electronics

Article

Vehicle-to-Everything-Car Edge Cloud Management with Development, Security, and Operations Automation Framework

DongHwan Ku [1], Hannie Zang [2], Anvarjon Yusupov [1], Sun Park [1] and JongWon Kim [1,*]

[1] AI Graduate School, Gwangju Institute of Science and Technology (GIST), Gwangju 61005, Republic of Korea; donghwan9@gm.gist.ac.kr (D.K.); ayusupov@gm.gist.ac.kr (A.Y.); sunpark@gist.ac.kr (S.P.)
[2] Cloud Computing SW Research Section, Electronics and Telecommunications Research Institute (ETRI), Daejeon 34129, Republic of Korea; hanizang@etri.re.kr
* Correspondence: jongwon@gist.ac.kr; Tel.: +82-62-715-2219

Abstract: Modern autonomous driving and intelligent transportation systems face critical challenges in managing real-time data processing, network latency, and security threats across distributed vehicular environments. Conventional cloud-centric architectures typically struggle to meet the low-latency and high-reliability requirements of vehicle-to-everything (V2X) applications, particularly in dynamic and resource-constrained edge environments. To address these challenges, this study introduces the V2X-Car Edge Cloud system, which is a cloud-native architecture driven by DevSecOps principles to ensure secure deployment, dynamic resource orchestration, and real-time monitoring across distributed edge nodes. The proposed system integrates multicluster orchestration with Kubernetes, hybrid communication protocols (C-V2X, 5G, and WAVE), and data-fusion pipelines to enhance transparency in artificial intelligence (AI)-driven decision making. A software-in-the-loop simulation environment was implemented to validate AI models, and the SmartX MultiSec framework was integrated into the proposed system to dynamically monitor network traffic flow and security. Experimental evaluations in a virtual driving environment demonstrate the ability of the proposed system to perform automated security updates, continuous performance monitoring, and dynamic resource allocation without manual intervention.

Keywords: cloud-native computing; edge computing; DevSecOps automation; V2X service; software-in-the-loop simulation; open-source software

Academic Editor: Wajid Rafique and Sudheer Kumar Battula

Received: 19 December 2024
Revised: 20 January 2025
Accepted: 23 January 2025
Published: 24 January 2025

Citation: Ku, D.; Zang, H.; Yusupov, A.; Park, S.; Kim, J. Vehicle-to-Everything-Car Edge Cloud Management with Development, Security, and Operations Automation Framework. *Electronics* **2025**, *14*, 478. https://doi.org/10.3390/electronics14030478

1. Introduction

In recent years, vehicle-to-everything (V2X) technology has emerged as a cornerstone in the development of autonomous driving and intelligent transportation systems (ITSs). V2X enables real-time communication between vehicles (V2V), infrastructure (V2I), networks (V2N), and pedestrians (V2P), optimizing traffic flow, preventing accidents, and enhancing the safety of autonomous vehicles [1,2]. However, the massive volume of data generated by V2X systems must be processed in real time, a challenge that central cloud infrastructures struggle to meet because of latency and bandwidth limitations [3,4]. To address these challenges, edge computing has attracted significant attention as a solution, where data are processed closer to the source, latency is reduced, and responsiveness is improved, particularly for mission-critical applications such as V2X [5,6].

Despite its advantages, edge computing introduces new complexities, particularly in managing distributed nodes, ensuring real-time data processing, and addressing security vulnerabilities [7]. The management of heterogeneous edge nodes and the orchestration of

artificial intelligence (AI) models for real-time decision making are difficult in such environments. In addition, ensuring the secure deployment and management of software in this decentralized infrastructure poses significant challenges [8]. This is where the integration of development, security, and operations (DevSecOps) comes into play. DevSecOps aims to integrate security practices within continuous integration and continuous deployment (CI/CD) pipelines, ensuring that software is deployed rapidly, securely, and efficiently in dynamic environments such as edge computing [9].

To address the challenges mentioned above, previous studies have attempted to integrate edge computing with V2X systems [10–12]. However, these studies often concentrated on specific functions of edge infrastructure or the connectivity of V2X at the edge, without providing a comprehensive framework for managing and operating the entire system. In response to these limitations, this study proposes the integration of the V2X-Car Edge Cloud with a DevSecOps Automation Framework to optimize the deployment and operation of V2X applications in edge computing environments. The proposed system leverages cloud-native infrastructures to manage multiple Kubernetes (K8S) clusters, thereby enabling scalable and automated orchestration of AI workloads across distributed edge nodes [13]. By employing a DevSecOps approach, the proposed framework not only automates the deployment process but also strengthens security through an integrated CI/CD pipeline, ensuring that V2X applications operate in a secure, scalable, and efficient manner [14].

Furthermore, this study addresses critical challenges in real-time data processing, resource management, and security in edge environments. By combining the V2X-Car Edge Cloud and DevSecOps, the proposed system provides a unified solution for deploying and managing AI-driven V2X services across edge nodes [15]. The contributions of this study are particularly important for advancing the reliability and scalability of edge computing infrastructures in autonomous driving and ITS contexts [16]. The key contributions of this study are as follows:

- Cloud-Native Infrastructure and Autonomous Driving Experimental Environment: A cloud-native architecture leveraging K8S and high-performance hardware is designed to support scalable AI workload orchestration. The proposed framework is validated through a comprehensive autonomous driving experimental environment that integrates software-in-the-loop simulation (SiLS) and hybrid communication technologies to replicate real-world V2X scenarios.
- DevSecOps-Driven Automation and Real-Time Visualization with MultiSec: This study introduces a tailored DevSecOps pipeline for V2X environments, incorporating the SmartX MultiSec framework as a real-time visualization tool for traffic flow and security monitoring. MultiSec provides intuitive, flow-centric visualizations that enhance system transparency and enable rapid response to network anomalies [17].
- Data-Fusion Pipeline for Enhanced AI Transparency: This data-fusion pipeline aggregates multisource inputs and supports transparency and interpretability in AI-driven decisions in dynamic V2X scenarios.

In summary, this study proposes a framework to enhance the performance and security of V2X applications deployed in edge environments. The proposed system is designed to optimize resource management, ensure real-time data processing, and enable secure operations, thereby contributing to the advancement of autonomous driving and ITSs. This study represents a step forward toward enabling the autonomous operation of V2X services in edge cloud hybrid environments, which is a critical component of the future of smart cities [18].

2. Background

The goal of this section is to provide a detailed overview of the key technologies and concepts that form the foundation of this study. First, we explore V2X technology, its applications, and the challenges it faces in real-time communication. Next, we examine edge computing as a solution to the latency and processing limitations of V2X. Then, we discuss the role of DevSecOps in securing and automating distributed edge environments. Finally, we demonstrate how AI workloads can be managed using K8S in edge computing systems and conclude with the motivation for an integrated framework that combines these technologies.

2.1. V2X Technology

V2X is a transformative technology that enables real-time communication between vehicles, infrastructure, networks, and pedestrians. This technology is essential for autonomous driving and ITSs, because it enhances road safety, improves traffic flow, and enables vehicles to make decisions based on real-time data. V2X encompasses four primary communication modes: V2V, V2I, V2N, and V2P [19]. Each mode plays a crucial role in maintaining a connected transportation ecosystem.

However, the wide-scale adoption of V2X faces several significant challenges. The most critical challenge is real-time data processing in a dynamic, fast-moving environment where delays can lead to safety risks. In addition, the high volume of data generated by V2X systems requires a robust infrastructure to handle communication across multiple entities in real time [20]. Issues such as network latency, bandwidth limitations, and security vulnerabilities further complicate the deployment of V2X [21]. The conventional reliance on centralized cloud infrastructures for processing these data typically results in latency that is unacceptable for safety-critical applications. Therefore, there is a growing need for decentralized solutions that can process data at the network edge.

2.2. Edge Computing and Its Role in V2X

Edge computing has emerged as a critical architecture for real-time data processing in distributed systems, particularly in environments requiring low-latency communication such as V2X [22]. Edge computing moves data processing closer to the source of data generation, reducing the need for long-distance data transmission and enabling faster, more responsive decision making, particularly when combined with Internet of Things-driven systems that require efficient handling of large, complex datasets [23]. In V2X environments, this approach is critical, because decisions about vehicle movement, traffic signals, and pedestrian interactions must be made in real time to ensure safety and efficiency [24].

Although edge computing provides significant advantages in latency reduction and bandwidth optimization, it introduces new complexities. Managing distributed edge nodes, which are typically heterogeneous in terms of hardware capabilities, presents challenges in resource allocation and workload distribution [25]. In addition, ensuring secure and reliable communication across a distributed network of edge nodes is difficult, particularly in environments such as V2X, where mobility and real-time data processing are paramount. The deployment and management of AI workloads across these distributed nodes further complicate resource management, requiring sophisticated orchestration mechanisms to effectively balance computational loads [26].

2.3. The Role of DevSecOps in Edge Environments

DevSecOps has emerged as a critical framework to address the security and automation challenges in edge computing environments. It integrates security practices within the CI/CD pipelines, ensuring that software is developed, deployed, and operated securely

from the outset [27]. In the context of V2X and edge computing, where the distributed nature of the infrastructure makes manual security management infeasible, DevSecOps plays an essential role in automating security processes.

In edge computing environments, where devices are distributed across a wide area and interact with each other in real time, security risks are heightened. Malicious attacks on V2X systems can compromise not only individual vehicles but also entire networks, posing significant risks to public safety [28]. DevSecOps enables the continuous monitoring and protection of these systems by automating security updates and ensuring that all deployed applications are compliant with the latest security standards. Despite its advantages, applying DevSecOps to highly dynamic and distributed edge environments is complex because it requires the integration of security protocols across multiple, typically heterogeneous, nodes. Therefore, a well-structured DevSecOps pipeline tailored to edge computing is critical for securing V2X deployments [29].

2.4. AI and Kubernetes for Workload Management

AI is increasingly being integrated into V2X systems to enable intelligent decision making based on real-time data from vehicles and infrastructures. AI models in V2X applications help process vast amounts of sensory data to predict traffic conditions, enhance autonomous driving capabilities, and improve overall road safety [30]. Similar AI applications in edge computing environments have been explored in smart manufacturing [31]. However, managing AI models in a distributed edge environment presents significant challenges. AI workloads must be deployed, scaled, and operated efficiently across a network of edge nodes, each with varying computational capabilities, which adds complexity to maintaining low-latency AI processing at the edge [32]. Federated learning offers a decentralized solution to distributing AI models across edge nodes while ensuring data privacy and minimizing latency [33].

To address these challenges, K8S, a cloud-native orchestration platform, has become the go-to solution for managing AI workloads in distributed environments. K8S allows the for automated deployment, scaling, and operation of AI models across multiple edge nodes [34]. This capability is particularly important in V2X systems, where AI models must continuously adapt to new data and provide real-time insights to improve decision making processes. However, managing multiple K8S clusters across distributed edge locations presents a set of challenges, including resource management, network synchronization, and workload distribution [35].

2.5. Motivation for an Integrated Framework

The increasing complexity of V2X systems, combined with the challenges presented by edge computing and the need for real-time AI-driven decision making, highlights the need for a comprehensive framework. This framework must integrate edge computing, DevSecOps, and AI workload orchestration to ensure that V2X applications are scalable, secure, and efficient. Current V2X deployments typically fail to address these challenges, particularly in terms of latency reduction, security automation, and resource optimization. Therefore, this study fills this gap by proposing an integrated V2X-Car Edge Cloud system that leverages DevSecOps to secure and automate the deployment of distributed AI workloads.

3. Requirements of the V2X-Car Edge Cloud for Driving Simulation

The V2X-Car Edge Cloud for driving simulation is a comprehensive framework designed to address the critical requirements (R#) for modern ITSs and autonomous vehicle simulations. By combining advanced edge computing capabilities with simulation environ-

ments, this framework bridges the gap between real-time processing needs and scalable, secure infrastructure. As illustrated in Figure 1, the V2X-Car Edge Cloud integrates three primary components:

- V2X-Car Edge Cloud: A distributed network of computational resources that enables real-time data processing and decision making for autonomous vehicles and ITSs.
- Smart Driving Simulations: These provide a controlled, realistic environment to validate and refine system performance under diverse scenarios, such as varying weather conditions, traffic density, and emergencies.
- Software Design for V2X-Car Edge Cloud: A modular and scalable architecture that leverages cloud-native technologies to orchestrate workloads, automate deployment, and enhance system security.

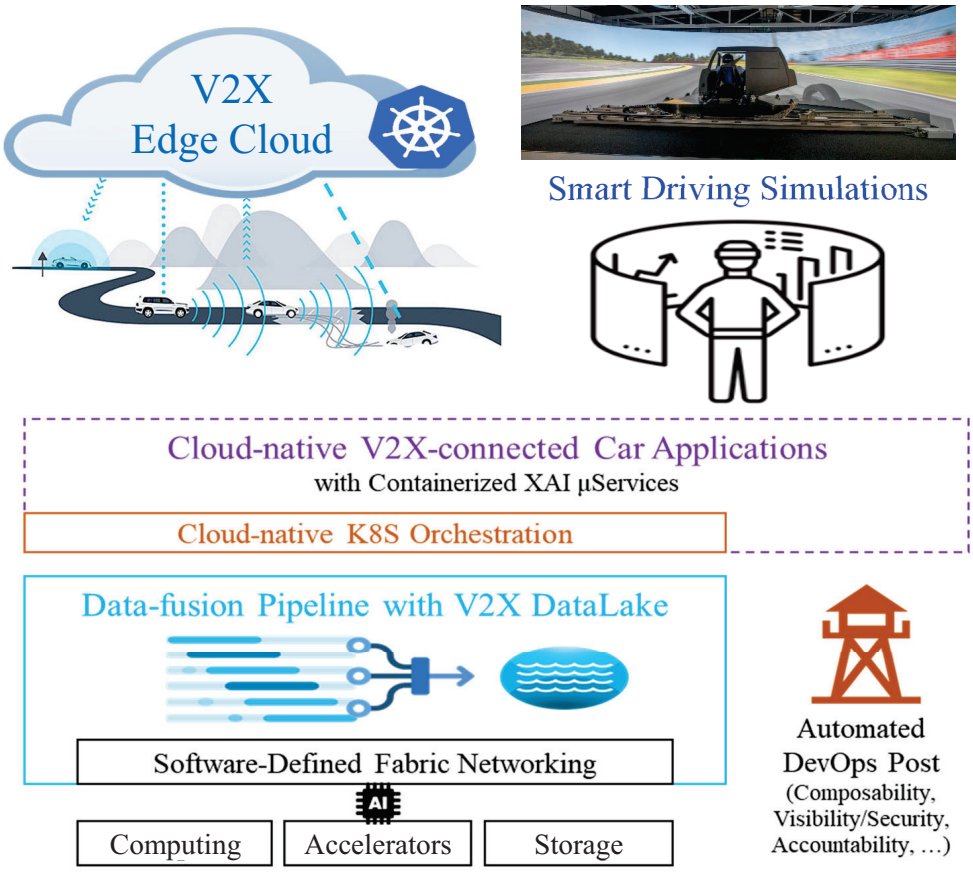

Figure 1. V2X-Car Edge Cloud for driving simulation.

