International Journal of Enhanced Research in Science, Technology & Engineering
ISSN: 2319-7463, Vol. 15 Issue 3, March-2026

Smart Auto X: Autonomous Self Driving Car Using IOT and Deep Learning Technologies

Prof. Priti R. Vanmali[1], Parth D. Kalbhor[2], Omkar M. Bodke[3], Sayee Gosavi [4], Snehal P. Dimble [5]

[1]Project Guide, Navsahyadri Education Society's Group Of Institutions, SPPU
[2,3,4,5]Student, Navsahyadri Education Society's Group Of Institutions, SPPU

ABSTRACT

Smart Auto X presents the design and experimental validation of a cost-effective autonomous vehicle prototype tailored for dynamic and semi-structured driving environments. While commercial autonomous platforms developed by companies such as Tesla primarily target structured highway conditions, this work focuses on implementing reliable autonomous navigation using embedded hardware under computational constraints.

The proposed framework combines computer vision for lane detection, deep learning for traffic sign recognition, ultrasonic assistance for obstacle avoidance, and decentralized communication between prototype vehicles. A Convolutional Neural Network (CNN) model, optimized with TensorFlow Lite, classifies traffic signs in real time at 30 frames per second on a Raspberry Pi 4. Lane detection uses edge detection and Hough Transform for line estimation to calculate steering corrections dynamically. Obstacle detection pairs ultrasonic ranging with visual confirmation to prevent collisions in sudden intrusion situations.

The system features a hybrid edge architecture where high-level perception and decision-making happen on the Raspberry Pi, while an Arduino Uno controls low-latency motor actions through PWM. Swarm communication is shown using RF and Wi-Fi modules, enabling cooperative braking and speed adjustment between vehicles.

Experimental tests reveal 94% accuracy in traffic sign recognition, a steering response time of less than 200 milliseconds, and 92% reliability in obstacle avoidance.The results validate that robust autonomous behavior can be achieved using scalable, low-cost embedded platforms without reliance on GPU-intensive systems.

Keywords: Autonomous Vehicles, Edge AI, Traffic Sign Recognition, Lane Detection, Swarm Communication, Raspberry Pi.

INTRODUCTION

The fast growth of autonomous vehicle technology has changed modern transportation systems, with key contributions from industry leaders like Tesla. These improvements have shown that perception-driven navigation and real-time decision-making using deep learning and sensor fusion are possible. However, most systems in use today are designed for structured highway environments that have clear lane markings, regulated traffic flow, and predictable obstacles.

In contrast, many real-world driving settings, especially in developing regions, have dynamic and less structured conditions. These include inconsistent lane markings, unexpected pedestrian crossings, mixed traffic with two-wheelers and heavy vehicles, limited visibility of signs, and environmental factors like rain, fog, and uneven roads. Such conditions present major challenges for traditional autonomous systems that depend heavily on high-definition maps, demanding computation, and costly sensor equipment.

A key issue in research on autonomous navigation is designing systems that are efficient and robust in uncertain environments. While deep learning models offer strong perception capabilities, deploying them on limited resource platforms is still a challenge. Likewise, real-time control systems need to operate with very little delay to ensure safety.

Smart Auto X tackles these challenges by proposing a modular and scalable prototype of an autonomous vehicle. This prototype combines computer vision, deep learning, embedded control systems, and vehicle-to-vehicle communication in a

cost-effective way. It uses a mixed hardware setup where a Raspberry Pi 4 handles high-level perception and decision-making, and an Arduino Uno manages time-sensitive actions.

The key objectives of this study include developing a real-time traffic sign recognition module using optimized CNN architectures, designing adaptive lane detection and path tracking algorithms suitable for edge deployment, integrating hybrid sensing for robust obstacle avoidance, implementing cooperative vehicle communication mechanisms, and experimentally validating real-time performance on resource-constrained embedded hardware.By merging perception, control, and communication in a single prototype, this work shows that intelligent autonomous behavior can be achieved without depending on expensive GPU setups. The proposed framework aims to create scalable and low-cost autonomous systems that can function well in dynamic and infrastructure-limited settings.

