peña-Ortiz, Miguel García-Pineda, and Jose M Claver. "Cloud-based elastic architecture for distributed video encoding: Evaluating H. 265, VP9, and AV1". In: *Journal of Network and Computer Applications* 171 (2020), p. 102782.

[51] Sohan Gyawali, Shengjie Xu, Yi Qian, and Rose Qingyang Hu. "Challenges and solutions for cellular based V2X communications". In: *IEEE Communications Surveys & Tutorials* 23.1 (2020), pp. 222–255.

[52] Tobias Hardes, Ion Turcanu, and Christoph Sommer. "Poster: A case for heterogenous co-simulation of cooperative and autonomous driving". In: *2023 IEEE Vehicular Networking Conference (VNC)*. IEEE. 2023, pp. 151–152.

[53] Faisal Hawlader and Raphael Frank. "Realistic Cooperative Perception for Connected and Automated Vehicles: A Simulation Review". In: *2023 8th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. IEEE, 2023. DOI: 10.1109/MT-ITS56129.2023.10241653.

[54] Faisal Hawlader and Raphael Frank. "The Ugly Truth of Realistic Perception in Vehicular Simulations". In: *2023 IEEE 20th Consumer Communications & Networking Conference (CCNC)*. IEEE, 2023. DOI: 10.1109/CCNC51644.2023.10060697.

[55] Faisal Hawlader and Raphael Frank. "Towards a Framework to Evaluate Cooperative Perception for Connected Vehicles". In: *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2021. DOI: 10.1109/VNC52810.2021.9644667.

[56] Faisal Hawlader, François Robinet, and Raphaël Frank. "Cooperative Perception Using V2X Communications: An Experimental Study". In: *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*. IEEE. 2024. DOI: 10.1109/VTC2024-Fall63153.2024.10757448.

[57] Faisal Hawlader, François Robinet, and Raphaël Frank. "Leveraging the edge and cloud for V2X-based real-time object detection in autonomous driving". In: *Computer Communications*. Elsevier, 2024. DOI: 10.1016/j.comcom.2023.11.025.

[58] Faisal Hawlader, François Robinet, and Raphaël Frank. "Lightweight Features Sharing for Real-Time Object Detection in Cooperative Driving". In: *2023 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2023. DOI: 10.1109/VNC57357.2023.10136339.

[59] Faisal Hawlader, François Robinet, and Raphaël Frank. "Vehicle-to-Infrastructure Communication for Real-Time Object Detection in Autonomous Driving". In: *18th Wireless On-Demand Network Systems and Services Conference (WONS)*. IEEE, 2023. DOI: 10.23919/WONS57325.2023.10061953.

[60] Faisal Hawlader, François Robinet, Elghazaly Gamal, and Raphaël Frank. "Cloud-Assisted 360-Degree 3D Perception for Autonomous Vehicles Using V2X Communication and Hybrid Computing". In: *20th Wireless On-demand Network systems and Services Conference (WONS)*. IEEE. 2025.

[61] Jianhua He, Zuoyin Tang, Xiaoming Fu, Supeng Leng, Fan Wu, Kaisheng Huang, Jianye Huang, Jie Zhang, Yan Zhang, Andrew Radford, et al. "Cooperative connected autonomous vehicles (CAV): research, applications and challenges". In: *IEEE 27th International Conference on Network Protocols (ICNP)*. IEEE. 2019.

[62] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

[63] Anupama Hegde and Andreas Festag. "Artery-C: An OMNeT++ Based Discrete Event Simulation Framework for Cellular V2X". In: *Proceedings of the 23rd International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. 2020, pp. 47–51.

[64] Jeremy Horwitz. "The definitive guide to 5G low, mid, and high band speeds". In: *Venture Beat Online Magazine* (2019).

[65] Chung-Ming Huang and Chi-Feng Lai. "The Mobile Edge Computing (MEC)-based Vehicle to Infrastructure (V2I) Data Offloading from Cellular Network to VANET using the Delay-Constrained Computing Scheme". In: *2020 International Computer Symposium (ICS)*. IEEE. 2020.

[66] Hui Huang, Wenqi Fang, and Huiyun Li. "Performance Modelling of V2V based Collective Perceptions in Connected and Autonomous Vehicles". In: *2019 IEEE 44th Conference on Local Computer Networks (LCN)*. 2019, pp. 356–363. DOI: 10.1109/LCN44214.2019.8990854.