The successful implementation of the V2X-Car Edge Cloud system requires us to address several critical challenges. These challenges are related to real-time data processing, scalability, security, and AI workload orchestration in a highly dynamic environment. The system must handle a large volume of real-time data while ensuring secure and efficient data processing across distributed edge nodes. In addition, the automation of the deployment and security processes is crucial to meet the requirements of modern V2X environments. Table 1 summarizes the key requirements for the V2X-Car Edge Cloud system. These requirements were identified through an extensive review of the latest literature on V2X communication, edge computing, AI workload management, and security automation [36–41].

Table 1. Key requirements for V2X-Car Edge Cloud system.

Requirements	Description	Challenge Addressed	Related Technology
R1: Cloud-Native Edge Intelligence Computing [36]	Efficient deployment of AI workloads across edge nodes using cloud-native principles.	Real-Time Data Processing, Scalability	K8S, Docker, Microservices
R2: SmartX DevSecOps Automation Framework [37]	Automating deployment and security for distributed environments.	Automation, Security	DevSecOps, CI/CD Pipelines
R3: Data-Fusion Pipeline for XAI Support [38,39]	Supporting XAI by integrating multisource data into a single pipeline.	Transparency, Data Management	XAI, Data Fusion
R4: SiLS-Based Service Realization in Hybrid-V2X Environments [40,41]	Validating AI services in SiLS environments with seamless integration into Hybrid-V2X networks.	AI Service Validation, Hybrid-V2X Communication	SiLS, C-V2X, WAVE

3.1. Cloud-Native Edge Intelligence Computing

The concept of cloud-native computing has transformed how distributed systems, including edge computing environments, handle complex workloads. In the context of V2X systems, the application of K8S and Docker facilitates the management of AI workloads across geographically distributed edge nodes. Previous research has shown that conventional centralized cloud infrastructures suffer from high latency and are unsuitable for the low-latency demands of V2X environments [38]. The adoption of cloud-native principles in edge computing allows for scalable AI processing with minimal latency, thereby meeting the critical real-time requirements of autonomous driving and ITSs [42].

Although earlier studies on edge computing focused on resource management at the infrastructure level, recent advances in microservice-based architectures and orchestration tools such as K8S enable more efficient deployment of AI models. This requirement builds on the limitations of centralized cloud approaches and extends them into decentralized edge environments [36]. By leveraging cloud-native technologies, the system ensures that AI models are efficiently deployed, scaled, and updated across multiple edge nodes.

3.2. SmartX DevSecOps Automation Framework

As V2X systems become more distributed and complex, conventional manual approaches to deployment and security management become impractical. DevSecOps, which integrates security into the CI/CD pipeline, has emerged as a robust solution for automating security and deployment tasks in distributed environments [27]. This is particularly critical in edge computing due to the distributed and dynamic nature of these environments. In the context of edge computing, security risks are heightened due to the distributed nature of the system and real-time data flows between vehicles, infrastructure, and the cloud.

Previous approaches for securing V2X systems focused on securing individual components or layers of the infrastructure. However, DevSecOps extends this by automating security updates, vulnerability scanning, and compliance checks across the entire distributed environment, thereby ensuring that security does not compromise performance [43]. By automating these processes, the system can respond to emerging threats more effectively while maintaining the required performance for real-time applications such as autonomous driving.

3.3. Data-Fusion Pipeline for XAI Support

The integration of data from multiple sources (e.g., sensors, cameras, and GPS) is critical for effective AI decision making in V2X environments, especially as the volume of data continues to increase in modern intelligent systems [44]. However, the complexity of these systems has given rise to concerns about the transparency and interpretability of AI decisions. Explainable AI (XAI) is a field dedicated to making AI decision making processes more transparent, enabling stakeholders to understand and trust the outputs of AI systems [41].

To achieve this, a robust data-fusion pipeline that aggregates data from multiple sources and provides coherent input to AI models is required. Previous systems have typically struggled with data integration issues, leading to fragmented or incomplete datasets for decision making. This requirement addresses these challenges by ensuring that the data provided to AI models are comprehensive, transparent, and suitable for XAI frameworks, thereby increasing the trust in and adoption of AI-driven V2X technologies.

3.4. SiLS-Based Service Realization in Hybrid-V2X Environments

The integration of SiLS with Hybrid-V2X communication technologies addresses critical challenges in validating and deploying AI services in vehicular environments. SiLS provides a controlled simulation platform for testing AI models and ensuring that they perform reliably in dynamic, real-world scenarios [45]. Through SiLS, AI workloads, such as object detection and path planning, can be evaluated iteratively without incurring the costs and risks associated with real-world deployments [46].

Hybrid-V2X communication, which encompasses C-V2X (cellular V2X) and WAVE (wireless access in vehicular environments), enables reliable and low-latency communication between vehicles, edge nodes, and cloud servers [47]. These communication protocols ensure seamless data exchange and real-time processing, which are critical in autonomous driving and emergency response scenarios.

Previous approaches have typically struggled with scalability and latency problems when integrating AI simulation and V2X communication frameworks [46]. Our implementation bridges this gap by leveraging SiLS for iterative validation and Hybrid-V2X for robust real-time communication, which aligns with requirement R3 (SiLS-Based Service Realization in Hybrid-V2X Environments).

The proposed system exhibits improved scalability, latency management, and AI validation reliability in complex vehicular scenarios, thereby addressing the challenges highlighted in earlier research. Through this integration, the V2X-Car Edge Cloud provides a scalable and efficient platform for ITSs.

4. Conceptual Design of V2X-Car Edge Cloud

Building upon the requirements (R#) identified in Section 3, this section outlines the design of the V2X-Car Edge Cloud system, focusing on how these requirements are realized through specific architectural components and methodologies. The design emphasizes scalability, low-latency processing, secure data handling, and efficient workload orchestration.

4.1. Conceptual Architecture of V2X-Car Edge Cloud

The conceptual architecture of the V2X-Car Edge Cloud is designed to meet the specific requirements (R#) identified in Figure 2.

The following descriptions explain how the architecture addresses each requirement:

- R1 (Cloud-Native Edge Intelligence Computing): This architecture employs a cloud-native approach to enable edge intelligence, where AI models and computing workloads are managed across distributed edge nodes. The AI Computing and Storage

Boxes execute real-time AI workloads, such as object detection and decision making, whereas the AI+X Post acts as the K8S master node, orchestrating dynamic workload distribution. This distributed infrastructure ensures seamless integration between cloud and edge layers, enabling efficient deployment and management of AI models

- R2 (SmartX DevSecOps Automation Framework): This architecture integrates the SmartX DevSecOps Automation Framework to realize secure and automated operations. This framework automates CI/CD processes, security policy enforcement, and system monitoring. The SmartX MultiSec tool enhances operational visibility by providing flow-centric visualizations and anomaly detection for network traffic. The integration of this framework ensures the secure and streamlined deployment of V2X services.
- R3 (Data-Fusion Pipeline for XAI Support): This architecture includes a data-fusion pipeline to support XAI in V2X applications. This pipeline aggregates data from multiple sources (e.g., sensors, cameras, and GPS) and feeds them to AI models for decision making. Results are visualized using the SmartX MultiSec tool to enhance the transparency and accountability of AI decisions. The data-fusion pipeline operates across the edge and cloud layers to enable effective data processing and explainability for V2X services.
- R4 (SiLS-Based Service Realization in Hybrid-V2X Environments): This architecture integrates an SiLS environment to validate AI services in controlled yet realistic driving scenarios. Using PreScan-based simulations, sensor and camera data streams are replicated to enable edge nodes to execute tasks such as object detection, Inference, and traffic analysis in a reproducible environment. In addition, Hybrid-V2X communication technologies (e.g., C-V2X and WAVE) ensure robust, low-latency data exchange between vehicles, infrastructure, and edge nodes.

In summary, the conceptual architecture of the V2X-Car Edge Cloud is designed to address four key requirements: cloud-native edge intelligence (R1), secure and automated operations (R2), data-fusion pipelines for XAI support (R3), and SiLS-Based Service Realization in Hybrid-V2X environments (R4). Each layer and component of the system, including the AI Computing and Storage Boxes, Data IO Boxes, AI+X Post, and SmartX MultiSec framework, play a critical role in achieving these objectives.

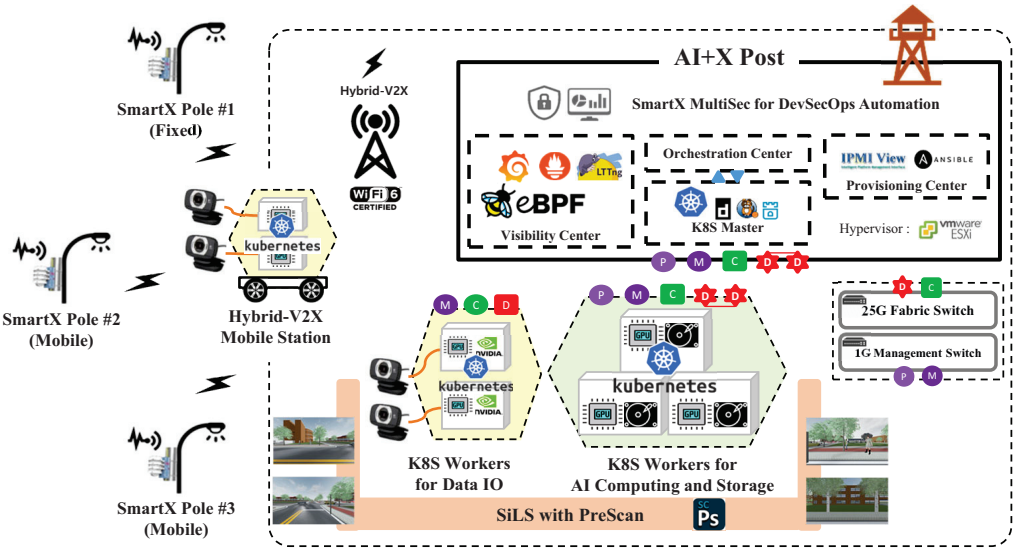

Figure 2. V2X-Car Edge Cloud architecture.

This architecture integrates cloud-native orchestration for scalable AI deployment, DevSecOps automation for secure and efficient operations, SiLS simulation environments

for realistic validation, and a data-fusion pipeline for enhanced AI transparency and explainability. These elements ensure seamless integration, scalability, and resilience across cloud and edge environments, thereby enabling the effective deployment and management of AI-driven V2X applications in dynamic vehicular contexts.

4.2. Integration of Requirements and Key Components

The integration of the V2X-Car Edge Cloud's key components and requirements (R#) ensures a seamless and unified framework for addressing the complex demands of V2X applications. Unlike the conceptual architecture described in Section 4, this section focuses on the software design of the proposed system and how its components collaborate to achieve end-to-end functionality. Figure 3 provides an overview of the software design, illustrating the relationships among the key components and their interactions to support the requirements. The integration process is structured into three primary stages:

- System Initialization and Resource Allocation: During the initialization phase, the AI+X Post orchestrates the registration of edge nodes, such as AI Computing and Storage Boxes and Data IO Boxes, ensuring that resources are dynamically allocated based on the real-time demands of the system (R1, R2). DevSecOps pipelines are activated to deploy security policies and CI/CD workflows, enabling secure and scalable operations (R2).
- Operational Coordination Across Components: Real-time sensor data collected by V2X devices flow through the Data IO Boxes to the AI Computing and Storage Boxes, where AI Inference models execute tasks such as object detection and traffic analysis (R1, R4). The data-fusion pipeline aggregates multisource inputs to support XAI (R3), whereas SmartX MultiSec visualizes traffic flows and performance metrics for enhanced transparency (R2, R3). These software modules collaborate to maintain consistent and efficient operations.
- Feedback and Adaptive Optimization: The proposed system continuously monitors latency, resource usage, and performance metrics through feedback loops supported by the SmartX MultiSec framework. This allows the AI+X Post to dynamically adjust workloads and update security policies as needed (R2, R4). The SiLS environment ensures validated and optimized AI models for deployment, further enhancing system reliability (R4). These adaptive workflows enable the system to effectively respond to dynamic vehicular environments.

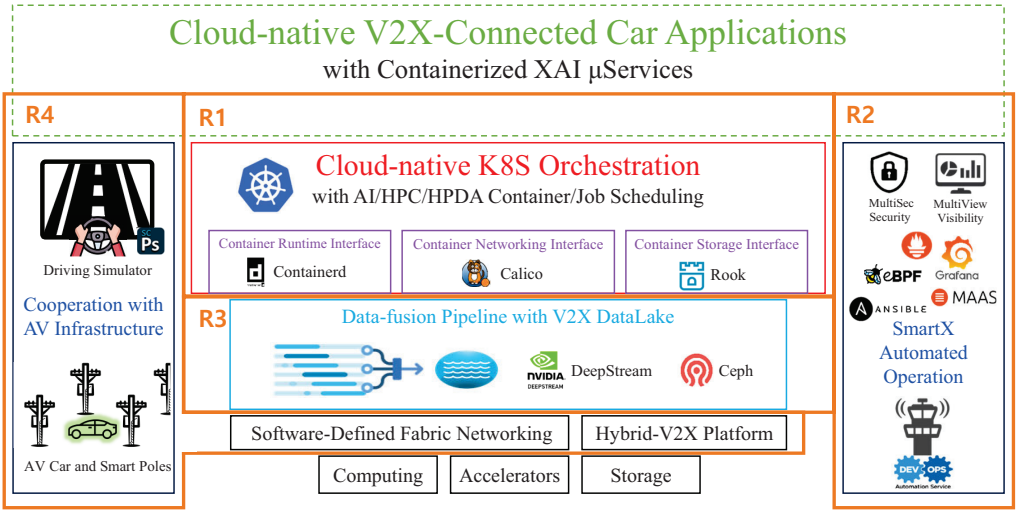

Figure 3. Software design for V2X-Car Edge Cloud.

In summary, the V2X-Car Edge Cloud integrates key software components and requirements into a cohesive framework, addressing the challenges of real-time processing, security, scalability, and transparency. By enabling seamless coordination across the initialization, operation, and optimization stages, the proposed system supports efficient and adaptive performance for advanced V2X services in dynamic vehicular environments.

5. Proof-of-Concept Implementation of V2X-Car Edge Cloud

This section provides an overview of the proof-of-concept implementation of the V2X-Car Edge Cloud. This section covers the hardware and software environment, initialization and preparation phases, and flow monitoring and security mechanisms of the proposed system. Section 5.1 outlines the simulation environment and infrastructure setup, and Section 5.2 details the startup procedures and connectivity configurations of the system. These subsections describe the practical implementation of the proposed V2X-Car Edge Cloud framework.