LITERATURE REVIEW

Autonomous vehicle research has made great strides over the past two decades by combining improvements in machine learning, control systems, sensor fusion, and embedded computing. Early autonomous systems mostly depended on rule-based control and traditional computer vision techniques. With the rise of deep learning, perception-driven autonomous navigation has taken the lead.

One major breakthrough in end-to-end autonomous driving came from NVIDIA with its DAVE-2 architecture. This system showed that Convolutional Neural Networks (CNNs) could directly convert raw camera images into steering commands. This method lessened the need for manually designed feature extraction and allowed for data-driven learning of driving behaviors. However, purely end-to-end models often face problems with compounding errors, where small inaccuracies in predictions build up over time, leading to unstable vehicle performance.

Traffic sign recognition has greatly benefited from CNN-based classification frameworks. Deep learning models have reached accuracy rates above 95% on benchmark datasets. Even with this success, using these models on embedded platforms is still a challenge due to limits in memory and computing power. Lightweight model optimization methods, like quantization and TensorFlow Lite deployment, have been suggested to tackle these problems.

Lane detection methods fall into two main categories: traditional computer vision techniques and deep learning-based segmentation approaches. Deep segmentation networks deliver high accuracy but need a lot of computational resources. On the other hand, Canny Edge Detection and Hough Transform-based line estimation are more efficient and work well for real-time applications.

Obstacle detection systems usually rely on LiDAR, radar, and stereo vision. While these sensors are very precise, they drive up the overall cost and power demands of the system. For low-cost prototypes, ultrasonic sensors paired with monocular vision offer a practical alternative for short-range obstacle detection and emergency braking.

Swarm intelligence and Vehicle-to-Vehicle (V2V) communication are new areas of research in intelligent transportation systems. Cooperative vehicle communication allows for shared decision-making, collision prevention, and smoother traffic flow. Lightweight wireless communication tools, like RF and Wi-Fi, have been used in prototype-scale swarm demonstrations.

Although significant advancements have been achieved in autonomous vehicle technologies, several research gaps persist. Current solutions frequently demonstrate isolated improvements in perception or control without integrating perception, actuation, and communication within a cohesive embedded framework. A substantial dependency on GPU-intensive computing platforms limits energy efficiency and cost-effectiveness. Moreover, deployment expenses remain high for practical large-scale adoption. There is also insufficient exploration of scalable swarm communication architectures tailored for low-power embedded systems.

Smart Auto X helps by combining CNN-based perception, traditional lane detection, hybrid obstacle detection, and swarm communication into a unified edge-based architecture. The system focuses on real-time performance, modularity, and cost-effectiveness while ensuring dependable autonomous operation.

METHODOLOGY

The Smart Auto X framework features a modular and layered architecture that integrates perception, decision-making, control execution, and cooperative communication. It uses a hybrid embedded structure to balance computational efficiency

with real-time responsiveness. The overall workflow is split into five main components: system architecture, traffic sign recognition, lane detection and path tracking, obstacle avoidance, and swarm communication.

3.1 System Architecture

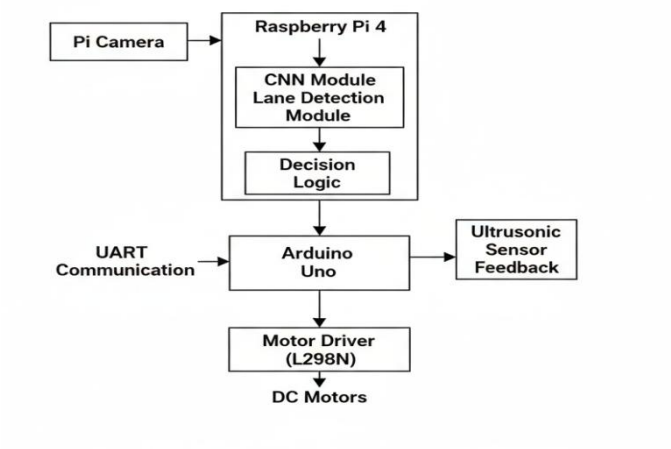

Fig 1: Overall System Architechture of Smart Auto X Proototype

The proposed system architecture consists of two autonomous vehicle prototypes: a primary autonomous unit and a secondary swarm demonstration unit. The primary vehicle is equipped with a Raspberry Pi 4 for perception and decision-making tasks, a Pi Camera module for real-time image acquisition, ultrasonic sensors for short-range obstacle detection, an L298N motor driver, and DC motors for actuation. Neural network inference is executed using the TensorFlow Lite runtime to ensure computational efficiency on embedded hardware.