[67] Hui Huang, Huiyun Li, Cuiping Shao, Tianfu Sun, Wenqi Fang, and Shaobo Dang. "Data redundancy mitigation in V2X based collective perceptions". In: *IEEE Access* 8 (2020), pp. 13405–13418.

[68] Tao Huang, Jianan Liu, Xi Zhou, Dinh C Nguyen, Mostafa Rahimi Azghadi, Yuxuan Xia, Qing-Long Han, and Sumei Sun. "V2X cooperative perception for autonomous driving: Recent advances and challenges". In: *preprint arXiv:2310.03525* (2023).

[69] Qingxiong Huangyuan, Li Song, Zhengyi Luo, Xiangwen Wang, and Yanan Zhao. "Performance evaluation of H. 265/MPEG-HEVC encoders for 4K video sequences". In: *Signal and Information Processing Association Annual Summit and Conference (APSIPA), 2014 Asia-Pacific*. IEEE. 2014, pp. 1–8.

[70] Robert Hult, Gabriel R Campos, Erik Steinmetz, Lars Hammarstrand, Paolo Falcone, and Henk Wymeersch. "Coordination of cooperative autonomous vehicles: Toward safer and more efficient road transportation". In: *IEEE Signal Processing Magazine* 33.6 (2016), pp. 74–84.

[71] Seungeon Hwang and Jongsun Park. "Percentile Clipping based Low Bit-Precision Quantization for Depth Estimation Network". In: *2022 19th International SoC Design Conference (ISOCC)*. IEEE. 2022, pp. 73–74.

[72] Hafiz Hasnain Imtiaz and Suhua Tang. "Multi-task partial offloading with relay and adaptive bandwidth allocation for the mec-assisted iot". In: *Sensors* (2022).

[73] Akhirul Islam, Arindam Debnath, Manojit Ghose, and Suchetana Chakraborty. "A survey on task offloading in multi-access edge computing". In: *Journal of Systems Architecture* 118 (2021).

[74] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. *Ultralytics YOLOv8*. Version 8.0.0. 2023.

[75] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Zeng Borovec, Diego Montes, et al. "Ultralytics/YOLOv5: v6. 2-yolov5 classification models, apple m1, reproducibility, clearml and deci. ai integrations". In: *Zenodo* (2022). DOI: `https://zenodo.org/records/5563715`.

[76] Glenn Jocher and Stoken. "ultralytics/YOLOv5n'Nano'models, Roboflow integration, TensorFlow export, OpenCV DNN support". In: *Zenodo* (2021). DOI: `10.5281/zenodo.5563715`.

[77] Mouna Karoui, Antonio Freitas, and Gerard Chalhoub. "Performance comparison between LTE-V2X and ITS-G5 under realistic urban scenarios". In: *2020 IEEE 91st vehicular technology conference (VTC2020-Spring)*. IEEE.

[78] Kishwer Abdul Khaliq, Omer Chughtai, Abdullah Shahwani, Amir Qayyum, and Jürgen Pannek. "Road accidents detection, data collection and data analysis using V2X communication and edge/cloud computing". In: *Electronics* (2019).

[79] Muhammad Asif Khan, Emna Baccour, Zina Chkirbene, Aiman Erbad, Ridha Hamila, Mounir Hamdi, and Moncef Gabbouj. "A Survey on Mobile Edge Computing for Video Streaming: Opportunities and Challenges". In: *IEEE Access* (2022).

[80] Seong-Woo Kim, Baoxing Qin, Zhuang Jie Chong, Xiaotong Shen, Wei Liu, Marcelo H. Ang, Emilio Frazzoli, and Daniela Rus. "Multivehicle Cooperative Driving Using Cooperative Perception: Design and Experimental Validation". In: *IEEE Transactions on Intelligent Transportation Systems* 16.2 (2015), pp. 663–680. DOI: `10.1109/TITS.2014.2337316`.

[81] Gergely Attila Kovács and László Bokor. "Integrating Artery and Simu5G: A Mobile Edge Computing use case for Collective Perception-based V2X safety applications". In: *2022 45th International Conference on Telecommunications and Signal Processing (TSP)*. IEEE. 2022, pp. 360–366.

[82] Vipin Kumar Kukkala, Jordan Tunnell, Sudeep Pasricha, and Thomas Bradley. "Advanced driver-assistance systems: A path toward autonomous vehicles". In: *IEEE Consumer Electronics Magazine* 7.5 (2018), pp. 18–25.

