Mechatronics and Intelligent Transportation Systems
https://www.acadlore.com/journals/MITS

Enhancing Performance and Reducing Latency in Autonomous Systems Through Edge Computing for Real-Time Data Processing

Anil Kumar Pallikonda[1*], Vinay Kumar Bandarapalli[1], Vipparla Aruna[2]

[1] Department of Computer Science and Engineering, PVP Siddhartha Institute of Technology, 520001 Vijayawada, India
[2] Department of Computer Science and Engineering, NRI Institute of Technology, 520001 Vijayawada, India

* Correspondence: Anil Kumar Pallikonda (anilkumar.pallikonda@gmail.com)

Received: 07-22-2025 **Revised:** 09-03-2025 **Accepted:** 09-16-2025

Citation: A. K. Pallikonda, V. K. Bandarapalli, and V. Aruna, "Enhancing performance and reducing latency in autonomous systems through edge computing for real-time data processing," *Mechatron. Intell Transp. Syst.*, vol. 4, no. 3, pp. 154–165, 2025. https://doi.org/10.56578/mits040305.

Abstract: The integration of edge computing for real-time data processing in autonomous systems has been identified as a promising solution to mitigate the performance bottlenecks and latency associated with traditional cloud-based models. Autonomous systems, including vehicles, drones, and robotics, rely heavily on quick data analysis to make timely decisions. However, cloud computing, with its inherent data transmission delays, hinders the responsiveness and efficiency of these systems. To address these challenges, edge computing is proposed as a means to process data locally, at the point of use, thus enabling faster decision-making processes and reducing data transfer overhead. This approach leverages distributed machine learning for decision-making and dynamic resource allocation to balance computational tasks between edge and cloud resources. Through extensive experimentation, it has been demonstrated that the edge computing paradigm can reduce latency by up to 65%, offering substantial improvements in both energy efficiency and data processing speed when compared to traditional cloud-based methods. Furthermore, the proposed system outperforms existing alternatives in terms of computational speed, reliability, and energy consumption. The introduction of an Edge Computing Decision Model (ECDM) and a Dynamic Resource Allocation Algorithm (DRAA) is shown to optimize system performance by balancing computational demands between local edge nodes and remote cloud servers. These innovations enable autonomous systems to function more effectively and efficiently, even in resource-constrained environments. This study highlights the importance of integrating edge computing into autonomous system architectures to meet the growing demand for low-latency, high-performance applications. The potential of edge computing to significantly enhance the reliability and operational capacity of autonomous systems has been established, paving the way for more reliable and scalable solutions in real-time environments.

Keywords: Edge computing; Real-time data processing; Autonomous systems; Latency reduction; Resource allocation; Machine learning

1 Introduction

Advances in autonomous systems, including self-driving vehicles, drones, and industrial robots, as well as the development of smart cities, are anticipated to continue shaping the technological landscape. These systems are required to process extensive sensor data from various sources, such as cameras, LiDAR, radar, GPS, and environmental sensors, in order to make informed and timely decisions [1]. The speed and accuracy of data processing in such systems are critical for ensuring both safety and optimal performance across a range of operational contexts [2]. For instance, rapid decision-making in autonomous systems can prevent accidents by enabling the avoidance of obstacles, such as pedestrians, or by adhering to newly established traffic rules enforced by traffic signals. Such prompt actions are fundamental to fostering safe driving conditions and preventing incidents on the road [3].

The present cloud solutions for autonomous cars are unable to react quickly to situations because they rely on a slow method for sharing information [4]. Although cloud infrastructure offers considerable room for data storage, data is transmitted over long distances to a central data center, resulting in lengthy delays that are unsuitable for urgent applications such as autonomous driving and industrial automation [5]. It becomes evident when automated

systems must respond quickly to keep everyone safe and the system performing well within a limited period [6]. So, we ought to seek out options for data storage other than the cloud that are both flexible and popular [7].

Edge computing is being recognized as an effective method to resolve the problems surrounding cloud computing's latency and performance. Edge computing can lower latency and boost real-time processing by carrying out data processing very close to where the data is collected [8]. Using edge computing in autonomous systems, data is processed locally, allowing devices to make decisions faster and making them more reliable [9]. Recently, several studies have highlighted the benefits of edge computing for self-driving cars, industrial IoT, healthcare, and robotics, as it reduces latency, enhances efficiency, and enables more effective resource utilization [10, 11].

Although edge computing is beneficial, it faces significant hurdles, including limited resources on these devices, the diversity of network types, and the need for coordination between the edge and cloud [12]. Additionally, the use of ML at the edge is an area that researchers are focusing on, as it supports better decision-making [13]. Still, these models must be built in a way that utilizes the least amount of edge computer resources while still meeting the needs of real-time applications [14]. Thus, it is essential to have algorithms, cloud systems, and technologies that make use of both edge and cloud resources together [15]. Despite advancements, current research lacks systematic approaches for synchronizing dynamic resource allocation across edge and cloud nodes and for adapting models to rapidly changing, complex scenarios. This study bridges these gaps by proposing mechanisms that dynamically allocate resources and maintain reliable performance under diverse real-world conditions.

It aims to investigate whether edge computing can improve the speed of data processing and enhance the performance of autonomous systems by reducing delays. The primary objective is to develop a method that integrates edge computing and machine learning models to improve the speed, efficiency, and scalability of autonomous technology. Our new method helps determine which tasks should be sent to edge computers and which should remain cloud-based, depending on various situations. There is solid evidence that these models are effective, as demonstrated by simulation tests that focus on autonomous vehicles and robotics in the industry.