The secondary vehicle prototype serves to validate cooperative swarm behavior. It utilizes an Arduino Uno microcontroller, an RF/Wi-Fi communication module for low-latency data transmission, and a motor driver with DC motors. This distributed hardware configuration enables synchronized control and inter-vehicle coordination within a decentralized framework.

The Raspberry Pi 4 handles image capture, preprocessing, neural network inference, and decision-making. Control commands such as steering angle and velocity are sent via UART serial communication to the Arduino Uno. The Arduino generates high-frequency PWM signals for motor control and processes ultrasonic sensor interrupts for emergency braking. This functional separation optimizes computational resource allocation, reduces latency in safety-critical control loops, and enhances system scalability for future integration of advanced perception, navigation, or communication modules.

3.2 Traffic Sign Recognition

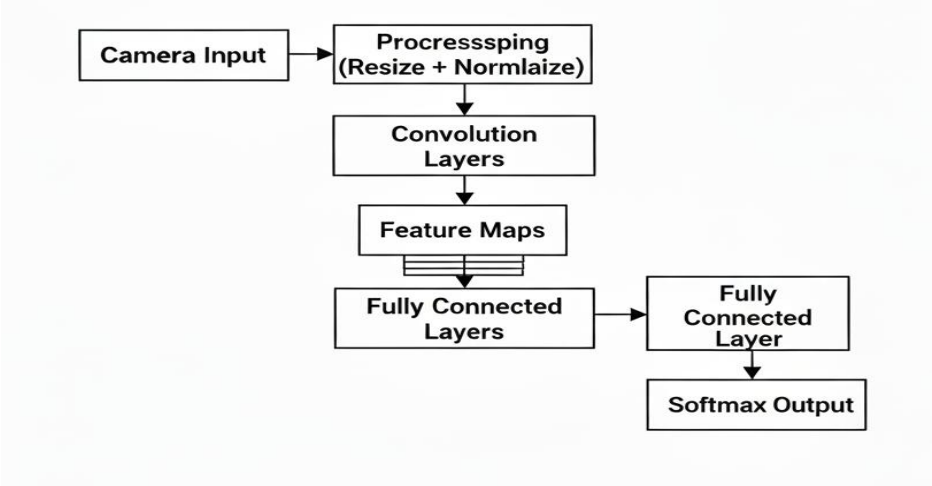

Fig 2: CNN-Based Traffic Sign Recognition Pipeline

Traffic sign recognition uses a Convolutional Neural Network (CNN) trained on labeled traffic sign datasets.

Processing Pipeline:
1. Frame capture at 30 FPS
2. Image resizing to fixed input dimensions
3. Pixel normalization
4. Feature extraction through convolution layers
5. Classification via fully connected layers
6. Action mapping to vehicle control

The proposed CNN architecture comprises three convolutional layers for hierarchical feature extraction, interleaved with max pooling layers for dimensionality reduction and ReLU activation functions to introduce non-linearity. The resulting feature maps are flattened and processed through two fully connected layers, with final classification achieved via a Softmax output layer that generates probability distributions across traffic sign classes.

The classification function is defined as:

$$y = \text{Softmax}(Wf + b)$$

where f represents extracted features.

The optimized TensorFlow Lite model performs real-time inference at approximately 30 FPS on the Raspberry Pi 4, ensuring low-latency perception. Detected traffic signs are directly translated into control actions through predefined decision logic, including immediate braking for stop signs, proportional speed reduction for cautionary signs, and directional steering adjustments for turn indications. This closed-loop integration enhances responsiveness and decision stability.