[83] Jong-Seok Lee and Touradj Ebrahimi. "Perceptual video compression: A survey". In: *IEEE Journal of selected topics in signal processing* 6.6 (2012).

[84] Ji Li, Hui Gao, Tiejun Lv, and Yueming Lu. "Deep reinforcement learning based computation offloading and resource allocation for MEC". In: *2018 IEEE Wireless communications and networking conference (WCNC)*. IEEE. 2018.

[85] Lingling Li, Jiuchun Ren, and Qian Zhu. "On the application of LoRa LPWAN technology in Sailing Monitoring System". In: *2017 13th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*. 2017, pp. 77–80. DOI: `10.1109/WONS.2017.7888762`.

[86] Zhiqi Li, Enze Wang, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. "BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers". In: *arXiv preprint arXiv:2203.17270* (2022).

[87] Li Lin, Xiaofei Liao, Hai Jin, and Peng Li. "Computation Offloading Toward Edge Computing". In: *Proceedings of the IEEE* 107.8 (2019), pp. 1584–1607. DOI: 10.1109/JPROC.2019.2922285.

[88] Tsung-Yi Lin, Pietro Maire, and Ramanan. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014.

[89] Cui Liqun and Hu Lei. "Clipping-based neural network post training quantization for object detection". In: *IEEE International Conference on Control, Electronics and Computer Technology (ICCECT)*. IEEE. 2023.

[90] Cheng Liu. "Enhance the 3D Object Detection With 2D Prior". In: *IEEE Access* 12 (2024), pp. 67161–67169. DOI: 10.1109/ACCESS.2024.3398373.

[91] Chenguang Liu and Shuang-Hua. "Cooperative Perception With Learning-Based V2V Communications". In: *IEEE Wireless Communications Letters* (2023). DOI: 10.1109/LWC.2023.3295612.

[92] Jianhui Liu and Qi Zhang. "To improve service reliability for AI-powered time-critical services using imperfect transmission in MEC: An experimental study". In: *IEEE Internet of Things Journal* 7.10 (2020), pp. 9357–9371.

[93] Liangkai Liu and Weisong Shi. "Computing systems for autonomous driving: State of the art and challenges". In: *IEEE Internet of Things Journal* (2020).

[94] Luyang Liu, Hongyu Li, and Marco Gruteser. "Edge assisted real-time object detection for mobile augmented reality". In: *The 25th annual international conference on mobile computing and networking*. 2019, pp. 1–16.

[95] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. "SSD: Single Shot MultiBox Detector". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling. Cham: Springer International Publishing, 2016, pp. 21–37. ISBN: 978-3-319-46448-0.

[96] Wei Liu, Seong-Woo Kim, Zhuang Jie Chong, XT Shen, and Marcelo H Ang. "Motion planning using cooperative perception on urban road". In: *2013 6th IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. IEEE. 2013, pp. 130–137.

[97] Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. "Post-training quantization for vision transformer". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 28092–28103.

[98] Ignacio Llatser, Thomas Michalke, Maxim Dolgov, Florian Wildschütte, and Hendrik Fuchs. "Cooperative automated driving use cases for 5G V2X communication". In: *2019 IEEE 2nd 5G World Forum (5GWF)*. IEEE. 2019, pp. 120–125.

[99] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. "Microscopic Traffic Simulation using SUMO". In: *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018.

[100] Xianghe Ma. "High-resolution image compression algorithms in remote sensing imaging". In: *Displays* (2023), p. 102462.

[101] Vasilis Maglogiannis, Dries Naudts, Seilendria Hadiwardoyo, Daniel van den Akker, Johann Marquez-Barja, and Ingrid Moerman. "Experimental V2X Evaluation for C-V2X and ITS-G5 Technologies in a Real-Life Highway Environment". In: *IEEE Transactions on Network and Service Management* 19.2 (2022). DOI: 10.1109/TNSM.2021.3129348.

[102] Enrique Marti, Miguel Angel De Miguel, Fernando Garcia, and Joshue Perez. "A review of sensor technologies for perception in automated driving". In: *IEEE Intelligent Transportation Systems Magazine* 11.4 (2019), pp. 94–108.

[103] Amir Emad Marvasti and Yaser P Fallah. "Bandwidth-adaptive feature sharing for cooperative lidar object detection". In: *2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS)*. IEEE.