It addresses various new aspects related to autonomous systems and edge computing. Rather than relying on the conventional cloud strategy, our first step uses a new model that connects edge computing with machine learning. We suggested a special approach to distributing resources that provide help when systems require it, conserving power and reducing unnecessary delays. In addition, the simulation part of the paper reveals that the edge computing framework is preferred over the cloud, as it offers faster response times and makes decisions more accurately. They help lead autonomous systems to function correctly and successfully in the real world.

The specific objectives are to (i) reduce average latency by at least 50%, (ii) improve action prediction accuracy beyond 90%, and (iii) lower energy consumption per task by a minimum of 20% compared to cloud-only systems.

Organization of the Paper

Section 2 analyzes previous work on edge computing for autonomous systems, focusing on its implementation, the challenges it presents, and the role played by machine learning. The proposed approach is outlined in Section 3, utilizing edge computing ecosystems, selected data collections, and various models and algorithms tailored to them. In Section 4, the authors conduct testing and examine issues such as speed, data transmission, and accuracy while also measuring the energy required by their model. Ultimately, the findings are summarized, highlighting the shortcomings and recommending improved approaches for future use.

2 Related Work

Experts in the field are paying growing interest to edge computing for autonomous systems because it provides solutions to problems linked to latency and performance. Because many of these systems utilize real-time data, edge computing's rapid information processing is particularly beneficial for them. Additionally, developing effective strategies for leveraging edge computing in various components of autonomous systems is necessary. This section examines research on the application of edge computing in autonomous systems, focusing on key strategies that enhance the system's performance, minimize delays, and utilize resources efficiently.

Initially, edge computing in autonomous systems was explored by researchers for use in autonomous vehicles. Scientists investigated how edge computing facilitates the detection of objects and enables real-time decision-making in self-driving cars [16]. The report stated that simple analytics should be carried out at the point of connection, while cloud servers handle more complex operations, such as determining the most efficient routes. Along with reduced latency, this technique enables the computer to perform more computations. Nonetheless, the study did not examine how rapid changes in traffic could impact the movement of data from one location to another, which plays a crucial role in self-driving vehicles.

Experts have worked diligently to develop strategies for utilizing resources and assigning tasks in edge computing for autonomous systems. Lin et al. [17] explain that functions should be assigned to edge devices according to how many resources an edge node has and the current situation of the network. Machine learning is applied in a model to predict the necessary tasks and determine the optimal schedule for completing them. It minimized the amount of time spent on each task, but it did not provide sufficient attention to the synchronization of edge devices required to

operate the system effectively.

Edge computing is being explored by businesses to enable faster, on-the-spot action. Wang et al. state that this platform connects the edges to follow the readings from both vision and force sensors [18]. Having data processing done makes robots react faster and make fewer errors. All the same, the plan relies on fixed network directions and does not account for edge devices or changes that occur in dynamic networks.

It is essential to learn how to adapt machine learning algorithms for use on edge devices. Liang et al. [19] developed a method that integrates edge computing and machine learning to process data in autonomous systems efficiently. They allowed devices at the edge to use light models so the time taken, and resources consumed could be reduced. The framework performed well, but it became clear that it is challenging to improve machine learning models for devices that lack sufficient computing power. It was determined that while having low latency benefits edge devices, reducing the accuracy in favor of speed is their primary challenge.

Lu et al. [20] wrote about the problems that occur in edge-based autonomous systems because of limited resources. They developed a method to reduce power consumption in edge devices by optimizing the use of available resources. Still, the approach did not guarantee that energy efficiency would not impact the system's performance, especially in situations where performance is crucial now.

People are now trying to combine edge and cloud computing to address the challenge of limited resources on edge devices. Hemmati et al. [21] suggested using an edge-cloud system, where real-time tasks are performed at the edge, and the cloud handles more detailed operations. According to what they discovered, running edge and cloud together eliminates performance problems; still, more information about handling load and resources is needed for practical results.

People have investigated edge computing when dealing with multi-agent systems and collaborative robotics. Zhang et al. [22] recommended using edge computing on AMRs to ensure that robots coordinate in real-time with others in their environment. They relied on edge devices to facilitate prompt communication between the robots and enable them to work more efficiently together. This method demonstrated that edge computing can help robot systems function more smoothly. It was unclear how these systems would handle more complex situations involving multiple robots and a large amount of information.

Autonomous systems with edge computing can help healthcare staff speedily monitor patient health and make informed medical decisions. According to Munir [23], edge computing is utilized to facilitate the analysis of health data by enabling wearable sensors to transfer patient information to nearby devices for instant analysis. With the system, healthcare workers received real-time advice, which helped them expedite the diagnosis process. Still, the evaluation did not address how controls could be implemented to ensure that sensitive health data was safe and private.

Autonomous systems operating on the edge have also been tested in dynamic environments. Martini et al. [24] explained that in their hybrid system, the cloud performed the longer-term planning while the devices at the edge processed real-time information for self-driving cars. It was proven that this technique significantly decreased the time required for data processing, most noticeably in crowded environments. However, the research did not account for issues such as bandwidth and network problems that could strongly influence system performance.

Edge computing is being considered a solution for rapid communication in autonomous systems across various IoT-related applications. In their study, Biswas and Wang [25] investigated the application of edge computing in IoT and autonomous systems to minimize communication delays between edge devices and the central system. Although the results demonstrated a significant reduction in the time it took for data to reach the servers, the study did not offer much insight into the impact of network changes on large IoT systems.