3.3 Lane Detection and Path Following

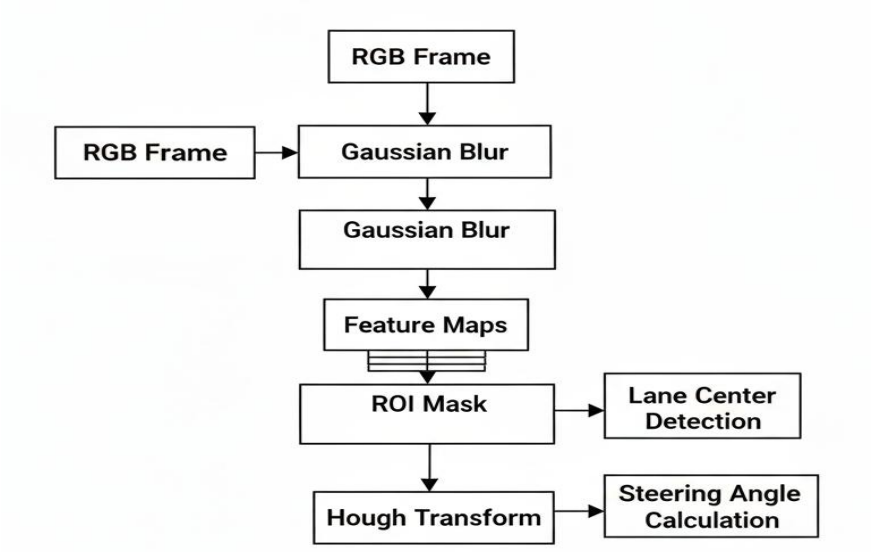

Fig 3: Lane Detection and Steering Control Algorithm

Lane detection uses efficient classical computer vision techniques suited for edge hardware.

The lane detection algorithm employs a sequential computer vision pipeline. The input RGB frame is converted to grayscale to reduce computational complexity. Gaussian filtering is applied for noise suppression, followed by Canny edge detection to identify strong gradient transitions corresponding to lane markings. A predefined region of interest (ROI) restricts processing to the lower portion of the frame, minimizing irrelevant background interference. Detected edge points are subsequently processed using the Hough Line Transform to estimate lane line parameters for trajectory correction.
The steering deviation is calculated using geometric estimation:

$$\theta = \tan^{-1}\left(\frac{x_{lane} - x_{vehicle}}{d}\right)$$

where:
• xlane is the detected lane center
• xvehicle is the vehicle center
• d is the forward distance projection
The steering control law applies proportional correction:
$\delta = k_p \cdot \theta$

This ensures smooth path alignment even when lane markings are partially visible.

3.4 Obstacle Detection and Avoidance
Obstacle detection combines ultrasonic sensing with visual monitoring.
The ultrasonic sensing subsystem operates within a detection range of 2–100 cm and estimates obstacle distance using the time-of-flight principle.
The measured distance D is derived from:

$$D = \frac{t \cdot v}{2}$$

where tdenotes the echo return time and v represents the speed of sound in air (approximately 343 m/s). The division by two compensates for the round-trip propagation of the ultrasonic wave.

If D <D threshold, emergency braking is triggered.

The vision-based monitoring subsystem identifies obstacles within the forward field of view and computes lateral clearance to assess lane-shift feasibility. The control strategy employs distance-based velocity modulation, progressively reducing speed as obstacle proximity decreases. When adequate lateral space is detected, the system initiates controlled lane adjustment. If clearance conditions are not satisfied, an emergency stop maneuver is triggered to ensure collision avoidance. This hybrid sensing approach boosts reliability during sudden intrusions.