5.1. Overall Box Configuration of V2X-Car Edge Cloud

Figures 4–6 illustrate the overall hardware configuration of the V2X-Car Edge Cloud framework, including the physical devices that host the K8S containers described in Section 4. These physical devices, or "boxes", serve as the infrastructure for running distributed AI workloads, managing V2X data, and ensuring operational efficiency. The proposed system comprises AI+X Post Box, multiple AI Computing and Storage Boxes, and a driving simulation environment (both virtual and real-world).

- AI+X Post Box (Physical Host for AI+X Post): The AI+X Post Box is the physical host for K8S-based orchestration, managing workloads and ensuring security across the network. The AI+X Post Box is configured to run the K8S control plane, which is responsible for deploying and managing the lifecycle of containers across the system. This physical box is equipped with Intel Xeon computer processing units (CPUs), large memory capacity, and high-speed 25G network cards (NICs), enabling it to handle the orchestration of multiple edge nodes with minimal latency. Furthermore, this physical host is responsible for running security automation processes via the SmartX MultiSec framework, which continuously monitors network traffic, applies security patches, and manages the DevSecOps pipeline. As shown in Figure 4, the AI+X Post Box serves as the central hub for orchestrating AI tasks and managing the networked edge devices securely
- AI Computing and Storage Boxes (Physical Host for AI Computing and Storage Nodes): The AI Computing and Storage Boxes are physical hosts that run the K8S containers for AI processing and data storage. Each of these physical boxes is configured to execute specific tasks, such as real-time AI Inference, object detection, and data storage. The hardware specifications of each box include NVIDIA Tesla T4 graphic processing units (GPUs) for accelerating AI computations, Intel Xeon CPUs for general processing, and high-speed dual 25G NICs for efficient data transfer. These physical boxes host the K8S containers that perform specialized roles in the V2X ecosystem, such as the AI nodes that execute real-time object detection tasks. Each AI Computing and Storage Box is designed to process data locally at the edge, reducing the need for constant communication with the cloud, thereby minimizing latency and improving responsiveness for V2X applications. Furthermore, these boxes provide local storage solutions using integrated Ceph and BeeGFS systems, allowing them to store and manage large amounts of data generated by edge devices, such as video feeds from cameras and sensor data. In terms of networking, the AI Computing and Storage Boxes are connected via a 25G Fabric Switch, ensuring that large volumes of data

can be processed and transferred in real time. The K8S containers deployed on these boxes are managed by the AI+X Post Box, which dynamically allocates resources based on workload requirements, thereby ensuring optimal performance across the distributed infrastructure.

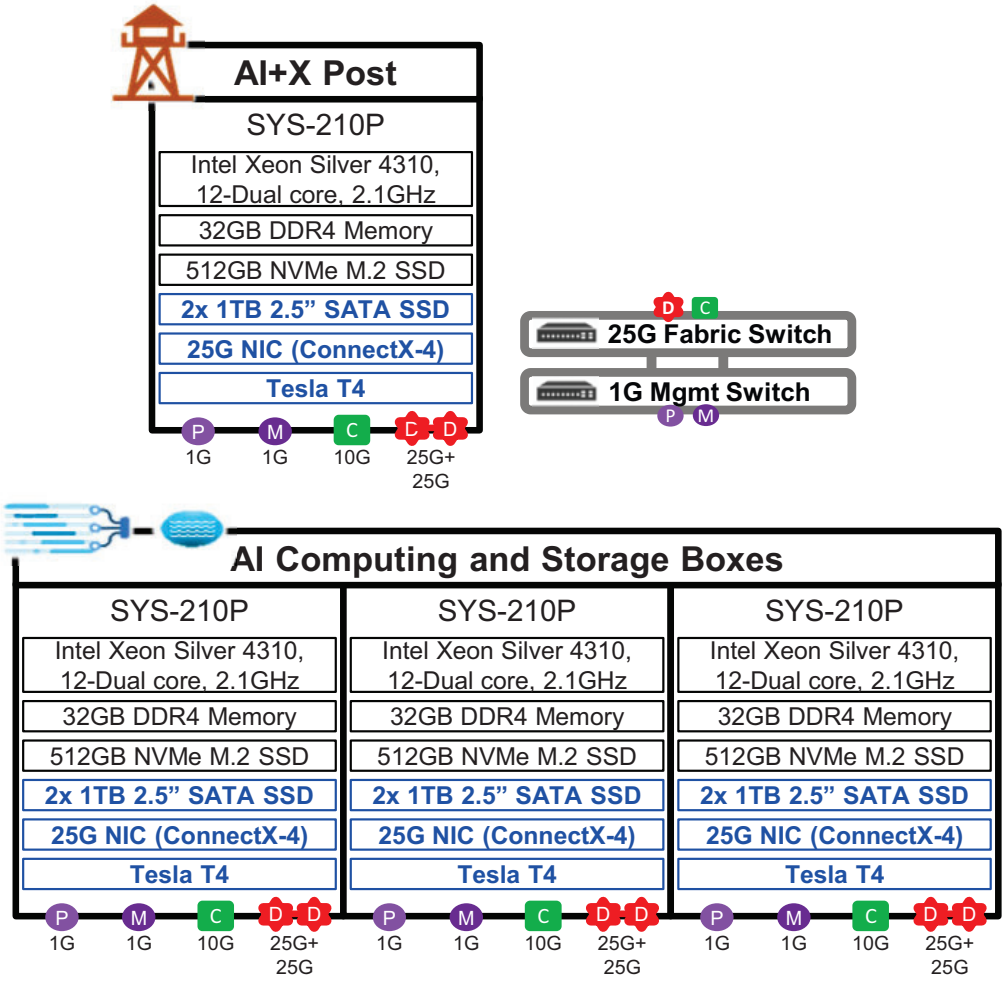

Figure 4. Hardware Specifications of AI+X Post and AI Computing and Storage Boxes.

- Virtual Driving Simulation Environment (PreScan Box and Data IO Boxes): The virtual driving simulation environment is built using PreScan, a simulation tool that replicates real-world driving conditions and traffic scenarios. In this setup, the PreScan Box runs the simulation software that shows a virtual driving environment on three tiled displays. The three tiled displays are captured by 12 cameras of Data IO Boxes, which are responsible for collecting and distributing these data to the AI Computing and Storage Boxes for real-time processing. The AI containers deployed on these physical boxes process the simulation data, executing tasks such as object detection and tracking to validate AI models in a controlled virtual environment. The virtual simulation environment allows for thorough testing of AI models and system behavior under various traffic conditions, such as vehicle-to-vehicle communication, emergency response scenarios, and pedestrian detection. The data generated from these simulations provide insights into how the system performs in real-world applications, thereby ensuring the robustness and reliability of AI models before deployment.

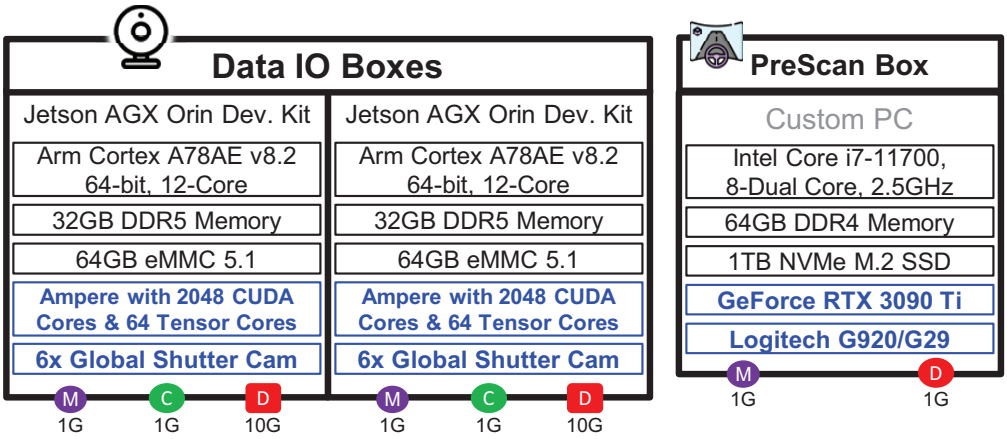

Figure 5. Hardware specifications of virtual driving simulation environment.

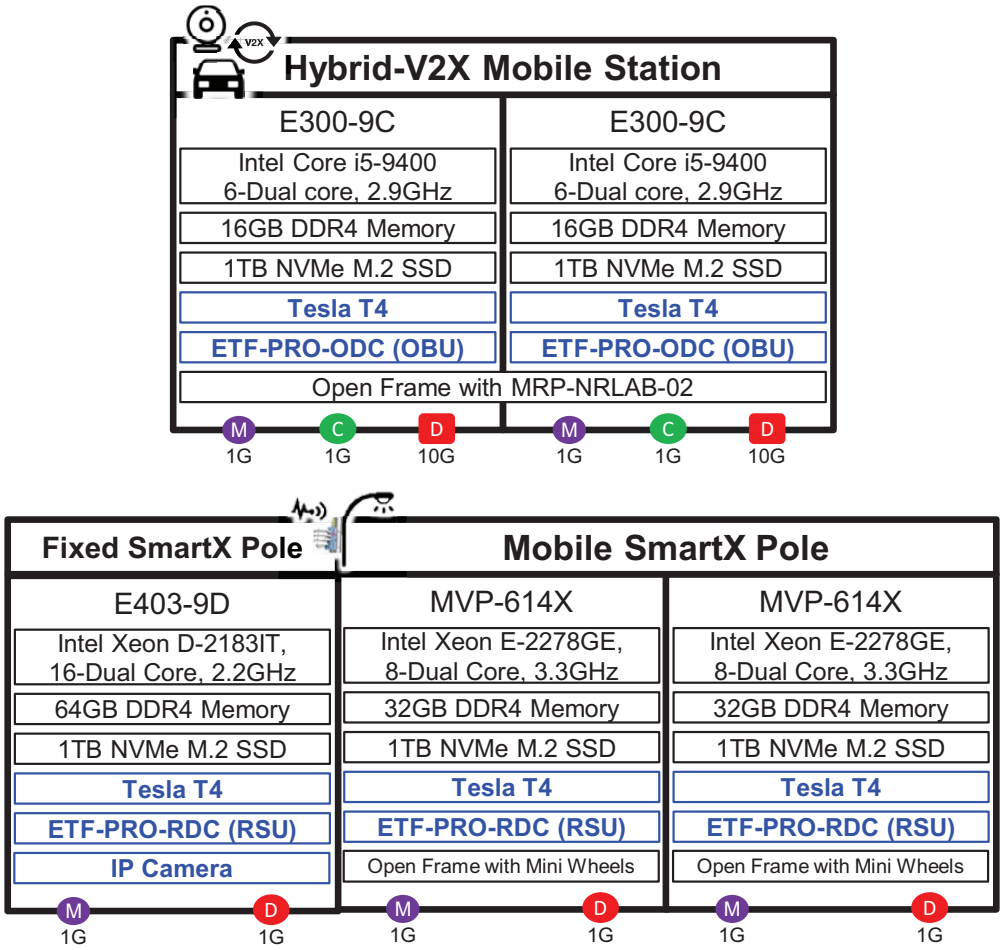

Figure 6. Hardware specifications of real-world driving environment.

- Real-World Driving Environment (Hybrid-V2X Mobile Station, Fixed SmartX Pole, Mobile SmartX Pole): In addition to the virtual environment, the real-world driving environment includes actual V2X devices and sensors that simulate real-world conditions. These include the following:
  - Hybrid-V2X Mobile Station: A mobile unit equipped with V2X communication technology that interacts with vehicles, infrastructure, and pedestrians in real time. This unit generates live data, such as vehicle speed, location, and proximity to other objects, which are then processed by the AI Computing and Storage Boxes.

- Fixed SmartX Pole: A stationary unit that monitors traffic and environmental conditions, equipped with sensors such as cameras and light detection and ranging (LiDAR). This unit communicates with vehicles and mobile stations and transmits data to the AI Computing and Storage Boxes for real-time analysis.
- Mobile SmartX Pole: This is similar to the Fixed SmartX Pole but is mobile, enabling dynamic monitoring of traffic conditions. This unit can be relocated to different areas to collect additional data or respond to changing traffic scenarios.

These real-world devices collect live data under actual road conditions and feed the data to the V2X-Car Edge Cloud. The AI Computing and Storage Boxes process these data using the same K8S containers that run the AI Inference tasks in the virtual environment, thereby ensuring consistency between the virtual simulations and real-world testing.

The overall network includes four key interfaces: Power (P), Management (M), Control (C), and Data (D). The Data (D) interface relies on bonded 25 G ports to enable high-throughput data transfer, whereas the Management (M) interface (over a 1 G switch) handles provisioning and monitoring tasks. The Control (C) interface synchronizes the orchestration between the AI+X Post Box and the edge nodes, whereas the Power (P) interface provides real-time power monitoring and control.

Figure 7 provides a visual representation of the hardware components used in the setup of the V2X-Car Edge Cloud. The figure illustrates the specific devices, including AI+X Post Boxes, AI Computing and Storage Boxes, and Data IO Boxes, allowing for a clear understanding of how these elements physically manifest in the proposed system. In this study, all experiments and validations were conducted in a virtual driving environment. This approach was adopted to simulate realistic V2X scenarios while maintaining a controlled and repeatable setup, ensuring that the functionalities of the system, including data flow, security, and AI model performance, could be thoroughly tested and evaluated. Consequently, the virtual environment serves as a comprehensive platform for validating the effectiveness and robustness of the V2X-Car Edge Cloud architecture.

Figure 8 illustrates a proof-of-concept implementation of a real-world driving environment. Three SmartX Poles, which recognize vehicles using V2X communication, are positioned near the road in a triangular formation. A mobile station, which acts as a vehicle, is set up in the middle of the road. In this implementation, we used a real vehicle; thus, we installed station boxes inside the real vehicle instead of using an open-frame cart. The roadside unit (RSU) on each SmartX Pole continuously engages in V2X communication with other RSUs and onboard units (OBUs) within range, constantly updating the status of the SmartX Poles and the mobile station. Unlike conventional V2X communication systems, this environment leverages AI Computing and Storage Boxes to rapidly process and update V2X communication data at the edge. This environment also enables dynamic experiments that are impractical by reproducing these scenarios within a virtual driving simulation environment.

In summary, the physical infrastructure of the V2X-Car Edge Cloud is designed to support the deployment of the K8S containers, as described in Section 4. The AI+X Post Box orchestrates the system, whereas the AI Computing and Storage Boxes execute AI tasks and store data locally. This physical setup ensures that real-time processing, secure communication, and scalable AI workloads are efficiently managed across the network, providing a robust platform for modern V2X applications. By integrating both virtual and real-world components, the V2X-Car Edge Cloud provides a comprehensive platform for AI model validation and system operation under various conditions. The virtual environment allows for controlled testing and model refinement, whereas the real-world environment ensures that the system performs effectively in real-time V2X scenarios.