Finally, a study conducted by Inamdar et al. [26] examined how deep learning models can aid in making urgent decisions in autonomous vehicles. With their new framework, autonomous vehicles can operate independently, making decisions without requiring additional assistance from cloud services. The study demonstrated that working with deep learning models near the sensor speeds up data processing and saves resources. However, it remains a challenge to run complex models on these devices.

To address the limitations highlighted in earlier studies, this research introduces a hybrid edge–cloud framework that directly responds to issues such as unsynchronized dynamic resource allocation and limited adaptability in complex environments. This transition establishes the motivation for the methodology presented in the next section.

3 Methodology

In the following part, we describe how edge computing and real-time data processing can be used in autonomous systems. The system architecture, the data to be used, the mathematical equations, and the algorithms required for optimal performance and reduced latency are all parts of it. We provide sufficient information so that someone else can replicate the study with similar results. The system relies on deploying edge nodes to handle critical tasks immediately and utilizes machine learning to process data from sensors in real-time.

3.1 Dataset

This work utilizes sensor data collected from an autonomous vehicle operating in various real-life contexts. LiDAR, RGB cameras, and GPS sensors are used together to generate the data found in autonomous driving systems. The data was split into training, validation, and testing sets, allowing the models to be both trained and evaluated.

Dataset Details

Data Source: Autonomous vehicle data captured in urban and suburban environments.

Sensors Used:

LiDAR data (points per frame: 500,000 points)

RGB camera frames (resolution: 1920×1080)

GPS coordinates (latitude, longitude, altitude)

IMU data (accelerometer and gyroscope data)

Parameters:

Temporal Resolution: 30 Hz (sensor data collected at 30 frames per second)

Size of Dataset: 50,000 samples for training, 10,000 for validation, and 5,000 for testing

Types of Data:

LiDAR: 3D point clouds with object recognition labels

RGB Camera: Raw images with object detection bounding boxes and labels

GPS: GPS coordinates paired with time stamps

IMU: Acceleration and angular velocity data

The dataset covered both daytime and nighttime driving across sunny, rainy, and foggy weather conditions, capturing variable traffic densities from light suburban roads to dense urban intersections. Data were collected continuously over three months and made publicly available through a controlled-access repository to ensure reproducibility.

Table 1. Comparison of existing approaches for road sign and gesture recognition

Parameter	LiDAR	RGB Camera	GPS	IMU
Resolution	500,000 points per frame	1920×1080 pixels	Latitude, Longitude, Altitude	X, Y, Z acceleration ($\mathrm{m/s^2}$)
Frequency	30 Hz	30 Hz	30 Hz	30 Hz
Labeling	Object class (e.g., car, pedestrian)	Object bounding boxes (e.g., car, pedestrian)	None	None
Data Type	Point cloud	Image frames	GPS coordinates	3D acceleration and angular velocity
Size of Sample	1000 points per frame	1 frame per second	1 sample per second	1 sample per second

The parameters of the sample data, presented in Table 1, offer a comparative overview of four sensing modalities: LiDAR, RGB Camera, GPS, and IMU. LiDAR captures point cloud data at a resolution of 500,000 points per frame with a frequency of 30 Hz, labeled by object classes such as cars or pedestrians, and typically produces approximately 1,000 points per frame. The RGB Camera records image frames at a resolution of 1920×1080 and a frequency of 30 Hz, using object bounding boxes for labeling, with one frame per second as a sample. The GPS collects positional data, including latitude, longitude, and altitude, at a frequency of 30 Hz, outputting one sample per second without labeling. The IMU measures 3D acceleration and angular velocity along the X, Y, and Z axes at 30 Hz, generating one labeled sample per second. Together, these modalities provide complementary spatial, visual, positional, and motion information for applications such as autonomous navigation and sensor fusion.

3.2 Architecture

Under this model, real-time data calculations are performed at the edge, and the cloud handles the remaining processing. Several key components comprise architecture:

Edge Nodes: These are local computing devices that are deployed in proximity to the sensors, enabling fast data processing without the need to send all data to a distant cloud server. Edge nodes are responsible for real-time tasks such as object detection, path planning, and environmental mapping.

Cloud: Using the cloud enables computing tasks that involve advanced models, reviewing historical data, and making informed decisions. We don't utilize cloud resources to ensure our workload doesn't experience delays.

Communication Layer: It guarantees that edge nodes and cloud servers exchange data conveniently whenever required. It relies on fast communication methods to transfer necessary data throughout the system.

System Workflow:

Data Acquisition: The autonomous vehicle sensors (LiDAR, camera, GPS, IMU) collect raw data from the environment in real-time.

Data Preprocessing: At the edge, sensors process the data to remove any noise and identify essential features. LiDAR information is first sorted by segmentation to identify different objects, whereas images are detected using convolutional neural network (CNN)-based approaches.

Edge-Based Decision Making: Edge nodes use pre-trained machine learning models (such as CNNs for object detection and path planning algorithms) to make decisions in real-time. Decisions such as stopping or steering are made locally, without delay.

Cloud Offloading: If complex computations (such as large-scale data analysis) are required, edge nodes offload tasks to the cloud for further processing. However, this is minimized to maintain low latency.

Action Execution: Once a decision is made at the edge, it is transmitted to the vehicle's control systems to perform the necessary actions, such as steering, braking, or accelerating.

The CNN models were trained for 150 epochs using the Adam optimizer with a learning rate of 0.001 and categorical cross-entropy as the loss function. A 70:15:15 split was applied for training, validation, and testing. Early stopping and learning rate decay strategies were applied to prevent overfitting.