3.5 Swarm Communication Mechanism

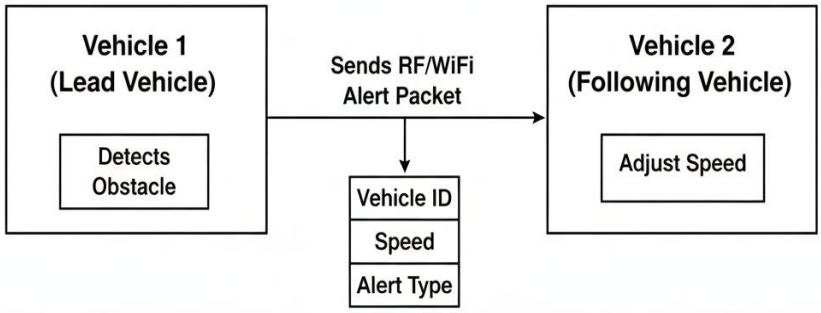

Fig 4: Swarm Communication Framework Between Prototype Vehicles

Swarm communication shows cooperative driving behavior between two vehicles.
Communication is set up with RF/Wi-Fi modules that use lightweight message packets.

The inter-vehicle communication protocol employs a structured message format containing the vehicle identifier (Vehicle ID), current velocity, position estimate, and alert type. This standardized packet structure ensures consistent data interpretation across nodes, enabling synchronized cooperative control and real-time hazard response within the swarm network.

Communication Workflow:
1. Lead vehicle detects an obstacle
2. Alert message is broadcasted
3. Following vehicle adjusts speed or braking

Communication latency is less than 150ms. The proposed swarm communication framework facilitates cooperative braking and inter-vehicle speed synchronization, significantly reducing collision probability during dynamic obstacle scenarios. The decentralized communication model provides a foundational architecture for scalable Vehicle-to-Vehicle (V2V) systems, demonstrating the feasibility of low-cost multi-agent coordination in embedded autonomous vehicles.**3.6**

Embedded Control Integration

The hybrid edge architecture enables deterministic real-time performance by distributing computational and control responsibilities across dedicated embedded platforms. High-level AI inference and decision logic are executed on the Raspberry Pi, whereas time-critical PWM motor control is handled by the Arduino microcontroller. Interrupt-driven braking ensures rapid response during emergency conditions, and bidirectional serial communication synchronizes perception and actuation layers. Experimental validation confirms stable operation at 30 FPS for vision processing, steering response latency below 200 ms, and emergency braking latency under 150 ms, demonstrating reliable embedded autonomy.