[104] Ehsan Emad Marvasti, Arash Raftari, Amir Emad Marvasti, Yaser P Fallah, Rui Guo, and Hongsheng Lu. "Feature Sharing and Integration for Cooperative Cognition and Perception with Volumetric Sensors". In: *preprint arXiv:2011.08317* (2020).

[105] Yoshitomo Matsubara and Marco Levorato. "Neural compression and filtering for edge-assisted real-time object detection in challenged networks". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021.

[106] Yoshitomo Matsubara, Marco Levorato, and Francesco Restuccia. "Split computing and early exiting for deep learning applications: Survey and research challenges". In: *ACM Computing Surveys* 55.5 (2022), pp. 1–30. DOI: 10.1145/3527155.

[107] Brian McCarthy and Aisling O'Driscoll. "OpenCV2X Mode 4: A Simulation Extension for Cellular Vehicular Communication Networks". In: *2019 IEEE 24th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. 2019, pp. 1–6. DOI: 10.1109/CAMAD.2019.8858436.

[108] Rishabh Mehta and Rajeev Shorey. "Deepsplit: Dynamic splitting of collaborative edge-cloud convolutional neural networks". In: *International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE. 2020.

[109] Arvind Merwaday, Satish C Jha, Kathiravetpillai Sivanesan, Ignacio J Alvarez, Leonardo Gomes Baltar, Vesh Raj Sharma Banjade, and Suman A Sehra. "Infrastructure Assisted Efficient Collective Perception Service for Connected Vehicles". In: *2021 IEEE Vehicular Networking Conference (VNC)*. IEEE. 2021, pp. 119–120.

[110] Domen Mongus and Borut Žalik. "Efficient method for lossless LIDAR data compression". In: *International journal of remote sensing* 32 (2011).

[111] Jose Eugenio Naranjo, Felipe Jimenez, Jose Javier Anaya, Rodrigo Castiñeira, Mauro Gil, and David Romero. "Validation Experiences on Autonomous and Connected Driving in AUTOCITS Pilot in Madrid". In: *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*. 2018, pp. 1–4. DOI: 10.1109/VTCSpring.2018.8417888.

[112] Arvind Narayanan, Eman Ramadan, Jason Carpenter, Qingxu Liu, Y. Liu, Feng Qian, and Zhi-Li Zhang. "A First Look at Commercial 5G Performance on Smartphones". In: *Proceedings of The Web Conference 2020* (2020).

[113] Giovanni Nardini, Dario Sabella, Giovanni Stea, Purvi Thakkar, and Antonio Virdis. "Simu5G–An OMNeT++ Library for End-to-End Performance Evaluation of 5G Networks". In: *IEEE Access* (2020). DOI: 10.1109/ACCESS.2020.3028550.

[114] Giovanni Nardini, Giovanni Stea, Antonio Virdis, and Dario Sabella. "Simu5G: A System-level Simulator for 5G Networks." In: *Simultech*. 2020, pp. 68–80.

[115] Anselme Ndikumana, Kim Khoa Nguyen, and Mohamed Cheriet. "Age of processing-based data offloading for autonomous vehicles in MultiRATs Open RAN". In: *IEEE Transactions on Intelligent Transportation Systems* (2022). DOI: 10.1109/TITS.2022.3192098.

[116] DR Niranjan, BC VinayKarthik, et al. "Deep Learning based Object Detection Model for Autonomous Driving Research using CARLA Simulator". In: *2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*. IEEE. 2021, pp. 1251–1258.

[117] Zhaoqing Pan, Jianjun Lei, Yun Zhang, Xingming Sun, and Sam Kwong. "Fast motion estimation based on content property for low-complexity H. 265/HEVC encoder". In: *IEEE Transactions on Broadcasting* 62.3 (2016), pp. 675–684.

[118] Trung Pham, Mehran Maghoumi, Wanli Jiang, Bala Siva Sashank Jujjavarapu, Mehdi Sajjadi, Xin Liu, Hsuan-Chu Lin, Bor-Jeng Chen, Giang Truong, Chao Fang, Junghyun Kwon, and Minwoo Park. "NVAutoNet: Fast and Accurate 360° 3D Visual Perception For Self Driving". In: *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. 2024, pp. 7361–7370. DOI: 10.1109/WACV57701.2024.00721.

[119] Christoph Pilz, Andrea Ulbel, and Gerald Steinbauer-Wagner. "The Components of Cooperative Perception-a Proposal for Future Works". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2021, pp. 7–14.