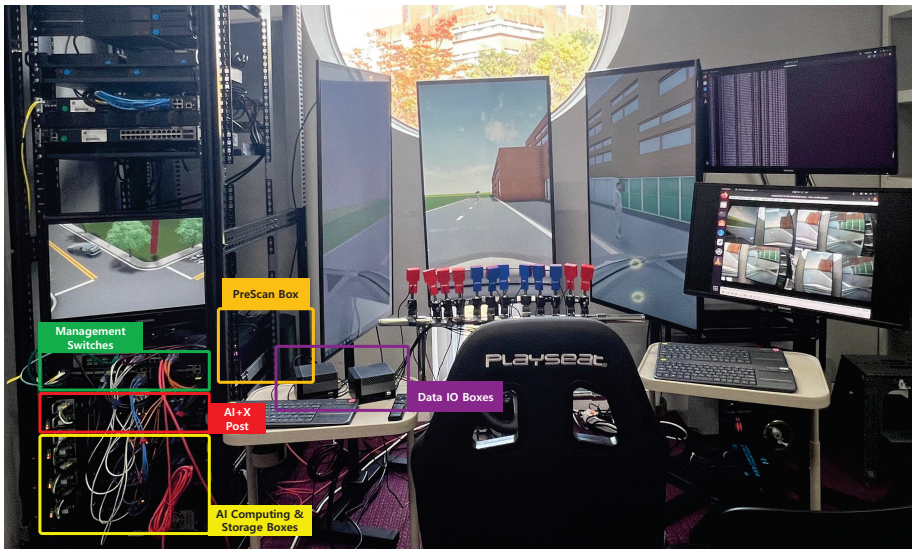

Figure 7. Virtual driving simulation environment over V2X-Car Edge Cloud.

Figure 8. Real-world driving environment over V2X-Car Edge Cloud.

5.2. Operation of V2X-Car Edge Cloud

The operation of the V2X-Car Edge Cloud involves multiple stages, from system initialization to real-time data processing and AI model execution. Figures 9–11 depict the sequence diagrams for the system initialization and preparation, respectively. These diagrams outline the detailed flow of operations at each stage, illustrating how the physical devices, specifically the PreScan Box and Data IO Boxes, collaborate to enable the full functionality of the proposed system for virtual driving simulations.

5.2.1. System Initialization in V2X-Car Edge Cloud

Figure 9 illustrates the initialization process of the proposed V2X-Car Edge Cloud system, focusing on the orchestration of the core components. The AI+X Post, acting as the K8S master node, initiates the power-up sequence for all connected edge devices, including the AI Computing and Storage Boxes, Data IO Boxes, and PreScan Box. The sequence begins with the AI+X Post sending power-up requests to each component and receiving confirmation of successful initialization.

Once powered up, the AI+X Post triggers clock synchronization across all devices to ensure consistent timekeeping for real-time data processing. Following synchronization, the K8S control plane on the AI+X Post initializes and registers all edge nodes (AI Computing and Storage Boxes) within the cluster, thereby establishing the infrastructure for distributed

workload orchestration. The process concludes with a system-wide status report, verifying the readiness of all components for the next operational phase.

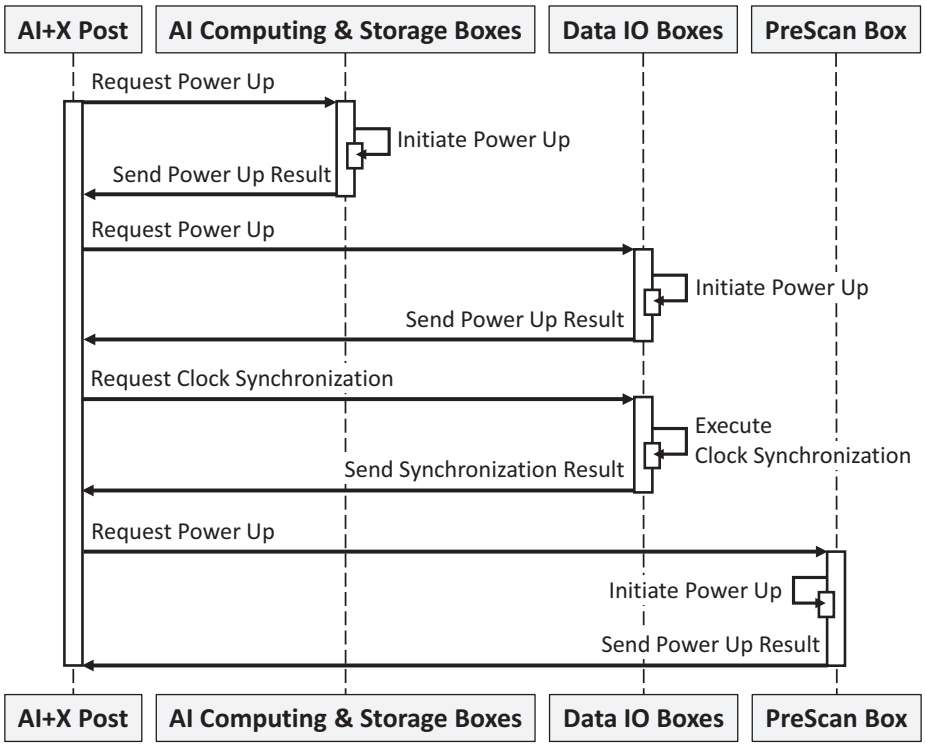

Figure 9. System initialization process in V2X-Car Edge Cloud.

This diagram highlights the coordinated startup of the physical and software components, ensuring seamless communication and synchronization across the V2X-Car Edge Cloud environment.

5.2.2. System Preparation for Traffic Analysis and Visualization

Figure 10 illustrates the preparation phase of the V2X-Car Edge Cloud, focusing on traffic flow monitoring and the visualization setup. This phase begins after the successful initialization of all system components. The AI+X Post coordinates the activation of monitoring tools across distributed nodes, including the AI Computing and Storage Boxes, Data IO Boxes, and PreScan Box.

During this phase, the proposed system initiates the SmartX MultiSec framework, which provides real-time, flow-centric traffic visualization. MultiSec containers are deployed on each node, capturing network traffic data and generating three-dimensional (3D) onion-ring visualizations that represent traffic flow patterns and their associated delay metrics. The visualization setup allows administrators to intuitively monitor system performance and identify anomalies in real time.

The sequence diagram highlights the interaction between the AI+X Post and other components during the visualization setup process. It captures the flow of data requests and responses required to establish synchronized traffic flow monitoring across all nodes, ensuring that the proposed system is equipped for real-time operation.

Figure 11 illustrates the workflow for object detection and latency analysis in the V2X-Car Edge Cloud, detailing the real-time processing tasks and the preparation of the supporting infrastructure for these operations. This workflow ensures that the proposed system can execute object detection tasks while capturing and analyzing latency metrics to optimize performance.

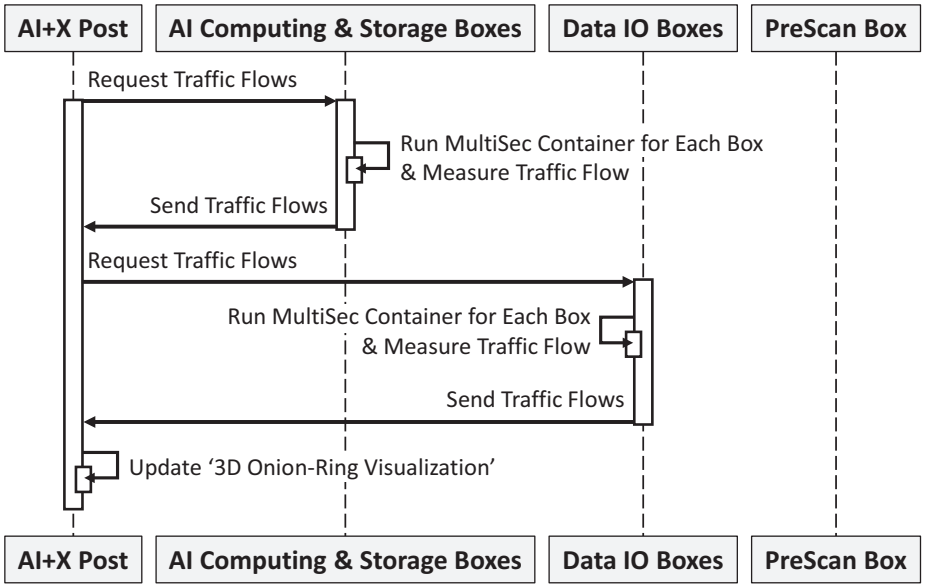

Figure 10. Traffic flow monitoring and visualization setup in V2X-Car Edge Cloud.

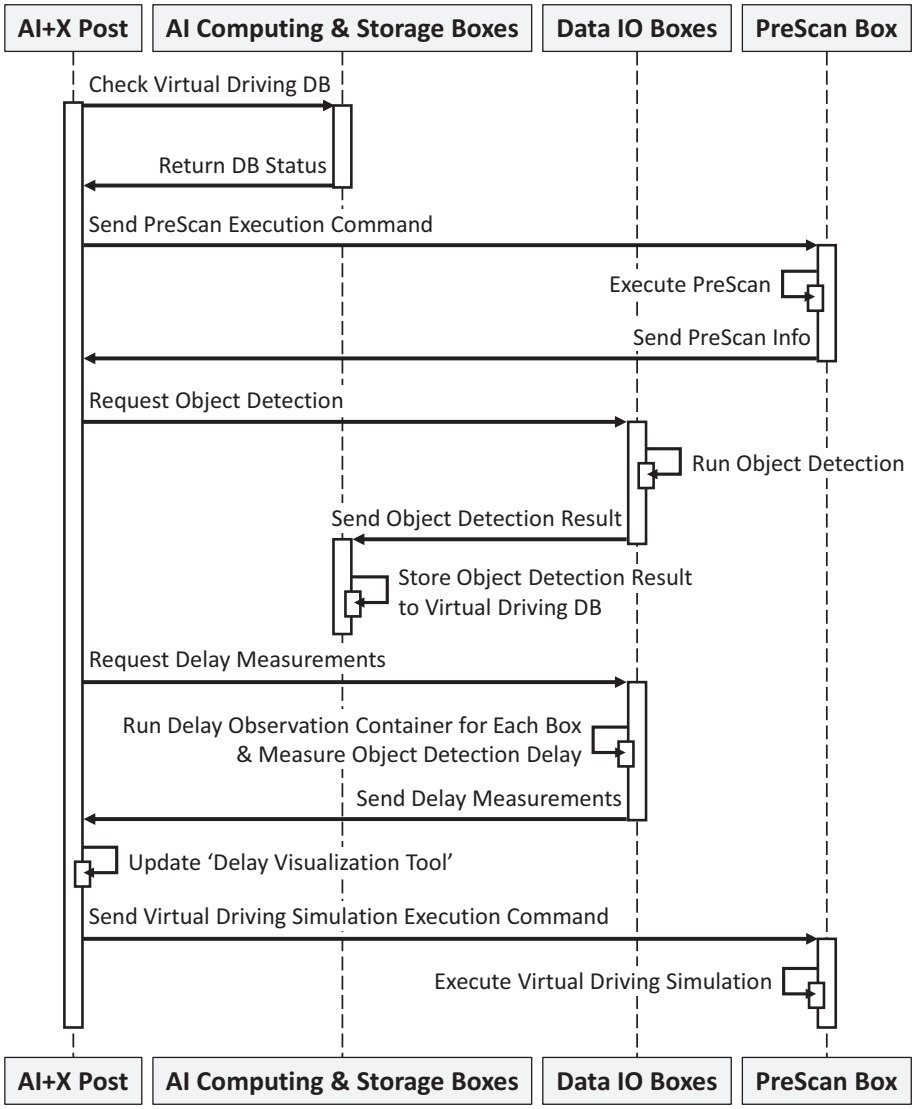

Figure 11. Object detection and latency analysis workflow in V2X-Car Edge Cloud.

The process begins with the AI+X Post verifying the status of the virtual driving database to ensure that the prior object detection results are correctly stored and accessible. Once the database readiness is confirmed, the AI+X Post sends a PreScan execution command to the PreScan Box, which initiates the virtual driving simulation and streams sensor data to the Data IO Boxes. These Data IO Boxes handle the preprocessing and forwarding of these data to the AI Computing and Storage Boxes.

Upon receiving the simulation data, the AI Computing and Storage Boxes deploy object detection models to identify and classify objects in the driving simulation. The detected objects and their associated results are stored in the virtual driving database for further analysis and validation. Simultaneously, the delay observation containers at each node measure the processing latency of the object detection task. These delay metrics, along with the object detection results, are visualized using a 3D onion-ring visualization tool to provide intuitive insights into system performance.

This workflow demonstrates the comprehensive coordination between data storage, simulation execution, AI-based object detection, and performance monitoring. By validating the database status upfront and integrating latency analysis into real-time operations, the V2X-Car Edge Cloud ensures reliable, efficient, and transparent processing for V2X applications.

6. Flow Monitoring and Security Automation in V2X-Car Edge Cloud

Figure 12 outlines the flow monitoring, security processing, and data management in the V2X-Car Edge Cloud architecture. This section details how these processes are handled by two primary components: the AI+X Post and the DevSecOps Tower. The AI+X Post is responsible for collecting and managing network flow data from edge nodes, whereas the DevSecOps Tower focuses on deeper security analysis, policy enforcement, and visualization.

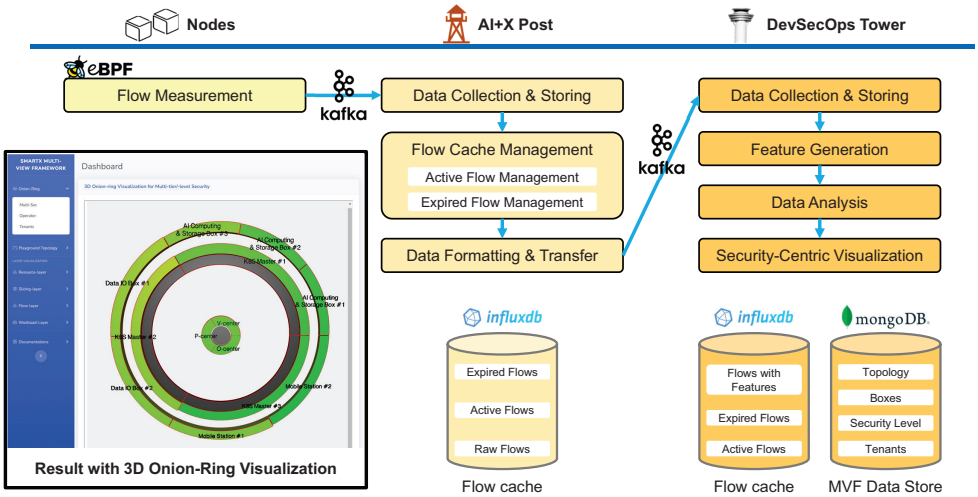

Figure 12. Functions for SmartX MultiSec.