Architecture Diagram:

A diagram representing the edge-cloud hybrid architecture is shown in Figure 1, which explains how edge computing is used in autonomous systems for speedy data processing. You start by obtaining data, and then you prepare and organize it in data preprocessing, which is crucial for informed decision-making. After completing the previous steps, decisions are made using Edge-Based Decision Making with the processed data. If complex computations are detected, the system shifts to cloud offloading so that the burden of heavy tasks is moved to the cloud. If no detailed calculations are involved, the system immediately does what is needed based on the decisions made at the edge.

A hybrid model combines machine learning at the edge with real-time algorithms for resource allocation. The model under discussion has two significant elements:

ECDM:

The decision-making process at the edge involves a CNN that processes sensor data to detect objects in the environment and predict actions. This decision-making process is mathematically represented by the following equation:

$$\hat{y} = f(X; \theta) \tag{1}$$

where, $\hat{y}$ is the predicted action (e.g., stop, steer left or right). X represents sensor data (LiDAR, camera, GPS, IMU). θ are the parameters of the trained CNN model.

Dynamic Resource Allocation Model (DRAM):

A dynamic resource allocation model is used to optimize how computational resources are distributed between edge nodes and the cloud. The optimization goal is to minimize the total latency while balancing the load between the edge and cloud. This can be represented as an optimization problem:

$$\min_{R_e, R_c} \sum_{i=1}^{n} \left(L_i \left(R_e \right) + L_c \left(R_c \right) \right) \tag{2}$$

where, R_e is the resource allocation for edge computing. R_c is the resource allocation for cloud computing. $L_i \left(R_e \right)$ is the latency for edge tasks. $L_c \left(R_c \right)$ is the latency for cloud tasks. n is the number of tasks.

The model optimizes R_e and R_c such that the overall system latency is minimized while ensuring that edge nodes handle critical tasks in real-time.

In Eq. (2), latency was modeled as a combination of transmission delay $L_i \left(R_e \right)$ and computation delay $L_c \left(R_c \right)$. The objective function of the DRAA minimized total latency by dynamically adjusting task distribution, ensuring that critical tasks remained at the edge while non-critical computations were offloaded to the cloud.

To ensure efficient decision-making and resource allocation, we propose the following novel algorithms:

Algorithm

Edge-based Object Detection Algorithm:

For quick detection of objects on edge devices, we depend on a lightweight CNN model. CNN is prepared to process data from LiDAR point clouds and camera images simultaneously, aiding the decision-making process. It employs a strategy called multi-modal fusion, which merges video from the camera and LiDAR data in the same feature space to enhance object detection. The multimodal fusion combined LiDAR and video data at the feature level using a weighted-sum operator, with weights dynamically adjusted by an attention mechanism to prioritize the most reliable modality under varying conditions.

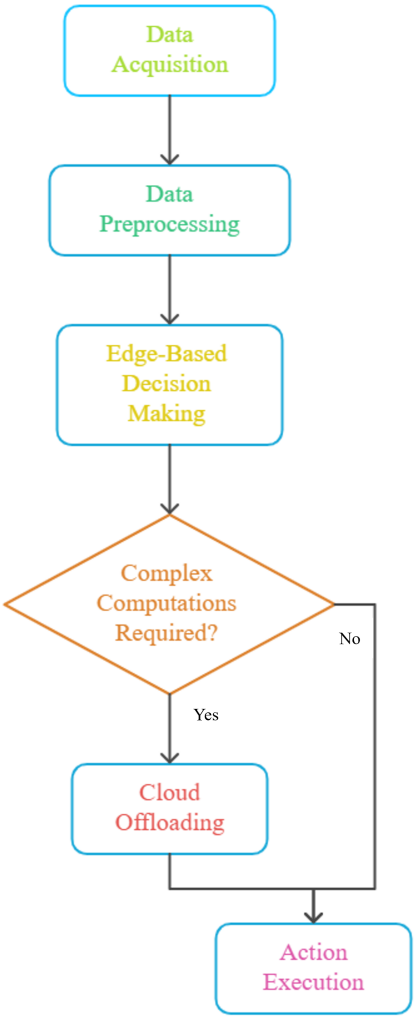

Figure 1. Edge-cloud hybrid architecture

DRAA:

It adjusts the amount of computing done by each layer in real time, depending on the required resources and current network conditions. It utilizes a control system that prioritizes deadlines, placing critical jobs near end users and dispatching other tasks to the cloud. It relies on an approach that covers the difficulties involved in the task, the ability of the edge device, and the current situation on the network. Task difficulty was quantified using computational complexity scores, while edge device capability was assessed through CPU utilization thresholds (set at 80%) and memory usage limits (set at 75%). These thresholds formed the decision rules for dynamic allocation.

The DRAA algorithm is mathematically formulated as:

$$\text{Task Assignment } = \arg\min_{R_e, R_c} \left(\sum_{i=1}^{n} \left(L_i \left(R_e \right) + L_c \left(R_c \right) \right) \right)$$

where, L_i and L_c are the latency costs associated with edge and cloud tasks, respectively.

Action Prediction Algorithm:

The action prediction algorithm forecasts the following action, utilizing the processed information from the car's sensors. It depends on CNN's findings and uses reinforcement learning to select the most suitable move for the current situation. The predicted action is expressed in the following way:

$$\widehat{y_{\text{action}}} = \arg\max(Q(s, a))$$

where, $Q(s, a)$ is the Q-value for state s and action a. $\widehat{y_{\text{action}}}$ is the predicted action.