[120] Theodore S Rappaport, Yunchou Xing, George R MacCartney, Andreas F Molisch, Evangelos Mellios, and Jianhua Zhang. "Overview of millimeter wave communications for fifth-generation (5G) wireless networks—With a focus on propagation models". In: *IEEE Transactions on antennas and propagation* 65.12 (2017).

[121] Andreas Rauch, Felix Klanner, Ralph Rasshofer, and Klaus Dietmayer. "Car2x-based perception in a high-level fusion architecture for cooperative perception systems". In: *2012 IEEE Intelligent Vehicles Symposium*. IEEE. 2012, pp. 270–275.

[122] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You Only Look Once: Unified, Real-Time Object Detection". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[123] Joseph Redmon and Ali Farhadi. "Yolov3: An incremental improvement". In: *arXiv preprint arXiv:1804.02767* (2018).

[124] Ju Ren, Yundi Guo, Deyu Zhang, Qingqing Liu, and Yaoxue Zhang. "Distributed and efficient object detection in edge computing: Challenges and solutions". In: *IEEE Network* 32.6 (2018).

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

[126] Martin Řeřábek and Touradj Ebrahimi. "Comparison of compression efficiency between HEVC/H. 265 and VP9 based on subjective assessments". In: *Applications of digital image processing XXXVII*. SPIE. 2014.

[127] Raphael Riebl, Hendrik-Jörn Günther, Christian Facchi, and Lars Wolf. "Artery: Extending veins for VANET applications". In: *2015 International Conference on Models and Technologies for Intelligent Transportation Systems*. IEEE. 2015.

[128] Guodong Rong, Byung Hyun Shin, Hadi Tabatabaee, Qiang Lu, Steve Lemke, Mārtiņš Možeiko, Eric Boise, Geehoon Uhm, Mark Gerow, Shalin Mehta, Eugene Agafonov, Tae Hyung Kim, Eric Sterner, Keunhae Ushiroda, Michael Reyes, Dmitry Zelenkovsky, and Seonman Kim. "LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving". In: *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. 2020, pp. 1–6. DOI: 10.1109/ITSC45102.2020.9294422.

[129] Ankur Sarker, Haiying Shen, Mizanur Rahman, Mashrur Chowdhury, Kakan Dey, Fangjian Li, Yue Wang, and Husnu S Narman. "A review of sensing and communication, human factors, and controller aspects for information-aware connected and automated vehicles". In: *IEEE transactions on intelligent transportation systems* 21.1 (2019), pp. 7–29.

[130] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. "AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles". In: *CoRR* (2017).

[131] Mao Shan, Karan Narula, Yung Fei Wong, Stewart Worrall, Malik Khan, Paul Alexander, and Eduardo Nebot. "Demonstrations of cooperative perception: safety and robustness in connected and automated vehicle operations". In: *Sensors* 21.1 (2021), p. 200.

[132] Joshua E Siegel, Dylan C Erb, and Sanjay E Sarma. "A survey of the connected vehicle landscape—Architectures, enabling technologies, applications, and development areas". In: *IEEE Transactions on Intelligent Transportation Systems* 19.8 (2017), pp. 2391–2406.

[133] Yushan Siriwardhana, Pawani Porambage, Madhusanka Liyanage, and Mika Yliant-tila. "A survey on mobile augmented reality with 5G mobile edge computing: architectures, applications, and technical aspects". In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 1160–1192. DOI: 10.1109/COMST.2021.3081960.

[134] A. Skodras, C. Christopoulos, and T. Ebrahimi. "The JPEG 2000 still image compression standard". In: *IEEE Signal Processing Magazine* 18.5 (2001), pp. 36–58. DOI: 10.1109/79.952804.

[135] Christoph Sommer, Reinhard German, and Falko Dressler. "Bidirectionally coupled network and road traffic simulation for improved IVC analysis". In: *IEEE Transactions on mobile computing* 10.1 (2010), pp. 3–15.

[136] Christoph Sommer, Stefan Joerer, and Falko Dressler. "On the applicability of Two-Ray path loss models for vehicular network simulation". In: *2012 IEEE Vehicular Networking Conference (VNC)*. 2012, pp. 64–69. DOI: 10.1109/VNC.2012.6407446.