6.1. Data Processing and Flow Management by AI+X Post

The AI+X Post acts as a central processing unit for managing network flow data. It performs three key functions: Data Collection and Storing, Flow Cache Management, and Data Formatting and Transfer.

1. Data Collection and Storing: The AI+X Post aggregates the flow data received from the distributed edge nodes. This includes traffic metrics, packet details, and communication patterns. The collected data are stored in InfluxDB, a time-series database designed to handle high-throughput data ingestion and real-time querying. InfluxDB

enables efficient storage and retrieval of network flows, ensuring that the V2X-Car Edge Cloud can handle large volumes of incoming traffic while supporting both real-time and long-term security analysis

2. Flow Cache Management: The AI+X Post maintains a cache to effectively manage active network flows. This Flow Cache Management mechanism is responsible for categorizing flows based on their status: Active Flow Management for ongoing data transfers and Expired Flow Management for completed flows. Active flows are cached to facilitate rapid access and real-time processing, ensuring that latency-sensitive V2X operations are handled efficiently. When flows become inactive, they are removed from the cache and transferred to InfluxDB for permanent storage and analysis. The caching strategy allows the AI+X Post to prioritize critical V2X traffic and manage system resources dynamically, ensuring that important data are processed rapidly while maintaining a comprehensive record of historical flows.
3. Data Formatting and Transfer: After caching and managing flows, the AI+X Post formats the flow data into structured datasets suitable for further security analysis by the DevSecOps Tower. This involves converting packet-level information into high-level metrics and features, which are necessary for advanced security functions. Once formatted, the data are securely transferred to the DevSecOps Tower using encrypted protocols to ensure data integrity and confidentiality. By working closely with InfluxDB, the AI+X Post guarantees that the data are consistently organized and prepared for deeper analysis and policy enforcement.

Overall, these components and configurations contribute to a cohesive and efficient architecture for the V2X-Car Edge Cloud. The integrated design of hardware and software components supports seamless data flow, rapid AI model deployment, and secure edge communication. These elements collaborate to realize real-time adaptability, optimized resource allocation, and reliable security enforcement, which are essential for handling the dynamic and complex nature of V2X scenarios. This overview lays the groundwork for understanding the performance capabilities of the proposed system and the specific workflows that support its operation across diverse vehicular applications.

6.2. Security Management and Policy Enforcement by DevSecOps Tower

The DevSecOps Tower is responsible for performing comprehensive security analysis of the data collected by the AI+X Post. It operates through four primary functions: Data Collection and Storing, Feature Generation, Data Analysis, and Security-centric Visualization. The tower uses both InfluxDB and MongoDB databases to efficiently manage different types of data and analysis results.

1. Data Collection and Storing: Upon receiving flow data from the AI+X Post, the DevSecOps Tower stores the data in InfluxDB and MongoDB. InfluxDB is used to store time-series data that require rapid querying and real-time processing, such as traffic volumes, timestamps, and network performance metrics. MongoDB serves as a flexible NoSQL database for storing more complex or unstructured data, such as node metadata, security policies, and relationships between various components of network flows. This dual-database approach allows the tower to effectively handle diverse data types, providing rapid access for real-time security operations and deeper insights for comprehensive policy management.
2. Feature Generation: In the Feature Generation phase, the DevSecOps Tower processes the raw flow data to extract structured features that are essential for security analysis. This includes creating feature sets to represent traffic patterns, flow durations, security attributes, and behavior across nodes. InfluxDB is used to store time-based features, allowing for the rapid access and analysis of temporal data. In contrast, MongoDB

handles more complex, multidimensional features and relationships that require flexible data structures. By efficiently generating and storing features in both databases, the tower can ensure that data are well organized and easily accessible for in-depth analysis and security decision making.

3. Data Analysis and Automated Security Response: The DevSecOps Tower uses the extracted features to analyze flow data through a combination of machine learning algorithms, statistical models, and rule-based security policies. The Data Analysis phase identifies potential anomalies, threats, or deviations from normal network behavior. InfluxDB supports real-time analysis by providing rapid access to temporal data, whereas MongoDB allows for complex analysis involving node relationships, security policies, and multidimensional data patterns. If a security incident or anomaly is detected, the DevSecOps Tower initiates automated security responses, such as isolating affected nodes, deploying security patches, or adjusting network configurations. This automated decision making helps maintain the security and stability of the V2X-Car Edge Cloud with minimal manual intervention.
4. Security-Centric Visualization: The final stage in the DevSecOps Tower workflow is Security-centric Visualization. This process converts the analyzed data into visual dashboards and monitoring tools that provide real-time insights into the security state of the system. InfluxDB provides time-series visualizations of traffic patterns, flow metrics, and detected threats, whereas MongoDB adds depth to the visualization by showing the complex relationships between nodes, flows, and security policies. These visualization tools allow administrators to rapidly identify and respond to security incidents, track system performance, and optimize resource allocations within the V2X-Car Edge Cloud. The integrated view from both InfluxDB and MongoDB ensures a comprehensive and actionable perspective on the security health of the system.

In summary, the detailed processes and mechanisms provide a comprehensive understanding of how data flows, security policies, and resource management are orchestrated in the V2X-Car Edge Cloud. By addressing the key aspects of secure data handling, AI model deployment, and real-time analytics, the proposed system demonstrates a robust and adaptable framework that meets the complex demands of edge computing in V2X environments. The interplay between the components, from data acquisition to policy enforcement, illustrates the capability of the architecture to provide reliable performance, security, and scalability across various use cases. This sets the stage for understanding how the proposed solutions effectively meet the dynamic needs of connected vehicular applications.

7. Scenario-Based Functional Verification

In this section, the smooth performance of self-driving AI services over an automated cloud infrastructure is verified. To realize this, we evaluate the orchestration, visibility, and visualization capabilities enabled by the DevSecOps Automation Framework for automated cloud operations. Due to safety constraints in real-world driving environments, conducting high-speed or dynamic driving tests is challenging. Therefore, the functionality of V2X communication is verified in a real-world environment, and other functionalities are tested in a virtual driving environment. As part of the verification, we perform an AI service that provides object detection to determine the virtual traffic situation. This service is primarily performed on the V2X-Car Edge Cloud using an SiLS-based virtual driving simulator. Consequently, the virtual environment setting is necessary, as are the settings for legacy hardware and software.

7.1. Simulation Overview

Figure 13 shows the functional verification workflow of the proposed V2X-Car Edge Cloud system, as validated in an SiLS environment. The simulation setup uses PreScan to create realistic virtual driving scenarios, generating real-time sensor and visual data streams that simulate autonomous driving conditions. The edge infrastructure, including the AI+X Post, Data IO Boxes, and AI Computing and Storage Boxes, operates collaboratively to ensure efficient data processing, storage, and visualization.

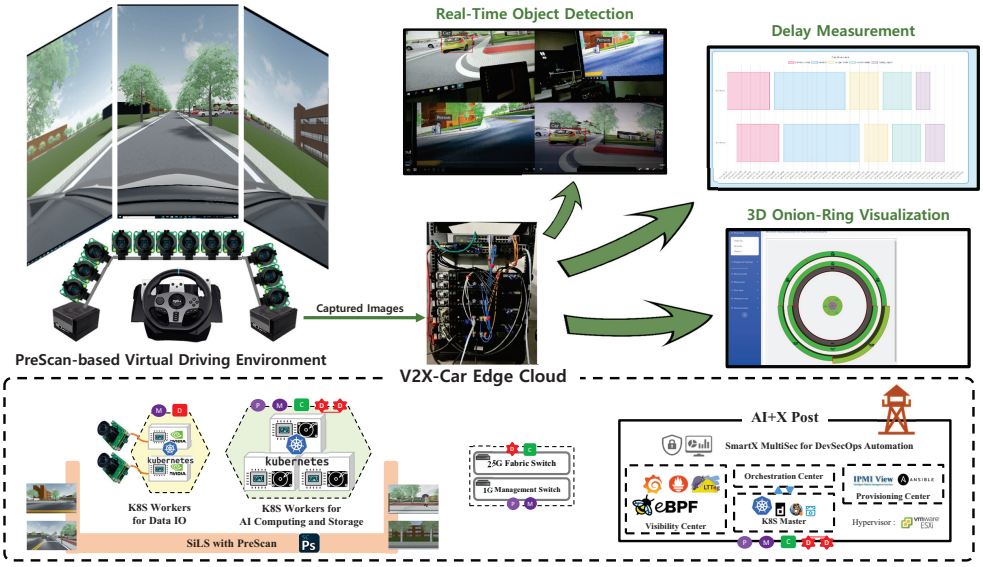

Figure 13. Functional verification workflow of V2X-Car Edge Cloud.

In this architecture, the Data IO Boxes perform object detection tasks using the YOLOv3 Inference model. The decision to run object detection on the Data IO Boxes stems from their ability to preprocess and analyze raw sensor data as close to the source as possible, thereby reducing the overall system latency and minimizing data transfer overhead to downstream components. Once the objects have been detected and classified, the results are forwarded to the AI Computing and Storage Boxes, where they are stored for further validation, analysis, and logging. This division of tasks ensures a balance between real-time data processing and efficient resource utilization across edge nodes.

The AI+X Post, acting as the control plane, manages the overall resource orchestration and system monitoring. Integrated with the SmartX MultiSec framework, the proposed system supports advanced visualization tools that provide operational insights. Specifically, 3D onion-ring visualization, as part of MultiSec, intuitively represents latency and detection results. Objects detected with minimal delay are displayed in the inner rings, whereas those experiencing higher latency are positioned in the outer rings. This visualization allows for the rapid identification of performance bottlenecks, enhancing system transparency and facilitating timely adjustments.

Through this simulation, the V2X-Car Edge Cloud demonstrates its ability to efficiently process real-time sensor data, execute object detection tasks with low latency, and dynamically monitor system performance. The integration of AI Inference, resource orchestration, and visualization tools ensures that the proposed system effectively supports latency-sensitive autonomous driving applications in dynamic environments.

7.2. Simulation Setting

The simulation experiment was conducted to evaluate the performance and scalability of the proposed V2X-Car Edge Cloud system in realistic autonomous driving scenarios. Figure 14 shows the virtual environment created using the PreScan SiLS platform. The environment replicates a dynamic urban intersection with virtual vehicles, pedestrians, and traffic signals to simulate real-world traffic conditions.

Figure 14. A section of GIST campus map and its virtual representation for simulation.

A hybrid approach was adopted in the experiment, where actual cameras were used to capture high-resolution video data of the PreScan environment. This setup ensures that the proposed system processes realistic visual inputs that are similar to those generated by onboard cameras in autonomous vehicles. In addition to camera data, PreScan generates simulated LiDAR and GPS data streams, which provide spatial and positional information for objects in the environment.

The specifications of the sensors used in this experiment are detailed in Table 2. The camera produces high-definition video at 30 frames per second with a resolution of 1920 × 1080 pixels, providing clear and continuous visual data for object detection tasks. The virtual LiDAR sensor operates with a range of 100 m and an angular resolution of 0.2 degrees, delivering accurate distance and spatial information. The simulated GPS outputs location data with an accuracy of ±0.5 m, enabling precise vehicle localization and trajectory monitoring.

The edge infrastructure processes these sensor inputs in real time. The Data IO Boxes handle the camera data and execute object detection tasks using the YOLOv3 Inference model. The detected objects, including their classifications and positional data, are forwarded to the AI Computing and Storage Boxes, where the results are stored and analyzed for further validation. The AI+X Post orchestrates resource allocation, ensures system synchronization, and monitors the overall performance of the edge infrastructure.

The hardware configurations supporting this experiment are summarized in Table 3. These settings include high-performance CPUs, GPUs, and memory resources to ensure efficient real-time processing and seamless workload management across edge components.

To validate the performance of the proposed system, a dynamic driving scenario was implemented (Figure 15). In this scenario, virtual vehicles interact with traffic signals, pedestrians, and other dynamic objects while navigating intersections. The video data captured by the real camera are processed on the edge infrastructure in real time to identify and classify objects. Additional LiDAR and GPS data streams were also used for spatial accuracy and performance validation.

Table 2. PreScan configuration.

Parameter	Description	Value
Campus Coverage	Simulation area covered	~30% of total campus + additional 4 km road + a hospital
Number of Cameras	A display that shows the view of Ego car	3
Camera FOV	Field of view (degrees)	60°
Simulation Time Step	Time per simulation step	0.05 s
Vehicle Count	Number of vehicles in the scenario	21
Pedestrian Count	Number of pedestrians	29
Distance Unit	A unit of distance between locations in simulation map	meter
Starting Position	Location where Ego car starts	x: 68.874153137207, y: 153.667175292969, z: 0.569999992847443
Injured Person's Position	Emergency situation location	x: 1.9, y:5.3, z:0.2
Hospital's Position	Ending point of the simulation	x: −1615.89830779929, y: −3874.55717468469 z: 0
Pole 1's Position	Located at each corner of the intersection	x: 2.7453999519348145, y: 10.487871170043945, z: 4
Pole 2's Position	Located at each corner of the intersection	x: −1.858980655670166, y: −2.70064640045166, z: 4
Pole 3's Position	Located at each corner of the intersection	x: 13.594693183898926, y: 2.9094009399414063, z: 4
Pole 4's Position	Located at each corner of the intersection	x: 10.242550373077393, y: −4.8233532905578613, z: 4
Lighting Conditions	Optical effects such as daylight/evening or specific lights	Clear, daylight
Weather Condition	Weather affecting the accuracy of object representations	Sunny

Table 3. YOLOv3 configuration.

Parameter	Description	Value
Input Resolution	Image dimensions	416 × 416 pixels
Confidence Threshold	Minimum confidence for detection	0.5
NMS Threshold	Overlap threshold for bounding boxes	0.45
Batch Size	Number of frames processed together	6
Number of Classes	Types of objects to detect	10

This simulation setting, which integrates PreScan virtual environments, real-world visual inputs, and edge computing infrastructure, provides a robust platform for evaluating the proposed V2X-Car Edge Cloud system. The experiment demonstrates the ability of the system to perform real-time object detection, store results efficiently, and monitor latency dynamically under realistic latency-sensitive conditions.

Figure 15. Roadside of GIST campus and its virtual driving environment.