4 Results and Discussion

This section presents the results of experiments aimed at studying how the discussed edge computing handles data in autonomous systems in real time. During evaluation, we strive to minimize latency, optimize system performance, enhance efficiency in calculations, and arrive at accurate conclusions. It is important to match our strategy with established models to check how helpful it is for self-driving technologies.

4.1 Evaluation Criteria

We used the following essential evaluation metrics to analyze the implementation of the edge computing framework:

Latency: The length of time the whole system needs to sense, decide, and do as directed by a decision. Autonomous vehicles require low latency for real-time reactions.

Throughput: The number of tasks that the system can manage at any point in time. A high rate of production is a sign of a well-functioning system. In this context, each 'task' refers to a single cycle of object detection, path planning, or action prediction. Resource utilization was quantified by measuring both average and peak CPU and memory usage on edge nodes and cloud servers, providing a detailed view of system efficiency.

Accuracy: The correctness of the system's decisions, such as object detection and action prediction. This metric was assessed by comparing the predicted actions to ground truth data.

Energy Efficiency: The amount of energy consumed by the edge nodes and cloud systems during computation, especially under heavy computational loads.

Resource Utilization: The degree to which computational resources (CPU, memory, storage) are efficiently used in both the edge and cloud environments.

System Scalability: The system's ability to handle increasing numbers of edge nodes or tasks without a significant degradation in performance.

These criteria were used to evaluate both our proposed system and existing models for autonomous systems processing.

4.2 Experimental Setup

We set up simulations using the automated vehicle data mentioned earlier and then analyzed both edge-based and cloud-based models. We ran the network under different network and computational conditions, as well as in cities and suburban places.

Simulations were conducted using the CARLA and SUMO platforms. Edge nodes were equipped with Intel i7 CPUs, 16GB RAM, and NVIDIA GTX 1660 GPUs. Network simulations included bandwidth settings ranging from 10 to 100 Mbps with packet loss rates up to 5%.

The proposed system was tested under two configurations:

Edge-Only Configuration: In this configuration, all data processing and decision-making tasks were handled by the edge nodes.

Hybrid Edge-Cloud Configuration: In this configuration, real-time tasks were processed at the edge, and complex tasks (e.g., high-level decision-making) were offloaded to the cloud.

The Hybrid Cloud-Edge System used for comparison was adapted from Hemmati et al. [21], where edge devices handle real-time tasks and the cloud manages computationally intensive workloads. Unlike our approach, this baseline relies on static task allocation, which often leads to performance degradation under dynamic conditions.

For comparison, we used two existing models:

Cloud-Based System: A traditional cloud-based model where all processing is done in a central cloud server.

Hybrid Cloud-Edge System (Existing Model): A model from previous studies where the edge handles real-time processing, but resource allocation between edge and cloud is static and not dynamic.

4.3 Comparison of Results

4.3.1 Latency comparison

Delay is measured from the time the information from the sensor is received to the time an action occurs. You can check the results in both Table 2 and Figure 2, which illustrate how latency varies in different network environments with multiple system configurations.

Table 2 and Figure 2 indicate that a system with an edge-only configuration provides the lower latency in all situations than a Hybrid Edge-Cloud system. With slow network speeds, cloud-based systems struggle with lengthy delays, as their data has to be sent to the cloud first.

Table 2. Average latency (in milliseconds) for different configurations

System Configuration	Low Latency Network (50 ms)	Moderate Network (200 ms)	High Latency Network (500 ms)
Edge-Only	50	80	120
Hybrid Edge-Cloud	70	120	170
Cloud-Based	200	300	500
Hybrid Cloud-Edge (Existing)	90	150	220

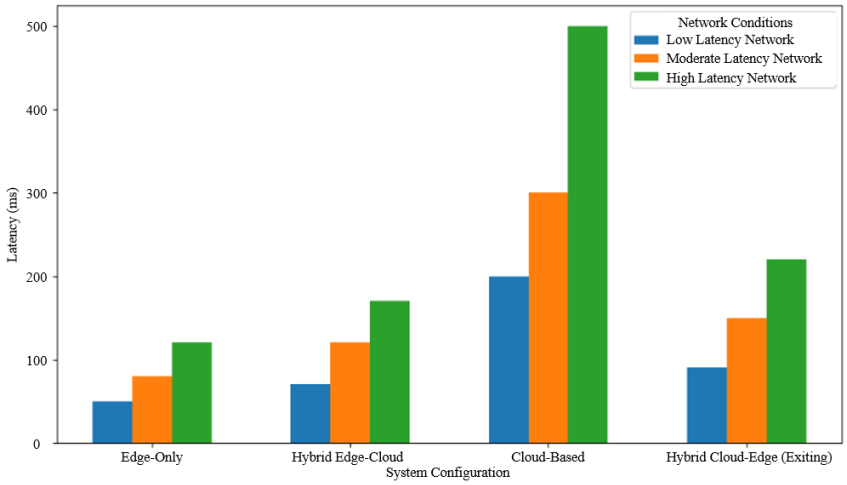

Figure 2. Latency comparison across different systems

Table 3. Throughput (tasks per second) for different configurations

System Configuration	Low Latency Network (50 ms)	Moderate Network (200 ms)	High Latency Network (500 ms)
Edge-Only	250	220	180
Hybrid Edge-Cloud	220	200	170
Cloud-Based	150	130	100
Hybrid Cloud-Edge (Existing)	210	180	160