[137] Sedat Sonko, Emmanuel Augustine Etukudoh, Kenneth Ifeanyi Ibekwe, Valentine Ikenna Ilojianya, and Cosmas Dominic Daudu. "A comprehensive review of embedded systems in autonomous vehicles: Trends, challenges, and future directions". In: *World Journal of Advanced Research and Reviews* 21.1 (2024), pp. 2009–2020.

[138] Jochen Stellwagen, Matthias Deegener, and Michael Kuhn. "Application-Dependent Support of ITS-G5 Car-to-Car Communication by Selected Usage of LTE-V2X Resources". In: *Automotive meets Electronics*. 2021.

[139] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. "Overview of the high efficiency video coding (HEVC) standard". In: *IEEE Transactions on circuits and systems for video technology* (2012).

[140] Muhammad Naeem Tahir and Marcos Katz. "Heterogeneous (ITS-G5 and 5G) vehicular pilot road weather service platform in a realistic operational environment". In: *Sensors* 21.5 (2021), p. 1676.

[141] Daniel Teixeira, Rui Meireles, and Ana Aguiar. "Wi-Fi throughput estimation and forecasting for vehicle-to-infrastructure communication". In: *Computer Communications* 214 (2024), pp. 223–233.

[142] Mehdi Testouri, Gamal Elghazaly, and Raphael Frank. "RoboCar: A Rapidly Deployable Open-Source Platform for Autonomous Driving Research". In: *arXiv preprint* (2024). DOI: 10.48550/arXiv.2405.03572.

[143] Gokulnath Thandavarayan, Miguel Sepulcre, and Javier Gozalvez. "Generation of Cooperative Perception Messages for Connected and Automated Vehicles". In: *IEEE Transactions on Vehicular Technology* 69.12 (2020). DOI: 10.1109/TVT.2020.3036165.

[144] Gokulnath Thandavarayan, Miguel Sepulcre, and Javier Gozalvez. "Redundancy mitigation in cooperative perception for connected and automated vehicles". In: *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*. IEEE. 2020, pp. 1–5.

[145] Lisa Torrey and Jude Shavlik. "Transfer learning". In: *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.

[146] ETSI ETSI TS. "103 324 V2. 1.1; Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Collective Perception Service". In: *ETSI: Sophia Antipolis, France* (2023).

[147] Shiao-Li Charles Tsao. "Enhanced GTP: an efficient packet tunneling protocol for General Packet Radio Service". In: *ICC 2001. IEEE International Conference on Communications. Conference Record (Cat. No. 01CH37240)*. Vol. 9. IEEE. 2001, pp. 2819–2823.

[148] Miroslav Uhrina, Juraj Bienik, and Martin Vaculik. "Coding efficiency of hevc/h. 265 and vp9 compression standards for high resolutions". In: *2016 26th International Conference Radioelektronika (RADIOELEKTRONIKA)*. IEEE. 2016, pp. 419–423.

[149] Jessica Van Brummelen, Marie O'Brien, Dominique Gruyer, and Homayoun Najjaran. "Autonomous vehicle perception: The technology of today and tomorrow". In: *Transportation research part C: emerging technologies* (2018).

[150] Han Vanholder. "Efficient inference with tensorrt". In: *GPU Technology Conference*. Vol. 1. 2016, p. 2.

[151] Andras Varga. "OMNeT++". In: *Modeling and tools for network simulation* (2010). DOI: 10.1007/978-3-642-12331-3.

[152] András Varga and Rudolf Hornig. "An overview of the OMNeT++ simulation environment". In: *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. 2008, pp. 1–10.

[153] András Varga and Rudolf Hornig. "An overview of the OMNeT++ simulation environment". In: *1st International ICST Conference on Simulation Tools and Techniques for Communications, Networks and Systems*. 2010.

[154] Jorge Vargas, Suleiman Alsweiss, Onur Toker, Rahul Razdan, and Joshua Santos. "An overview of autonomous vehicles sensors and their vulnerability to weather conditions". In: *Sensors* 21.16 (2021), p. 5397.

[155] Georgios Varisteas and Raphaël Frank. "Poster: JUNIOR, A Research Platform for Connected and Automated Driving". In: *2019 IEEE Vehicular Networking Conference (VNC)*. 2019, pp. 1–2. DOI: 10.1109/VNC48660.2019.9062824.

[156] S. Varrette, H. Cartiaux, S. Peter, E. Kieffer, T. Valette, and A. Olloh. "Management of an Academic HPC & Research Computing Facility: The ULHPC Experience 2.0". In: *Proc. of the 6th ACM High Performance Computing and Cluster Technologies Conf. (HPCCT 2022)*. Association for Computing Machinery (ACM), 2022.