7.3. Simulation Scenario

The simulation scenario highlights the coordination and real-time processing capabilities of the V2X-Car Edge Cloud system in a dynamic virtual environment. Figure 16 illustrates a critical emergency scenario in which a virtual vehicle communicates with a virtual pole to transport an injured person to the nearest hospital. This process involves seamless information exchange, resource orchestration, and real-time decision making to ensure timely execution of the emergency response task.

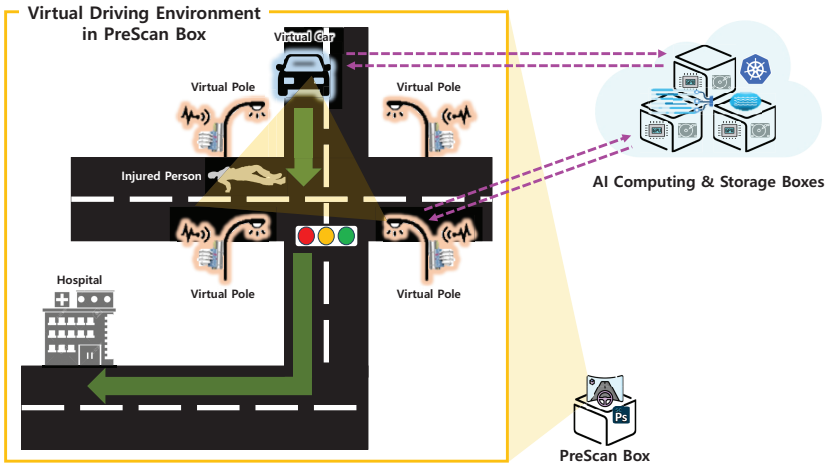

Figure 16. Virtual driving scenario overview.

In this scenario, the virtual vehicle generates and transmits vehicle information, including its current position, speed, and status, to the virtual pole. Simultaneously, the virtual pole gathers and forwards environmental information, such as detected objects (e.g., injured individuals or obstacles), traffic light status, and road conditions, to the AI Computing and Storage Boxes. This exchange allows the edge infrastructure to determine the most efficient route for the virtual vehicle and manage the flow of data to ensure smooth operations.

The visual data required for object detection are captured by a high-resolution virtual camera in the PreScan-generated environment. These data are processed by the Data IO Boxes, where the YOLOv3 model performs real-time object detection. Objects such as pedestrians, vehicles, and infrastructure components are identified and classified. By processing data locally at the Data IO Boxes, latency is minimized, enabling the proposed system to rapidly respond to critical events.

Once the detection results, including object classifications and positional information, are generated, they are transmitted to the AI Computing and Storage Boxes. These boxes store the results for further validation, analysis, and decision making. The AI+X Post oversees resource orchestration, ensuring that the virtual vehicle receives appropriate instructions via the virtual pole. This process ensures that the emergency response workflow is executed efficiently and reliably.

Figure 17 illustrates the detailed sequence diagram for the emergency response scenario described in the virtual driving simulation. The diagram visualizes the flow of information between the system components as the virtual vehicle executes a mission to transport an injured person to the nearest hospital. The process highlights the interaction between the Data IO Boxes, AI Computing and Storage Boxes, and AI+X Post to ensure real-time decision making and task execution.

The sequence begins when a virtual vehicle equipped with a camera detects an injured person in the PreScan-generated environment. The captured data are transmitted to the Data IO Boxes, where the YOLOv3 object detection model processes the visual input, identifies the injured person, and generates detection results, including bounding box coordinates and classifications.

The detection results, such as the injured person's location, are then sent to the AI+X Post. As the central control unit, the AI+X Post validates the incoming information and forwards it to the AI Computing and Storage Boxes for further analysis and decision making. The AI Computing and Storage Boxes determine the optimal route for the virtual vehicle to transport the injured person to the nearest hospital.

Once the route is computed, the AI Computing and Storage Boxes send instructions back to the Data IO Boxes. The Data IO Boxes, which control the virtual vehicle, receive the updated route information and initiate emergency transport operations. Throughout this process, the proposed system ensures minimal latency by efficiently orchestrating the flow of data and tasks across the components.

The sequence diagram demonstrates the ability of the V2X-Car Edge Cloud to manage real-time object detection, validate and analyze critical information, and provide task execution instructions. By enabling seamless communication between system components, the architecture supports reliable and low-latency operations, particularly for mission-critical scenarios such as emergency response.

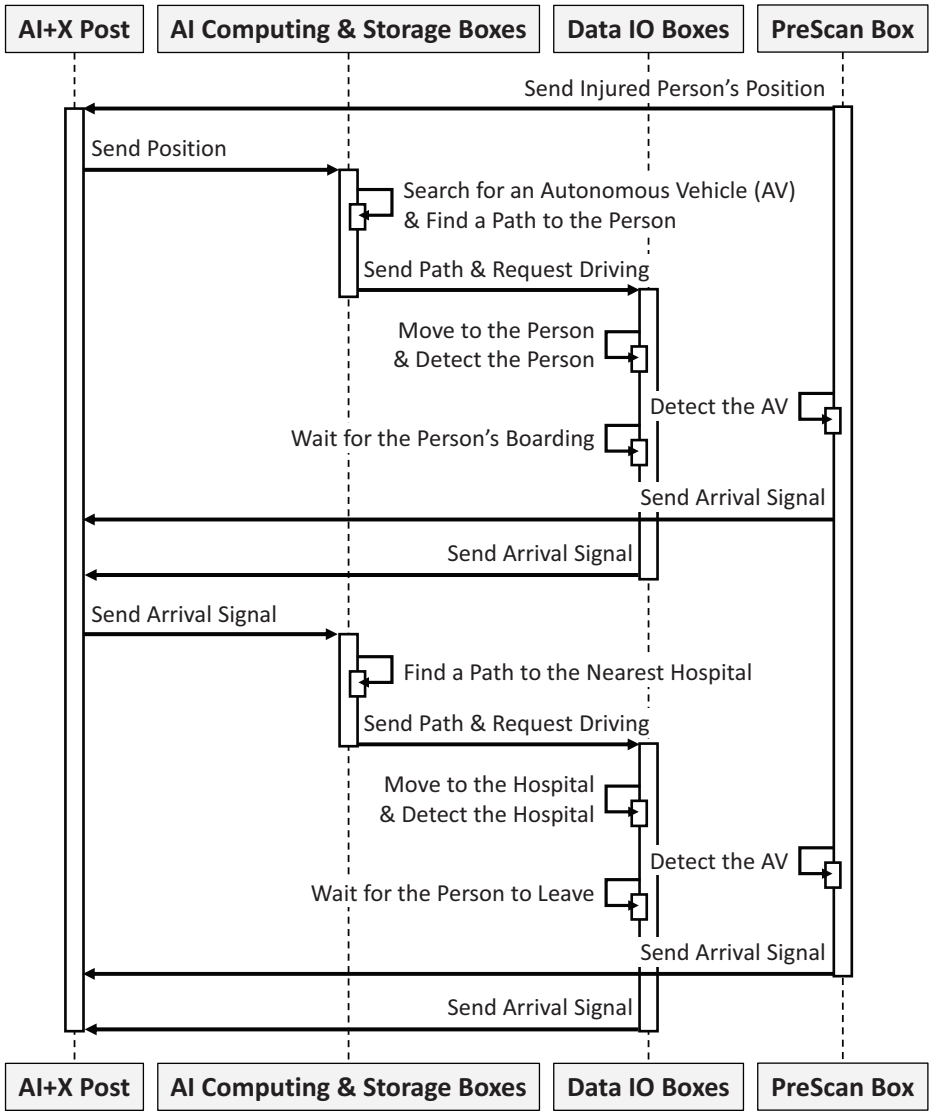

Figure 17. Sequence diagram for the virtual driving simulation.

7.4. Simulation Result

Figure 18 presents a real-time visualization of the vehicle and environmental data in the virtual driving simulation, which are captured immediately after the simulation has started. This initial view provides a snapshot of the early conditions of the simulation, where key entities such as the virtual vehicle and SmartX Pole are actively monitoring and interacting in the environment.

In this frame, the vehicle displays critical data such as current speed, direction (current angle), and location (x, y coordinates), which are mapped to the layout of the campus for spatial accuracy. These initial data allow operators to verify the vehicle's starting parameters, orientation, and readiness to respond to events as they unfold.

The SmartX Pole functions as a stationary monitoring device, collecting data on nearby entities, and is ready to trigger an emergency signal in response to any detected emergencies, such as the presence of an injured pedestrian. As the simulation progresses, additional elements such as detected objects are identified and tracked using the YOLOv3 algorithm, allowing for real-time monitoring of pedestrians, vehicles, and other objects in the virtual environment.

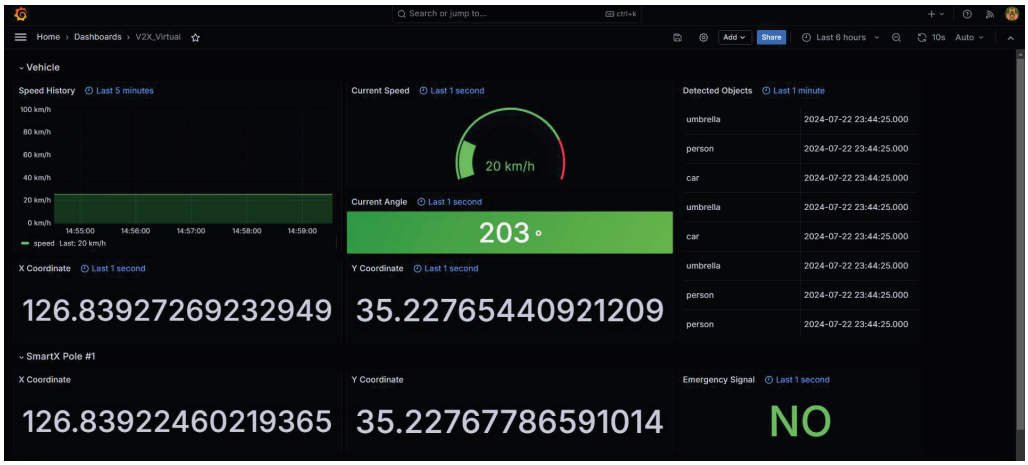

Figure 18. Real-time visualization of vehicle and environmental data in virtual driving simulation.

Similar to Figure 18, the dashboard in Figure 19 provides real-time monitoring of V2X communication data in a real-world driving environment. The dashboard incorporates the status of the actual SmartX Poles and the mobile station, providing a detailed overview of the operational state of the proposed system. The OBU Status table at the bottom of the dashboard displays the actual GPS coordinates of the mobile station. In addition, it includes acknowledgment (Ack) signals that confirm successful communication with each RSU. This visualization demonstrates the ability of the proposed system to effectively handle various types of data, including V2X communication data, in future applications.

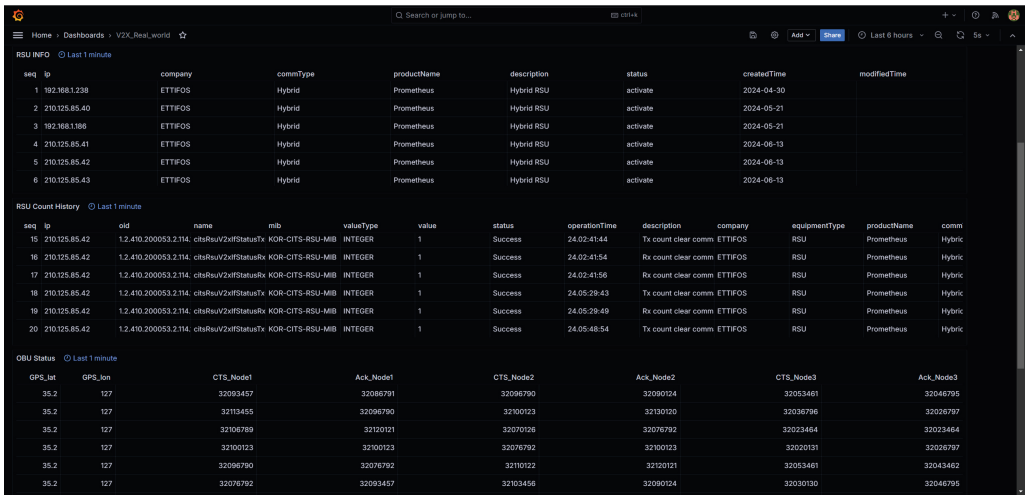

Figure 19. V2X data visualization in real-world driving.

The Grafana-powered visualization in Figure 19 provides operators with an interactive, data-rich view of the initial conditions of the simulation, setting the stage for dynamic analysis of real-time interactions and the responsive capabilities of the system in the V2X-Car Edge Cloud environment.

Figure 20 shows the real-time object detection results in the virtual driving simulation based on the data collected by the cameras mounted on the Data IO Boxes. In this visualization, the YOLOv3 algorithm identifies and classifies various entities, including pedestrians, vehicles, and other objects in the environment, providing essential information about the locations and identities of the objects as the simulation unfolds.

The detected objects are displayed with bounding boxes and labels, allowing operators to continuously monitor the locations and interactions of objects. This capability is vital for assessing the ability of the proposed system to maintain situational awareness in dynamic

scenarios. The movement of each detected object and its interaction with other entities, such as the vehicle and SmartX Pole, are updated in real time, providing a comprehensive view of the environment.

This frame captures a key moment prior to the virtual vehicle reaching the intersection where an injured pedestrian is positioned. This initial detection allows the proposed system to proactively respond to emergencies as they occur. By integrating object detection into the running simulation, the V2X-Car Edge Cloud can dynamically assess and react to unfolding events, demonstrating its effectiveness in handling complex scenarios within a real-time framework.

This real-time visualization, supported by Grafana, provides an intuitive and data-rich interface for operators, allowing them to track and respond to critical developments in the virtual environment. This capability is essential for evaluating the responsiveness and reliability of the V2X-Car Edge Cloud system, especially in scenarios where split-second decisions are crucial for safety.

Figure 20. Real-time object detection during virtual driving simulation.

Figure 21 illustrates the total processing time required for each frame captured by the Data IO Boxes during the virtual driving simulation. Each bar represents the end-to-end processing time of a single frame from capture to output. The red vertical markers indicate 50 ms intervals, providing a reference for assessing processing delays in real time.

Frames processed within 50 ms are displayed in bright blue, whereas frames processed within a time range above this 50 ms threshold are highlighted in red, highlighting processing delays that exceed the target limit. This color coding allows for a quick visual assessment of the performance of the system, emphasizing instances where frame processing times may impact the overall real-time responsiveness of the V2X-Car Edge Cloud.

This visualization offers a clear, cumulative view of frame-by-frame processing times, which is essential for evaluating the ability of the system to meet low-latency requirements under continuous data input conditions.