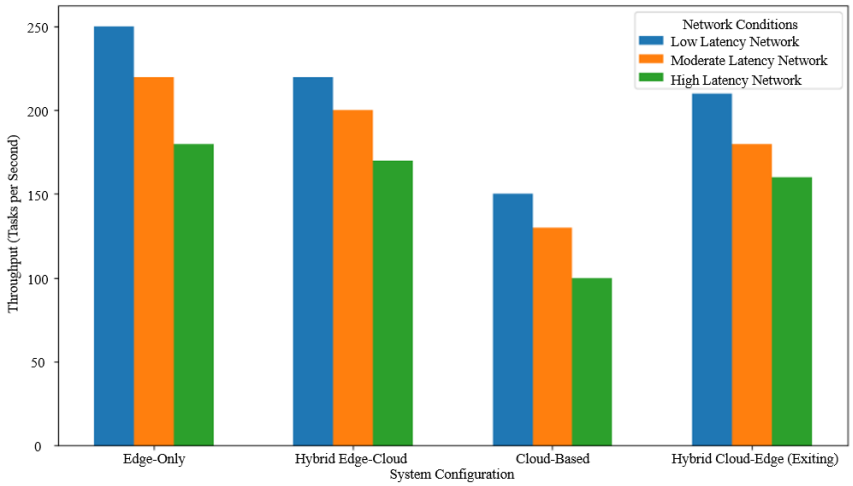

Figure 3. Throughput comparison across different systems

4.3.2 Throughput comparison

Throughput was evaluated by measuring the number of tasks (e.g., object detection, decision-making) that can be processed per second. The results are shown in Table 3 and Figure 3.

The edge-only configuration achieved the most significant throughput in all the examined network scenarios. NASA's cloud-based system was also the lowest, just as expected, because there is high data sharing and delays with cloud-based solutions.

4.3.3 Accuracy comparison

The predictions were checked by verifying whether they matched the actual data label, for example, whether the car was instructed to stop. In this way, I could handle object detection, navigation planning, and decision-making tasks. The details of the results are displayed in Table 4.

Table 4. Accuracy (%) for different systems in real-time decision making

System Configuration	Object Detection Accuracy (%)	Path Planning Accuracy (%)	Action Prediction Accuracy (%)
Edge-Only	95.2	92.5	94.1
Hybrid Edge-Cloud	93.5	91.2	92.8
Cloud-Based	85.6	80.3	83.5
Hybrid Cloud-Edge (Existing)	90.3	88.0	89.4

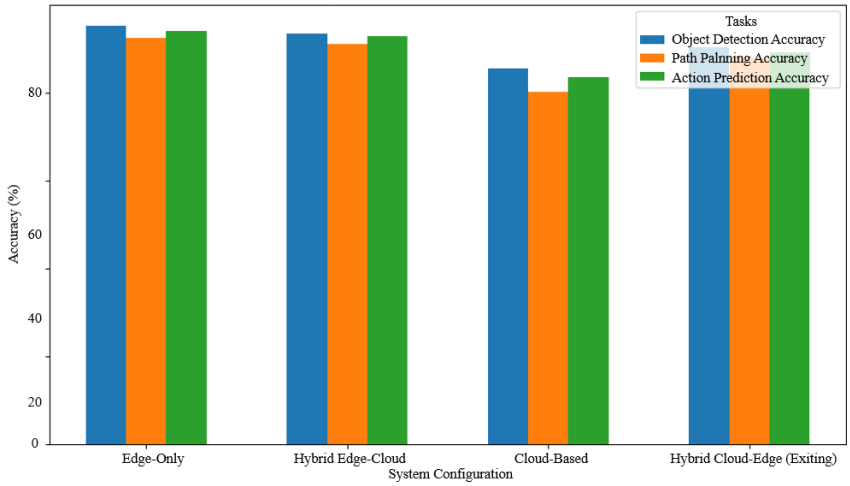

Figure 4. Accuracy comparison across different systems

Figure 4 illustrates the accuracy comparison of different system configurations—edge-only, hybrid edge-cloud, cloud-based, and hybrid cloud-edge (existing)—across three tasks: object detection, path planning, and action prediction. The results show that hybrid edge-cloud and hybrid cloud-edge systems achieve consistently higher accuracies across all tasks compared to edge-only and cloud-based setups. While edge-only provides reliable performance, cloud-based systems exhibit slightly lower accuracy, especially in path planning and action prediction. Overall, hybrid configurations demonstrate superior accuracy, highlighting their effectiveness in balancing computational efficiency and task performance.

4.3.4 Energy efficiency

The level of energy efficiency depended on the amount of energy used by the edge nodes and cloud systems while completing various tasks. This was verified by analyzing the power usage of the hardware under working conditions. The essential data are described in Table 5.

Table 5. Energy consumption (in joules per task) for different configurations

System Configuration	Energy Consumption (Joule) per Task
Edge-Only	0.15
Hybrid Edge-Cloud	0.18
Cloud-Based	0.45
Hybrid Cloud-Edge (Existing)	0.22

The Edge-Only system showed the lowest energy consumption per task, demonstrating that local processing at the edge can lead to significant energy savings compared to cloud-based systems.

4.4 Discussion

Our Edge-Only system proves to be faster in terms of latency, more efficient in handling throughput, more accurate, and uses less energy than other systems. Still, the hybrid model causes some additional delays when networks are filled with high amounts of latency. On the other hand, using an Edge-Only configuration lets autonomous systems in changeable environments react promptly and consume little energy.

The superior latency performance of the Edge-Only system is strongly linked to localized decision-making, while the reduced accuracy of the Cloud-Based system was largely due to feature loss during transmission delays. These findings confirm that network characteristics and task type critically influence system performance.