[157] P. Viola and M. Jones. "Rapid object detection using a boosted cascade of simple features". In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*. 2001. DOI: 10.1109/CVPR.2001.990517.

[158] Gregory K Wallace. "The JPEG still picture compression standard". In: *IEEE transactions on consumer electronics* 38.1 (1992). DOI: 10.1109/30.125072.

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

[160] Zeyu Wang, Dingwen Li, Chenxu Luo, Cihang Xie, and Xiaodong Yang. "Distillbev: Boosting multi-camera 3d object detection with cross-modal knowledge distillation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 8637–8646.

[161] Jared Winick and Sugih Jamin. *Inet-3.0: Internet topology generator*. Tech. rep. Technical Report CSE-TR-456-02, University of Michigan, 2002.

[162] Hao Xu, Shuaicheng Liu, Guangfu Wang, Guanghui Liu, and Bing Zeng. "Omnet: Learning overlapping mask for partial-to-partial point cloud registration". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021.

[163] Runsheng Xu, Yi Guo, Xu Han, Xin Xia, Hao Xiang, and Jiaqi Ma. "OpenCDA: an open cooperative driving automation framework integrated with co-simulation". In: *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE. 2021.

[164] Chenyu Yang, Yuntao Chen, Haofei Tian, Gao Huang, Hongyang Li, Y. Qiao, Lewei Lu, Jie Zhou, and Jifeng Dai. "BEVFormer v2: Adapting Modern Image Backbones to Bird's-Eye-View Recognition via Perspective Supervision". In: *ArXiv* (2022).

[165] Abid Yaqoob, Ting Bi, and Gabriel-Miro Muntean. "A survey on adaptive 360 video streaming: Solutions, challenges and opportunities". In: *IEEE Communications Surveys & Tutorials* 22.4 (2020), pp. 2801–2838.

[166] Egon Ye, Philip Spiegel, and Matthias Althoff. "Cooperative raw sensor data fusion for ground truth generation in autonomous driving". In: *IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE.

[167] Zhang Yi, Shen Yongliang, and Zhang Jun. "An improved tiny-yolov3 pedestrian detection algorithm". In: *Optik* 183 (2019), pp. 17–23.

[168] DoHyun Daniel Yoon, GG Md Nawaz Ali, and Beshah Ayalew. "Data association and fusion framework for decentralized multi-vehicle cooperative perception". In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 59216. American Society of Mechanical Engineers. 2019, V003T01A019.

[169] Guizhen Yu, Han Li, Yunpeng Wang, Peng Chen, and Bin Zhou. "A review on cooperative perception and control supported infrastructure-vehicle system". In: *Green Energy and Intelligent Transportation* (2022), p. 100023.

[170] Ruozhou Yu, Dejun Yang, and Hao Zhang. "Edge-Assisted Collaborative Perception in Autonomous Driving: A Reflection on Communication Design". In: *2021 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE. 2021.

[171] Syed Adnan Yusuf, Arshad Khan, and Riad Souissi. "Vehicle-to-everything (V2X) in the autonomous vehicles domain–A technical review of communication, sensor, and AI technologies for road user safety". In: *Transportation Research Interdisciplinary Perspectives* 23 (2024), p. 100980.

[172] Wenyi Zhao, Shiyuan Liu, Xinyi Li, Xi Han, and Huihua Yang. "Fast and accurate wheat grain quality detection based on improved YOLOv5". In: *Computers and Electronics in Agriculture* 202 (2022), p. 107426.

[173] Haibo Zhou, Wenchao Xu, Jiacheng Chen, and Wei Wang. "Evolutionary V2X Technologies Toward the Internet of Vehicles: Challenges and Opportunities". In: *Proceedings of the IEEE* 108.2 (2020), pp. 308–323. DOI: 10.1109/JPROC.2019.2961937.

[174] Guangxu Zhu, Zhonghao Lyu, Xiang Jiao, Peixi Liu, Mingzhe Chen, Jie Xu, Shuguang Cui, and Ping Zhang. "Pushing AI to wireless network edge: An overview on integrated sensing, communication, and computation towards 6G". In: *Science China Information Sciences* 66.3 (2023), p. 130301.

[175] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. "Object detection in 20 years: A survey". In: *Proceedings of the IEEE* 111.3 (2023).