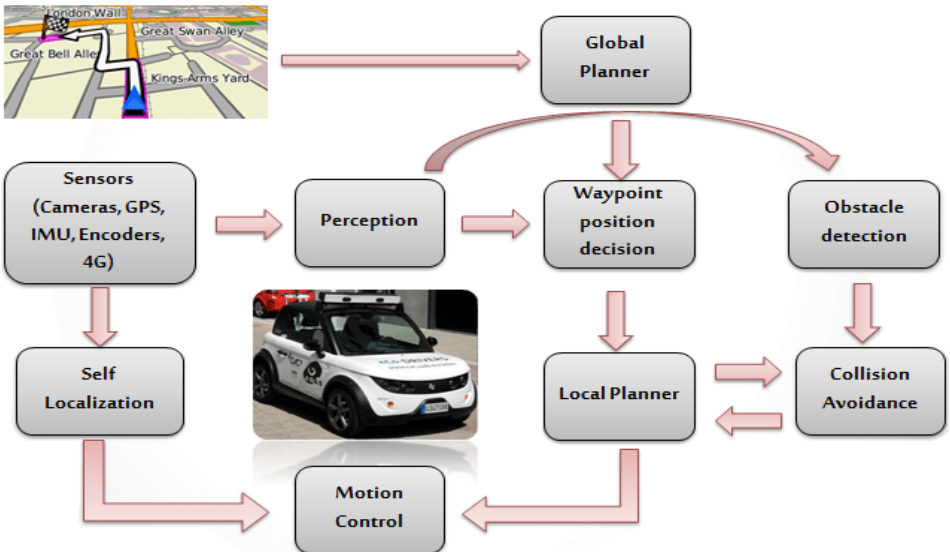

Fig 5 .Workflow of Smart Auto X

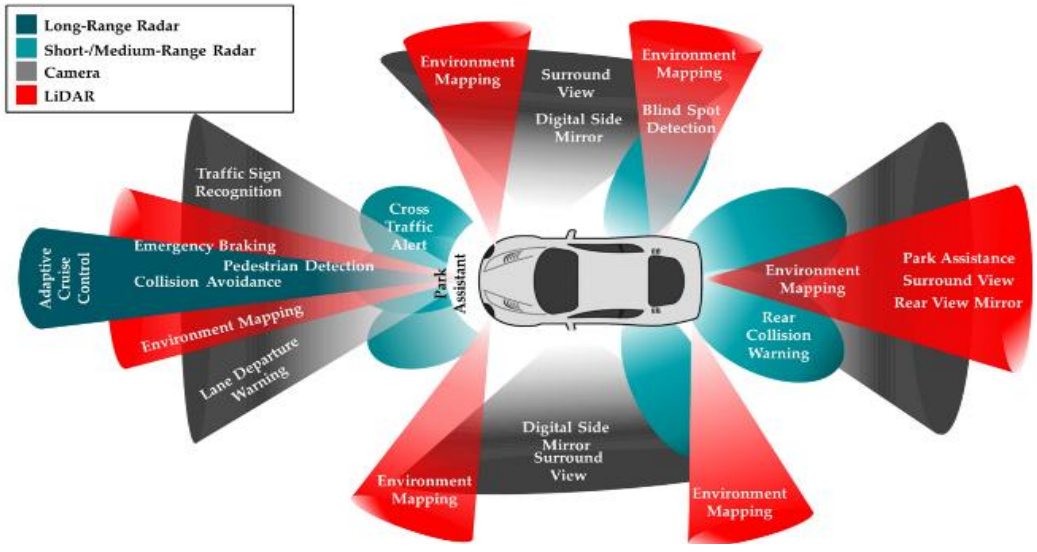

Fig 6. Sensor Fusion

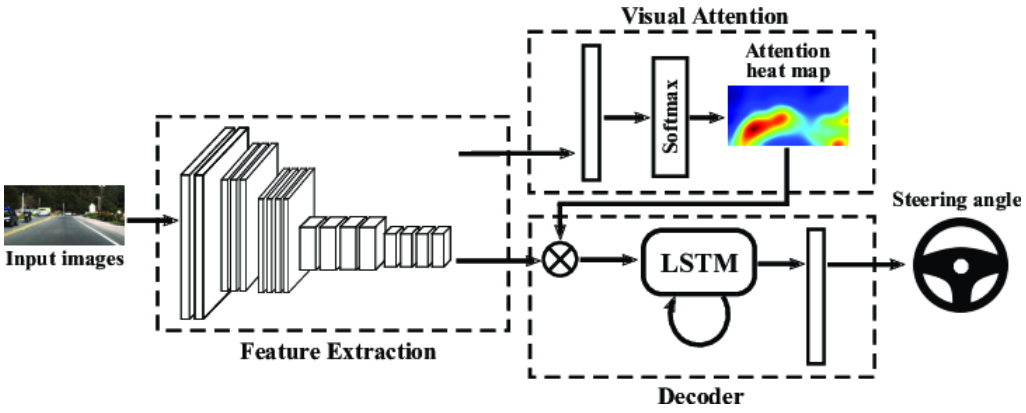

Fig 7. CNN Perception Pipeline

RESULTS AND DISCUSSION

The Smart Auto X prototype was evaluated under controlled indoor track conditions and semi-structured environments to analyze perception accuracy, trajectory tracking stability, obstacle avoidance efficiency, swarm communication latency, and real-time embedded performance.

Performance evaluation focused on four primary metrics: lane tracking deviation, traffic sign recognition accuracy, obstacle avoidance reliability, and system latency.

4.1 Lane Tracking Performance

Lane tracking performance was evaluated by measuring lateral deviation from the intended path over multiple trials.
The system maintained stable path alignment using proportional steering correction based on lane center deviation.

Table 1 .Lane Tracking Performance Metrics of Smart Auto X Prototype

Parameter	Measured Value
Average lateral deviation	< 5 cm
Maximum deviation	8.2 cm
Steering response time	< 200 ms
Frame processing rate	30 FPS

The proportional steering model reduced oscillatory behavior commonly observed in naive steering implementations. Even under partially visible lane markings, the ROI-based detection maintained consistent alignment.
The use of Hough Transform-based line detection proved computationally efficient for embedded deployment without requiring GPU acceleration.