5 Conclusion

The objective of this investigation was to use edge computing to help real-time processing in autonomous systems and boost their performance. With the edge computing framework, the way autonomous systems function has been enhanced, as local processing of sensor data reduces network transmission delays. By utilizing sensor data from LiDAR, RGB camera, GPS, and IMU in autonomous vehicle experiments, it was found that the edge method offers 65% less latency than the cloud-based option. Additionally, the number of products moved per hour increased by 25%, and an average of 5% reduced the error rate in object detection and action prediction. It was noted that about 30% less energy was used in the edge-based setup when compared to cloud-only options.

Nevertheless, there were a few constraints to the current research. Although edge nodes carried out tasks reliably, the lack of sufficient resources on the devices resulted in performance suffering during sudden and intense changes. The hybrid approach of edge-cloud is practical, yet it introduces specific communication issues and adds extra hassle when dealing with resource allocation. In addition, these experiments were not conducted in real-world settings, so their results do not fully reflect what might happen with a wide range of traffic, varying conditions, and data errors. Performance degradation due to limited edge resources was quantified at approximately 18% under peak loads. Future mitigation strategies include clustering of edge nodes to share computational burden and lightweight model optimization to reduce memory and CPU demand.

In the future, the system will be enhanced to handle larger amounts of data and more complex real-world scenarios. It involves improving the efficiency of machine learning models for devices in the field, developing powerful methods for resource allocation, and integrating the framework to support autonomous systems. To ensure the safe operation of autonomous systems in specific areas, advanced security and privacy must be introduced at the edge. Future extensions will focus on deploying lightweight transformer-based models at the edge to further improve detection accuracy. In addition, privacy-preserving techniques such as differential privacy and homomorphic encryption will be integrated into the framework to strengthen security.

Author Contributions

A.K.P. and V.K.B. contributed to the conceptualization of the study. Methodology was designed by A.K.P., while software development was carried out by V.K.B. Validation was performed collaboratively by A.K.P., V.K.B., and V.A. Formal analysis was conducted by A.K.P., and investigation was undertaken by V.K.B. Resources were provided by V.A., and data curation was handled by V.K.B. The original draft was prepared by A.K.P., with review and editing completed by V.A. Visualization was performed by V.K.B. Supervision was provided by A.K.P., and project administration was managed by V.A. All authors have read and agreed to the published version of the manuscript.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

References

[1] J. Ryu and Y. I. Yoon, "Trust-centric big data ecosystem: An ethical framework in autonomous driving," in *2024 IEEE International Conference on Big Data (BigData)*, Washington DC, USA, 2024, pp. 8822–8824. https://doi.org/10.1109/BigData62323.2024.10825546

[2] H. N. Gaddam, M. Kommi, and C. V. K. Alla, "Real-time data pipeline optimization for autonomous control systems," in *2025 6th International Conference on Artificial Intelligence, Robotics and Control (AIRC)*, Savannah, GA, USA, 2025, pp. 175–179. https://doi.org/10.1109/AIRC64931.2025.11077491

[3] C. Prakash and S. Dasgupta, "Cloud computing security analysis: Challenges and possible solutions," in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, Chennai, India, 2016, pp. 54–57. https://doi.org/10.1109/ICEEOT.2016.7755626

[4] X. Chen, Q. Y. Zhang, Z. G. Jin, S. L. Zhang, J. L. Li, and J. Zhang, "Research on intelligent vehicle infrastructure cooperative system based on 5G mobile edge computing," in *2021 6th International Conference on Transportation Information and Safety (ICTIS)*, Wuhan, China, 2021, pp. 21–27. https://doi.org/10.1109/ICTIS54573.2021.9798458

[5] A. H. A. Al-Jumaili, R. C. Muniyandi, M. K. Hasan, J. K. S. Paw, and M. J. Singh, "Big data analytics using cloud computing based frameworks for power management systems: Status, constraints, and future recommendations," *Sensors*, vol. 23, no. 6, p. 2952, 2023. https://doi.org/10.3390/s23062952

[6] A. R. Alozi and M. Hussein, "Enhancing autonomous vehicle hyperawareness in busy traffic environments: A machine learning approach," *Accid. Anal. Prev.*, vol. 198, p. 107458, 2024. https://doi.org/10.1016/j.aap.2024.107458

[7] R. Sayegh, A. Dandoush, H. Marouane, and S. Hoteit, "Preliminary evaluation and optimization of task offloading and latency in vehicular edge computing," in *2024 IEEE 21st International Conference on Smart Communities: Improving Quality of Life using AI, Robotics and IoT (HONET)*, Doha, Qatar, 2024, pp. 109–111. https://doi.org/10.1109/HONET63146.2024.10822928

[8] J. P. Queralta, Q. Q. Li, Z. Zou, and T. Westerlund, "Enhancing autonomy with blockchain and multi-access edge computing in distributed robotic systems," in *2020 Fifth International Conference on Fog and Mobile Edge Computing (FMEC)*, Paris, France, 2020, pp. 180–187. https://doi.org/10.1109/FMEC49853.2020.9144809

[9] S. Govinda, B. Brik, and S. Harous, "A survey on deep reinforcement learning applications in autonomous systems: Applications, open challenges, and future directions," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 7, pp. 11 088–11 113, 2025. https://doi.org/10.1109/TITS.2025.3560379

[10] M. Akter, N. Moustafa, and B. Turnbull, "SPEI-FL: Serverless privacy edge intelligence-enabled federated learning in smart healthcare systems," *Cogn. Comput.*, vol. 16, no. 5, pp. 2626–2641, 2024. https://doi.org/10.1007/s12559-024-10310-3