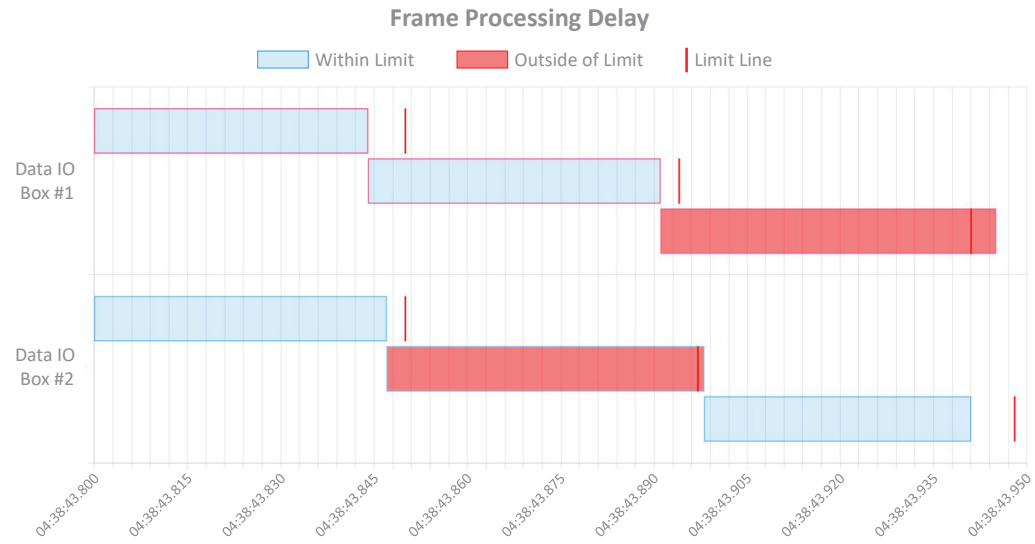

Figure 21. Frame processing delay.

Figure 22 provides a detailed breakdown of the processing delays across individual stages within the frame processing workflow for the Data IO Boxes. Unlike Figure 21, which presents the total processing time per frame, Figure 22 tracks each of the five stages of Frame Collection, Inference, Collage Frames, Post Processing, and Tracking Objects, illustrating the duration of each stage in real-time processing.

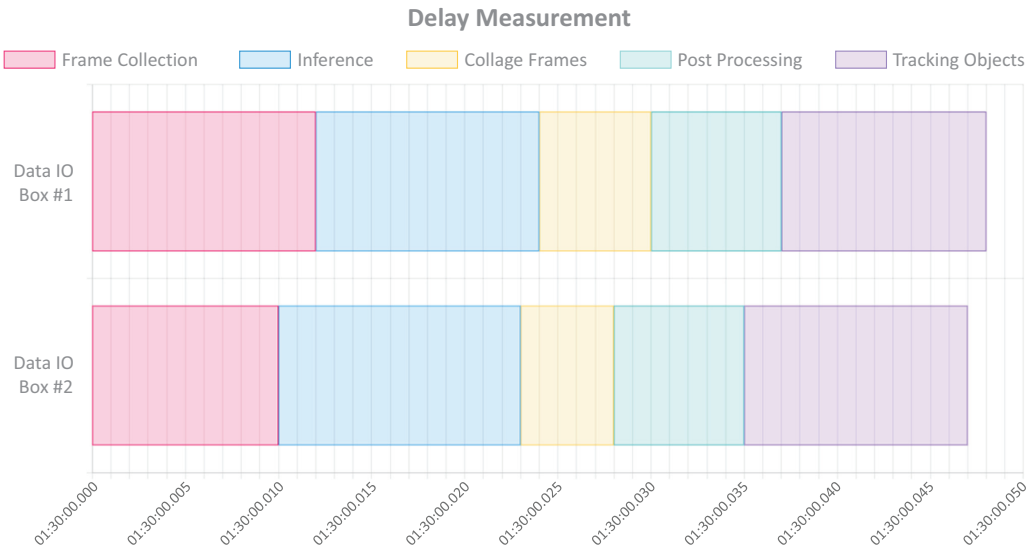

Figure 22. Delay Visualization Tool (DVT) for internal process tracking of two Data IO Boxes.

The Delay Visualization Tool (DVT) monitors each stage as follows:

1. Frame Collection: Captures video frames from cameras attached to the Data IO Boxes, marking the start of the processing workflow.
2. Inference: Runs the YOLOv3 model on the collected frames to detect objects, a crucial step in real-time object recognition.
3. Collage Frames: Combines processed frames into a cohesive view, which aids in maintaining situational awareness.
4. Post Processing: Refines the results from Inference, ensuring accurate object classifications and bounding box placements.
5. Tracking Objects: Monitors detected objects across frames, maintaining continuity as objects move through the environment.

The DVT color-codes delays that exceed set thresholds, highlighting the stages where optimization may be required to maintain the low-latency goals of the V2X-Car Edge Cloud system. This tool enables targeted adjustments, allowing the system to meet real-time responsiveness requirements by minimizing bottlenecks at each stage of the frame processing pipeline.

Figure 23 presents a table that records the time required to process individual frames during the virtual driving simulation. Each entry in the table represents the total time required for a frame to complete the processing sequence, from the initial capture to the output. On average, each frame requires approximately 33.954 ms to process, indicating that the proposed system generally maintains low-latency performance within the 50 ms target threshold.

Although most frames are processed within the 50 ms target, a few entries exceed this threshold, highlighting potential fluctuations in the processing time. These outliers are valuable for identifying areas where the proposed system can be optimized to improve consistency. However, the majority of frames remain within the acceptable range, demonstrating the ability of the V2X-Car Edge Cloud system to efficiently handle real-time data.

These data provide a critical benchmark for evaluating the overall efficiency of the system, ensuring that it meets the stringent performance requirements essential for real-time object detection and seamless simulation. By monitoring these processing times, the proposed system can be adjusted as required to further enhance responsiveness and maintain optimal performance across continuous data input.

(Unit: ms)

Frame No.	Delay	Frame No.	Delay		Frame No.	Delay
1	1.205246	11	4.101602		8991	188.7031
2	1.193148	12	77.90123		8992	9.673403
3	13.25596	13	3.291809		8993	11.49037
4	3.188465	14	15.08583		8994	188.1614
5	3.189048	15	72.51938	...	8995	10.63265
6	38.93290	16	3.285554		8996	15.11878
7	4.577168	17	24.86970		8997	165.0257
8	3.460401	18	68.60888		8998	21.55534
9	69.18519	19	3.519186		8999	14.38788
10	4.516572	20	27.36045		9000	169.7277
Average Delay: 33.954 ms						

Figure 23. The result of delay measurement for first 9000 frames.

Figures 24 and 25 depict the application of onion-ring visualization to monitor the networking flow in the V2X-Car Edge Cloud system. This visualization provides a layered and hierarchical view of the real-time network activity of the system, facilitating the identification and analysis of communication flows between key components such as the AI Computing and Storage Boxes, Data IO Boxes, and Hybrid-V2X Mobile Station.

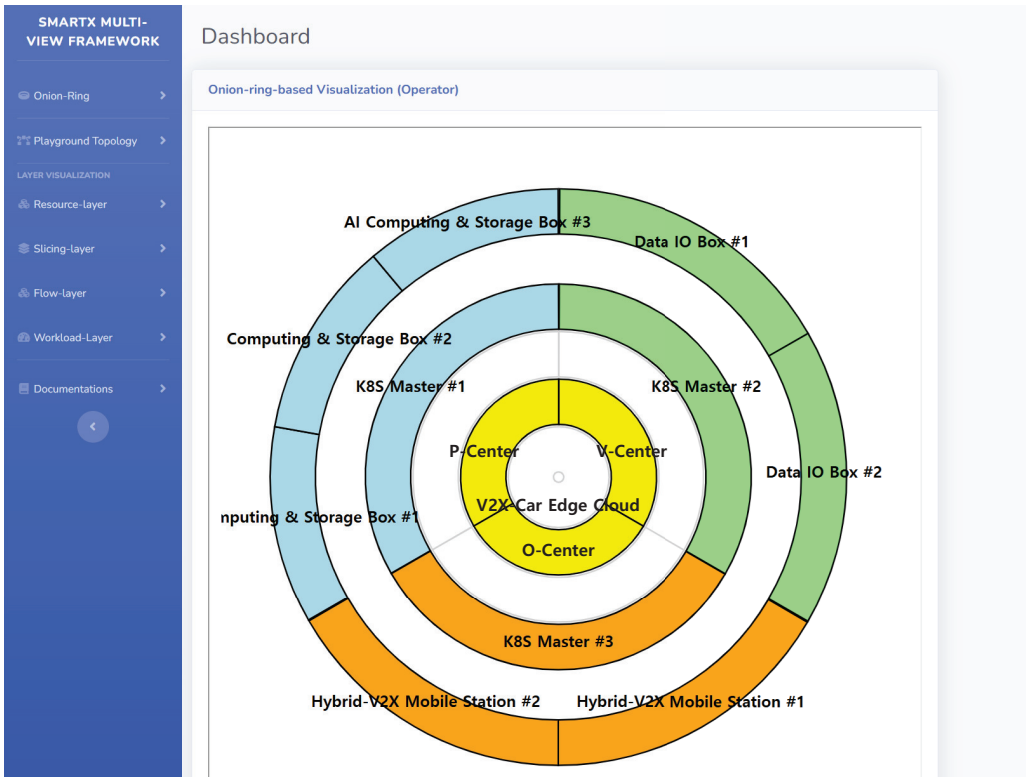

Figure 24. The implemented result of 2D onion-ring visualization for box layer visibility.

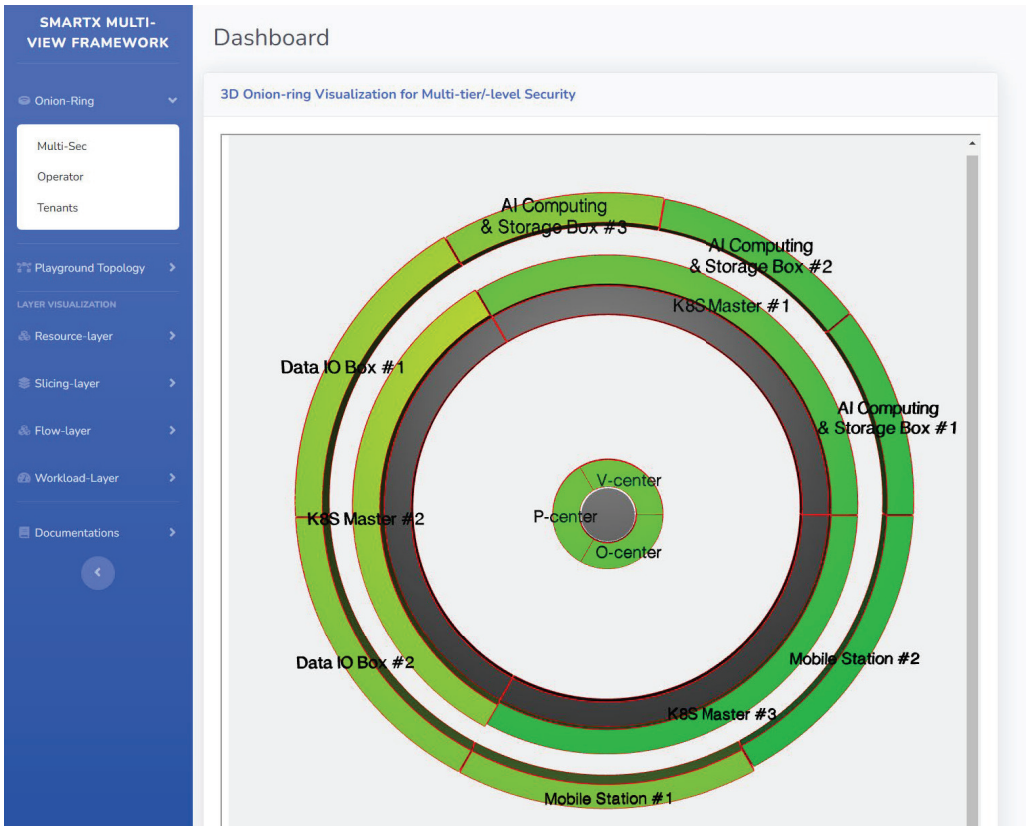

Figure 25. The implemented result of 3D onion-ring visualization for networking flow visibility.

In the displayed scenario, the onion-ring visualization highlights the flow of network traffic during a relatively balanced state, with no significant congestion or bottlenecks observed. Each concentric ring represents a specific layer in the network, with inner

rings corresponding to high-priority traffic and outer rings indicating auxiliary or low-priority flows. Although the figure demonstrates a calm state of network activity, it also showcases the ability of the visualization tool to track and map interactions across multiple nodes in real time. This tool is particularly useful for identifying potential anomalies, optimizing data flow, and ensuring that the proposed system operates efficiently under various traffic conditions. Visualization is integral to maintaining system performance, providing a proactive method for managing and balancing network resources. By offering an intuitive and dynamic representation of the operational state of the network, onion-ring visualization enhances situational awareness and supports real-time decision making within the proposed V2X-Car Edge Cloud framework.

In summary, the proposed framework supports self-driving AI services through object recognition and V2X communication (refer to Figures 18–20). The average data processing latency is approximately 33 ms, as shown in Figure 23. In a similar study [10], the latency between the vehicle and the cloud was between 27 and 74 ms, indicating that the latency of our framework is acceptable in autonomous driving. Unlike other studies that focused solely on object detection in V2X systems, our framework provides additional functionalities. It features the Delay Visualization Tool that measures latency for each component, making it easier to catch the bottleneck. Furthermore, the state of the edge cloud can be monitored using 3D onion-ring visualization combined with DevSecOps automation. Lastly, the SiLS-Based Service Realization enables the simulation of complex scenarios that are difficult to verify in practice.

8. Conclusions and Future Work

In this study, we proposed the V2X-Car Edge Cloud, a cloud-native platform designed to address challenges in real-time processing, scalability, and security within latency-sensitive V2X applications. By integrating DevSecOps automation principles, cloud-native Kubernetes orchestration, and hybrid communication protocols (C-V2X, 5G, WAVE), the system demonstrated secure deployment, dynamic resource allocation, and real-time monitoring capabilities across decentralized edge environments. In Section 7, we validated object detection and cloud infrastructure management through a specific scenario, which revealed improvements in latency, resource management, and operational stability. Conducted in a virtual driving simulation environment, this validation utilized NVIDIA Jetson AGX Orin Developer Kits and Supermicro SYS-210P servers. Tools such as LTTng tracing and the DeepStream SDK facilitated data flow monitoring and system performance optimization, while YOLOv3 models deployed in Data IO Boxes enabled real-time object detection and classification. Moreover, the AI Computing and Storage Boxes supported cloud storage, result aggregation, and further data analysis.