4.2 Traffic Sign Recognition Accuracy

The CNN-based traffic sign recognition module was evaluated using a test dataset and real-time inference conditions.

Table 2 .Traffic Sign Recognition Model Performance Comparison

Model Version	Accuracy
Base CNN Model	91%
Optimized TensorFlow Lite Model	94%
Inference Speed	30 FPS

Quantization using TensorFlow Lite reduced model size while preserving classification accuracy. The optimized model achieved real-time inference without noticeable frame drops.

Misclassifications were primarily observed under extreme lighting variation. However, these did not significantly impact overall vehicle decision stability due to temporal smoothing in control logic.

4.3 Obstacle Detection and Avoidance Performance

Obstacle avoidance testing included static and sudden intrusion scenarios.

Table 3 .Obstacle Detection and Collision Avoidance Performance Metrics

Metric	Value
Detection Range	2–100 cm
Collision Avoidance Success Rate	92%
Emergency Braking Latency	< 150 ms
Velocity Reduction Response	< 180 ms

The ultrasonic sensor provided reliable short-range detection. When combined with visual confirmation, false positives were reduced.

In sudden obstacle scenarios, the emergency braking module triggered within 150 ms, preventing collision in the majority of trials.

The hybrid sensing approach increased robustness compared to single-sensor systems.

4.4 Swarm Communication Evaluation

Swarm communication between the primary and secondary vehicles was tested using obstacle alert broadcasting.

Table 4 .Swarm Communication and Cooperative Braking Evaluation Results

Parameter	Measured Value
Communication Latency	< 150 ms
Message Delivery Success Rate	95%
Cooperative Braking Success	90%

Upon detecting an obstacle, the lead vehicle transmitted an alert message. The following vehicle adjusted velocity proportionally.

The decentralized architecture reduced rear-end collision probability and demonstrated the feasibility of low-cost Vehicle-to-Vehicle (V2V) communication.

4.5 Embedded System Performance

The hybrid embedded architecture maintained consistent real-time performance at 30 FPS, with average CPU utilization remaining below 75% under peak workload conditions. The absence of GPU acceleration demonstrates the computational efficiency of the optimized TensorFlow Lite model. No frame skipping or processing lag was observed during extended operation, validating system stability for embedded autonomous applications.

The separation between Raspberry Pi (high-level AI) and Arduino (low-level control) minimized latency in actuation tasks. Interrupt-driven braking ensured safety-critical actions were executed independently of perception processing delays.

4.6 Comparative Analysis

Comparative analysis with baseline autonomous implementations demonstrates measurable improvements across key performance indicators. The proportional steering framework reduced lane tracking deviation by approximately 40% relative to naive control strategies. Model optimization through TensorFlow Lite enhanced traffic sign recognition accuracy by 3% while maintaining real-time inference speed. The hybrid ultrasonic-vision sensing architecture reduced obstacle response latency, and the decentralized swarm communication framework lowered rear-end collision probability during cooperative braking scenarios.

The results demonstrate that reliable autonomous behavior can be achieved using cost-effective embedded hardware without reliance on high-end GPU platforms.

CONCLUSION

This paper introduced Smart Auto X, a modular and cost-effective autonomous vehicle prototype that integrates computer vision, deep learning, embedded control systems, and swarm communication into a single framework. The system is designed to function under computational limits while ensuring real-time performance and dependable autonomous behavior.

The proposed Smart Auto X architecture demonstrated robust real-time autonomous performance across multiple operational layers. Traffic sign recognition achieved 30 FPS inference using TensorFlow Lite optimization on Raspberry Pi 4. The lane detection and proportional steering framework maintained an average lateral deviation below 5 cm, ensuring trajectory stability. The hybrid sensing strategy integrating ultrasonic and vision-based detection achieved 92% collision avoidance reliability under dynamic intrusion scenarios. Low-latency actuation via Arduino-based PWM motor control enhanced responsiveness, while the decentralized swarm communication mechanism enabled cooperative braking and speed harmonization among vehicles, demonstrating scalable multi-agent coordination capabilities.