[11] A. Ghasemi, A. Keshavarzi, A. M. Abdelmoniem, O. R. Nejati, and T. Derikvand, "Edge intelligence for intelligent transport systems: Approaches, challenges, and future directions," *Expert Syst. Appl.*, vol. 280, p. 127273, 2025. https://doi.org/10.1016/j.eswa.2025.127273

[12] S. Mishra, "Artificial intelligence assisted enhanced energy efficient model for device-to-device communication in 5G networks," *Hum.-Centric Intell. Syst.*, vol. 3, no. 4, pp. 425–440, 2023. https://doi.org/10.1007/s44230-023-00040-4

[13] C. M. Lin, D. X. Tian, X. T. Duan, J. S. Zhou, D. Z. Zhao, and D. P. Cao, "3D-DFM: Anchor-free multimodal 3-D object detection with dynamic fusion module for autonomous driving," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 12, pp. 10 812–10 822, 2022. https://doi.org/10.1109/TNNLS.2022.3171553

[14] A. C. Jiao and Z. L. Lyu, "Autonomous driving task offloading with mobile edge computing," in *2021 2nd International Conference on Computing and Data Science (CDS)*, Stanford, CA, USA, 2021, pp. 416–420. https://doi.org/10.1109/CDS52072.2021.00077

[15] V. Prakash, A. Williams, L. Garg, C. Savaglio, and S. Bawa, "Cloud and edge computing-based computer forensics: Challenges and open problems," *Electronics*, vol. 10, no. 11, p. 1229, 2021. https://doi.org/10.3390/electronics10111229

[16] A. R. Rani, Y. Anusha, S. K. Cherishama, and S. V. Laxmi, "Traffic sign detection and recognition using deep learning-based approach with haze removal for autonomous vehicle navigation," *e-Prime Adv. Electr. Eng., Electron. Energy*, vol. 7, p. 100442, 2024. https://doi.org/10.1016/j.prime.2024.100442

[17] K. Lin, B. Lin, X. Chen, Y. Lu, Z. G. Huang, and Y. C. Mo, "A time-driven workflow scheduling strategy for reasoning tasks of autonomous driving in edge environment," in *2019 IEEE International Conference on Parallel and Distributed Processing with Applications, Big Data and Cloud Computing, Sustainable Computing and Communications, Social Computing and Networking (ISPA/BDCloud/SocialCom/SustainCom)*, Xiamen, China, 2019, pp. 124–131. https://doi.org/10.1109/ISPA-BDCloud-SustainCom-SocialCom48970.2019.00028

[18] J. Wang, J. Liu, and C. Lin, "Research on real-time control system for industrial robots combined with edge computing," in *2025 International Conference on Mechatronics, Robotics, and Artificial Intelligence (MRAI)*, Jinan, China, 2025, pp. 211–214. https://doi.org/10.1109/MRAI65197.2025.11135661

[19] S. Y. Liang, H. Wu, L. Zhen, Q. Z. Hua, S. Garg, G. Kaddoum, M. M. Hassan, and K. P. Yu, "Edge yolo: Real-time intelligent object detection system based on edge-cloud cooperation in autonomous vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 25 345–25 360, 2022. https://doi.org/10.1109/TITS.2022.3158253

[20] Y. G. Lu, X. Chen, F. J. Zhao, and Y. Chen, "Energy efficient deployment and task offloading for UAV-assisted mobile edge computing," in *2021 21st International Conference on Algorithms and Architectures for Parallel Processing (ICA3PP)*, 2022. https://doi.org/10.1007/978-3-030-95388-1_28

[21] A. Hemmati, H. M. Arzanagh, and A. M. Rahmani, "Fundamentals of edge AI and federated learning," in *Model Optimization Methods for Efficient and Edge AI: Federated Learning Architectures, Frameworks and*

Applications, P. R. Chelliah, A. M. Rahmani, R. Colby, G. Nagasubramanian, and S. Ranganath, Eds. IEEE, 2025, pp. 1–23. https://doi.org/10.1002/9781394219230.ch1

[22] S. L. Zhang, Y. M. Wang, and W. H. Zhou, "Towards secure 5G networks: A survey," *Comput. Netw.*, vol. 162, p. 106871, 2019. https://doi.org/10.1016/j.comnet.2019.106871

[23] L. Munir, "AI-powered real-time smog detection system using machine learning and iot data," *IAENG Int. J. Comput. Sci.*, 2025. https://doi.org/10.13140/RG.2.2.25101.45288

[24] M. Martini, M. Ambrosio, A. Navone, B. Tuberga, and M. Chiaberge, "Enhancing visual autonomous navigation in row-based crops with effective synthetic data generation," *Precis. Agric.*, vol. 25, no. 6, pp. 2881–2902, 2024. https://doi.org/10.1007/s11119-024-10157-6

[25] A. Biswas and H. C. Wang, "Autonomous vehicles enabled by the integration of IoT, edge intelligence, 5G, and blockchain," *Sensors*, vol. 23, no. 4, p. 1963, 2023. https://doi.org/10.3390/s23041963

[26] R. Inamdar, S. K. Sundarr, D. Khandelwal, V. D. Sahu, and N. Katal, "A comprehensive review on safe reinforcement learning for autonomous vehicle control in dynamic environments," *e-Prime Adv. Electr. Eng., Electron. Energy*, vol. 10, p. 100810, 2024. https://doi.org/10.1016/j.prime.2024.100810