The proposed V2X-Car Edge Cloud interconnects real and virtual driving environments, enabling cost-effective validation of autonomous driving services through software-in-the-loop simulations. This platform is anticipated to increase network transparency and data analysis in V2X settings, potentially lowering costs for automotive manufacturers and telecom providers while improving system reliability. It includes tools for monitoring latency during real-time object detection and managing network traffic bottlenecks, addressing the needs of autonomous vehicles. Additionally, the edge-managed V2X system is poised to support the development of ITSs and could be applied to transportation and logistics sectors. By optimizing data management in edge-core cloud environments, it is expected to aid in standardizing real-time data analysis and management across various industries.

However, the study has several limitations. First, there is a dependency on the hardware set outlined in Section 5. While most functions are containerized, requiring

less specific hardware, certain specialized devices are necessary, such as the PreScan box requiring a driving wheel and the SmartX Pole equipped with V2X modules. Using hardware other than the Data IO Box, which is specialized for image processing, may impede the achievement of the same data processing speeds observed in our results. Moreover, our validation did not include tests in extreme environments. For instance, although the proposed system shows consistent performance in simulated environments with predefined workloads, further analysis is needed to understand how the system performs under high data influx, network congestion, and security attack scenarios. This means we cannot fully assess the system's resilience in extreme conditions yet.

To address these gaps, future work will introduce quantitative performance benchmarks for scalability, latency, and security across various operational scenarios, including high-load stress tests, edge node failures, and security breach simulations. Extensive real-world testing will be conducted with physical vehicles and distributed edge nodes to validate the resilience of the proposed system in uncontrolled environments. In addition, efforts will be made to improve hardware compatibility, optimize network synchronization protocols, and refine AI decision transparency to meet large-scale deployment demands.

Author Contributions: Conceptualization, S.P. and J.K.; Funding acquisition, J.K.; Investigation, H.Z.; Project administration, S.P. and J.K.; Software, D.K. and A.Y.; Supervision, D.K., H.Z., and J.K.; Validation, D.K. and A.Y.; Visualization, D.K. and A.Y.; Writing—original draft, H.Z.; Writing—review and editing, H.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a Korea Institute for Advancement of Technology (KIAT) grant funded by the Korean Government (MOTIE) (P0020535, The Competency Development Program for Industry Specialist), and this work was supported by an IITP grant funded by the Korean Government (MSIT) (No.2019-0-01842, Artificial Intelligence Graduate School Program (GIST).

Data Availability Statement: The dataset is available on request from the authors.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen, S.; Hu, J.; Zhao, L.; Zhao, R.; Fang, J.; Shi, Y.; Xu, H. *Cellular Vehicle-to-Everything (C-V2X)*; Springer: Berlin/Heidelberg, Germany, 2023. [CrossRef]
2. Stoma, M.; Dudziak, A.; Caban, J.; Droździel, P. The Future of Autonomous Vehicles in the Opinion of Automotive Market Users. *Energies* **2021**, *14*, 4777. [CrossRef]
3. Cabeza, L.F.; Verez, D.; Teixidó, M. Hardware-in-the-Loop Techniques for Complex Systems Analysis: Bibliometric Analysis of Available Literature. *Appl. Sci.* **2023**, *13*, 8108. [CrossRef]
4. Avcı, l.; Koca, M. Intelligent Transportation System Technologies, Challenges and Security. *Appl. Sci.* **2024**, *14*, 4646. [CrossRef]
5. Liu, S.; Liu, L.; Tang, J.; Yu, B.; Wang, Y.; Shi, W. Edge computing for autonomous driving: Opportunities and challenges. *Proc. IEEE* **2019**, *107*, 1697–1716. [CrossRef]
6. Georgiades, M.; Poullas, M.S. Emerging Technologies for V2X Communication and Vehicular Edge Computing in the 6G era: Challenges and Opportunities for Sustainable IoV. In Proceedings of the 2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT), Pafos, Cyprus, 19–21 June 2023; pp. 684–693. [CrossRef]
7. Abdel Hakeem, S.A.; Hady, A.A.; Kim, H. 5G-V2X: Standardization, architecture, use cases, network-slicing, and edge-computing. *Wirel. Netw.* **2020**, *26*, 6015–6041. [CrossRef]
8. Rajapakse, R.N.; Zahedi, M.; Babar, M.A.; Shen, H. Challenges and solutions when adopting DevSecOps: A systematic review. *Inf. Softw. Technol.* **2022**, *141*, 106700. [CrossRef]
9. Lumpatki, S.S.; Patwardhan, S.; Kulkarni, M. Implementing 'DevSecOps as a Culture'—The Concept, Benefits, Execution Strategies, and Challenges. In Proceedings of the Smart Trends in Computing and Communications, Pune, India, 12–13 January 2024; pp. 189–197. [CrossRef]
10. Hawlader, F.; Robinet, F.; Frank, R. Leveraging the edge and cloud for V2X-based real-time object detection in autonomous driving. *Comput. Commun.* **2024**, *213*, 372–381. [CrossRef]

11. Zhang, X.; Cao, X.; Zhang, H.; Shen, Y.; Yuan, X.; Cui, Z.; Lu, Z. An Intelligent Obstacle Detection for Autonomous Mining Transportation With Electric Locomotive via Cellular Vehicle-to-Everything and Vehicular Edge Computing. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 3177–3190. [CrossRef]
12. Arin, J.; Velez, G.; Bustamante, P. A C-ITS Architecture for MEC and Cloud Native Back-End Services. *IEEE Access* **2024**, *12*, 64531–64550. [CrossRef]
13. Xiong, Y.; Sun, Y.; Xing, L.; Huang, Y. Extend cloud to edge with kubeedge. In Proceedings of the 2018 IEEE/ACM Symposium On Edge Computing (SEC), Seattle, WA, USA, 25–27 October 2018; pp. 373–377. [CrossRef]
14. Toka, L.; Dobreff, G.; Fodor, B.; Sonkoly, B. Machine learning-based scaling management for kubernetes edge clusters. *IEEE Trans. Netw. Serv. Manag.* **2021**, *18*, 958–972. [CrossRef]
15. Ma, H.; Li, S.; Zhang, E.; Lv, Z.; Hu, J.; Wei, X. Cooperative autonomous driving oriented MEC-aided 5G-V2X: Prototype system design, field tests and AI-based optimization tools. *IEEE Access* **2020**, *8*, 54288–54302. [CrossRef]
16. Sandu, C.; Susnea, I. Edge computing for autonomous vehicles-a scoping review. In Proceedings of the 2021 20th RoEduNet Conference: Networking in Education and Research (RoEduNet), Iasi, Romania, 4–6 November 2021; pp. 1–5. [CrossRef]
17. Shin, J.S.; Kim, J. SmartX Multi-Sec: A Visibility-Centric Multi-Tiered Security Framework for Multi-Site Cloud-Native Edge Clusters. *IEEE Access* **2021**, *9*, 134208–134222. [CrossRef]
18. Kanagachidambaresan, G.R. *Role of Edge Analytics in Sustainable Smart City Development: Challenges and Solutions*; Wiley: Hoboken, NJ, USA, 2020.
19. Yang, X.; Shi, Y.; Xing, J.; Liu, Z. Autonomous driving under V2X environment: State-of-the-art survey and challenges. *Intell. Transp. Infrastruct.* **2022**, *1*, liac020. [CrossRef]
20. Bréhon-Grataloup, L.; Kacimi, R.; Beylot, A.L. Mobile edge computing for V2X architectures and applications: A survey. *Comput. Netw.* **2022**, *206*, 108797. [CrossRef]
21. Vladyko, A.; Khakimov, A.; Muthanna, A.; Ateya, A.A.; Koucheryavy, A. Distributed edge computing to assist ultra-low-latency VANET applications. *Future Internet* **2019**, *11*, 128. [CrossRef]
22. Kawalpreet, K.; Salaria, A.; Kumar, J.; Kaur, A. Edge computing technologies: A comprehensive review and future perspectives. In Proceedings of the AIP Conference Proceedings, Mohali, India, 12–13 October 2023; Volume 3121. [CrossRef]
23. Lim, E.H.; Chai, T.Y.; ap Muniandy, M.; Yong, T.F.; Ooi, B.Y.; Lin, J.M. Edge Computing and AI for IoT: Opportunities and Challenges. In Proceedings of the 2023 International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan), PingTung, Taiwan, 17–19 July 2023; pp. 357–358. [CrossRef]
24. Soto, I.; Calderon, M.; Amador, O.; Urueña, M. A survey on road safety and traffic efficiency vehicular applications based on C-V2X technologies. *Veh. Commun.* **2022**, *33*, 100428. [CrossRef]
25. Guim, F.; Metsch, T.; Moustafa, H.; Verrall, T.; Carrera, D.; Cadenelli, N.; Chen, J.; Doria, D.; Ghadie, C.; Prats, R.G. Autonomous lifecycle management for resource-efficient workload orchestration for green edge computing. *IEEE Trans. Green Commun. Netw.* **2021**, *6*, 571–582. [CrossRef]
26. Duan, S.; Wang, D.; Ren, J.; Lyu, F.; Zhang, Y.; Wu, H.; Shen, X. Distributed artificial intelligence empowered by end-edge-cloud computing: A survey. *IEEE Commun. Surv. Tutorials* **2022**, *25*, 591–624. [CrossRef]
27. Myrbakken, H.; Colomo-Palacios, R. DevSecOps: A multivocal literature review. In Proceedings of the Software Process Improvement and Capability Determination: 17th International Conference, SPICE 2017, Palma de Mallorca, Spain, 4–5 October 2017; pp. 17–29. [CrossRef]
28. Abkenar, F.S.; Ramezani, P.; Iranmanesh, S.; Murali, S.; Chulerttiyawong, D.; Wan, X.; Jamalipour, A.; Raad, R. A survey on mobility of edge computing networks in IoT: State-of-the-art, architectures, and challenges. *IEEE Commun. Surv. Tutorials* **2022**, *24*, 2329–2365. [CrossRef]
29. Patwary, M.; Ramchandran, P.; Tibrewala, S.; Lala, T.; Kautz, F.; Coronado, E.; Riggio, R.; Ganugapati, S.; Ranganathan, S.; Liu, L. Edge Services and Automation. In Proceedings of the 2022 IEEE Future Networks World Forum (FNWF), Montreal, QC, Canada, 10–14 October 2022; pp. 1–49. [CrossRef]
30. Christopoulou, M.; Barmpounakis, S.; Koumaras, H.; Kaloxylos, A. Artificial Intelligence and Machine Learning as key enablers for V2X communications: A comprehensive survey. *Veh. Commun.* **2023**, *39*, 100569. [CrossRef]
31. Nain, G.; Pattanaik, K.; Sharma, G. Towards edge computing in intelligent manufacturing: Past, present and future. *J. Manuf. Syst.* **2022**, *62*, 588–611. [CrossRef]
32. Gill, S.S.; Golec, M.; Hu, J.; Xu, M.; Du, J.; Wu, H.; Walia, G.K.; Murugesan, S.S.; Ali, B.; Kumar, M.; et al. Edge AI: A taxonomy, systematic review and future directions. *Clust. Comput.* **2025**, *28*, 18. [CrossRef]
33. Abreha, H.G.; Hayajneh, M.; Serhani, M.A. Federated learning in edge computing: A systematic survey. *Sensors* **2022**, *22*, 450. [CrossRef] [PubMed]
34. Faticanti, F.; Santoro, D.; Cretti, S.; Siracusa, D. An application of kubernetes cluster federation in fog computing. In Proceedings of the 2021 24th Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), Paris, France, 1–4 March 2021; pp. 89–91. [CrossRef]

35. Carrión, C. Kubernetes scheduling: Taxonomy, ongoing issues and challenges. *ACM Comput. Surv.* **2022**, *55*, 1–37. [CrossRef]
36. Kayal, P. Kubernetes in fog computing: Feasibility demonstration, limitations and improvement scope. In Proceedings of the 2020 IEEE 6th World Forum on Internet of Things (WF-IoT), New Orleans, LA, USA , 2–16 June 2020; pp. 1–6. [CrossRef]
37. Yasar, H.; Teplov, S.E. Devsecops in embedded systems: An empirical study of past literature. In Proceedings of the 17th International Conference on Availability, Reliability and Security, Vienna, Austria, 23–26 August 2022; pp. 1–6. [CrossRef]
38. Srinivasa, R.; Naidu, N.K.S.; Maheshwari, S.; Bharathi, C.; Kumar, A.H. Minimizing latency for 5G multimedia and V2X applications using mobile edge computing. In Proceedings of the 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), Jaipur, India, 28–29 September 2019; pp. 213–217. [CrossRef]
39. Wang, W.; Guo, K.; Cao, W.; Zhu, H.; Nan, J.; Yu, L. Review of Electrical and Electronic Architectures for Autonomous Vehicles: Topologies, Networking and Simulators. *Automot. Innov.* **2024**, *7*, 82–101. [CrossRef]
40. Shah, G.; Saifuddin, M.; Fallah, Y.P.; Gupta, S.D. Rve-cv2x: A scalable emulation framework for real-time evaluation of cv2x-based connected vehicle applications. In Proceedings of the 2020 IEEE Vehicular Networking Conference (VNC), New York, NY, USA, 16–18 December 2020; pp. 1–8. [CrossRef]
41. Tong, W.; Hussain, A.; Bo, W.X.; Maharjan, S. Artificial intelligence for vehicle-to-everything: A survey. *IEEE Access* **2019**, *7*, 10823–10843. [CrossRef]
42. Nouhas, H.; Belangour, A.; Nassar, M. Cloud and Edge Computing Architectures: A Survey. In Proceedings of the 2023 IEEE 11th Conference on Systems, Process & Control (ICSPC), Malacca, Malaysia, 16 December 2023; pp. 210–215. [CrossRef]
43. Sánchez-Gordón, M.; Colomo-Palacios, R. Security as culture: A systematic literature review of DevSecOps. In Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops, Seoul, Republic of Korea, 27 June–19 July 2020; pp. 266–269. [CrossRef]
44. Hemmati, A.; Raoufi, P.; Rahmani, A.M. Edge artificial intelligence for big data: A systematic review. *Neural Comput. Appl.* **2024**, *36*, 11461–11494. [CrossRef]
45. Wang, J.; Zhu, Y. A Hardware-in-the-Loop V2X Simulation Framework: CarTest. *Sensors* **2022**, *22*, 5019. [CrossRef]
46. Gelbal, Ş.Y.; Tamilarasan, S.; Cantaş, M.R.; Güvenç, L.; Aksun-Güvenç, B. A connected and autonomous vehicle hardware-in-the-loop simulator for developing automated driving algorithms. In Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017; pp. 3397–3402. [CrossRef]
47. Ma, J.; Zhou, F.; Huang, Z.; James, R. Hardware-in-the-loop testing of connected and automated vehicle applications: A use case for cooperative adaptive cruise control. In Proceedings of the 2018 21st International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2878–2883. [CrossRef]