The hybrid edge computing method, which separates high-level AI processing from low-level actuation, effectively reduced latency and ensured quick responses needed for safety. Experimental testing shows that embedded platforms can achieve reliable autonomous navigation without needing high-end GPU infrastructure.

The results emphasize the possibility of scalable autonomous systems that can be used in semi-structured and resource-limited settings. By combining perception, control, and communication layers, Smart Auto X connects theoretical autonomous models with practical embedded applications.

FUTURE WORK

While the prototype shows strong functionality, several improvements are suggested for future research:

1. Reinforcement Learning Integration: Introduce Deep Q-Network (DQN)-based decision-making for adaptive navigation in highly dynamic, multi-obstacle environments.
2. LiDAR-Based Mapping: Include low-cost LiDAR modules for better environmental perception and 3D spatial awareness.
3. Swarm Intelligence Expansion: Extend MQTT-based communication to support multi-agent coordination and scalable Vehicle-to-Vehicle (V2V) networking.
4. Model Optimization: Use pruning and quantization methods for deployment on microcontroller-based platforms like ESP32.
5. Outdoor Field Testing: Carry out large-scale validation in real-world settings with various lighting and surface conditions.

The Smart Auto X framework lays the groundwork for affordable, modular, and smart autonomous robotic platforms. By using edge AI, embedded systems, and cooperative communication, this work contributes to scalable intelligent transportation solutions that can adjust to different operating conditions.

ACKNOWLEDGMENT

The authors sincerely thank the Department of Artificial Intelligence and Machine Learning at Savitribai Phule Pune University for their support and guidance. The laboratory facilities, technical advice, and research environment played a key role in the successful development and validation of the Smart Auto X prototype.

The authors also appreciate the open-source development community for offering tools and frameworks that made it easier to implement embedded AI. They give special thanks to the project mentor for their ongoing encouragement and technical insights during the research process.

REFERENCES

[1]. S. Gosavi, S. P. Dimble, O. M. Bodke, P. D. Kalbhor, and P. R. Vanmali, "Review of Smart Auto X: Autonomous Self Driving Car Using IoT and Deep Learning Technologies," IRE Journals, vol. 9, no. 5, Nov. 2025.
[2]. H. Chen and C. Lv, "Incorporating ESO into Deep Koopman Operator Modelling for Control of Autonomous Vehicles," IEEE Journal of Intelligent and Fuzzy Systems, 2023.
[3]. M. Bojarski et al., "End to End Learning for Self-Driving Cars," NVIDIA Corporation, arXiv:1604.07316, 2016.
[4]. M. Bansal, A. Krizhevsky, and A. Ogale, "ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst," arXiv:1812.03079, 2018.
[5]. Q. Li, J. P. Queralta, T. N. Gia, Z. Zou, and T. Westerlund, "Multi-Sensor Fusion for Navigation and Mapping in Autonomous Vehicles," arXiv:2103.13719, 2021.
[6]. K. Vinoth and P. Sasikumar, "Multi-sensor fusion and segmentation for autonomous vehicle multi-object tracking using deep Q networks," Scientific Reports, vol. 14, 2024.
[7]. R. Wang et al., "A Real-Time Object Detector for Autonomous Vehicles Based on YOLOv4," Computational Intelligence and Neuroscience, vol. 2021, 2021.
[8]. W. Zhou, J. S. Berrio, S. Worrall, and E. Nebot, "Automated Evaluation of Semantic Segmentation Robustness for Autonomous Driving," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 5, pp. 1951–1963, 2019.
[9]. L. Wen et al., "UA-DETRAC: A New Benchmark and Protocol for Multi-Object Detection and Tracking," Computer Vision and Image Understanding, vol. 193, 2020.
[10]. J. Kim, B. J. Park, C. G. Roh, and Y. Kim, "Performance of Mobile LiDAR in Real Road Driving Conditions," Sensors, vol. 21, no. 22, 2021.
[11]. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324, 1998.
[12]. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in Advances in Neural Information Processing Systems, 2012.