Journal of Trends in Computer Science and Smart Technology (ISSN: 2582-4104)
www.irojournals.com/tcsst/

# Enhancing Road Safety and User Experience with Adaptive Object Detection in Car Infotainment Systems

**Manisabari S.[1], Nachiappan Adhithiyan N.[2], Rithish Nayagan R.[3], Sivaharish S.[4], Merlin Sajini M. L.[5]**

[1-4]Students, [5]Assistant Professor, Department of EEE CIT, Coimbatore, India

**Email:** [1]smanisabari07@gmail.com, [2]nachiadhi.n@gmail.com, [3]rithishnayagan2004@gmail.com, [4]sivaharishs18@gmail.com, [5]merlinsajini@cit.edu.in

### Abstract

This study presents an smart vehicle infotainment system designed to enhance driving safety through real-time object detection, depth estimation, and motion tracking. The system utilizes a pre-trained YOLOv9 deep learning model, specifically customized to identify objects such as pedestrians, motorcyclists, and vehicles in the Indian road context. The model is deployed on a Raspberry Pi, enabling low-cost, on-device inference without the need for high-end computing hardware. To determine the object proximity, a stereo vision-based depth estimation technique is implemented, allowing the system to calculate the distance of detected objects from the vehicle. Motion detection algorithms are integrated to track the movement of objects across frames, thereby improving detection accuracy and reliability in dynamic environments. The system also features a real-time voice alert module, which notifies the driver when an object is detected within a one-meter range. These alerts can be activated or deactivated based on user preference. A key capability of the system is blind spot detection, which helps the vehicles to identify the areas that are not directly visible to the driver, thereby reducing the risk of side collisions. Overall, the proposed solution demonstrates the feasibility and effectiveness of combining deep learning, embedded vision, and smart alert systems on affordable edge devices for safer driving and potential applications in AI-based autonomous vehicles.

**Keywords:** YOLOv9; Real-Time Object Detection, Depth Estimation, Motion Detection, Voice Alerts, Raspberry Pi, Blind Spot Detection, Autonomous Systems.


*DOI: https://doi.org/10.36548/jtcsst.2025.1.004*
*Received: 26.02.2025, received in revised form: 30.03.2025, accepted: 15.04.2025, published: 24.04.2025*

## 1. Introduction

The integration of artificial intelligence in the vehicle safety systems has significantly changed the automobile industry. Technologies powered by AI, particularly ADAS, have greatly enhanced situational awareness, thereby significantly decreasing human error that leads to fatal accidents. The most vital applications of this area are real-time object detection and distance estimation, which is capable enough to give timely alerts to the drivers and avoid collisions. Real-time object detection is a very important aspect of the modern vehicle and safety systems. Among all deep learning-based detection models, YOLO outperforms in processing due to its very good performance at fast speeds.

The entire process for an image occurs in only one pass, making it applicable for real-time situations, unlike other typical models such as R-CNN or SSD, which process image content by passing over the image multiple times. Among them, YOLOv9 has upgraded with higher features and has its effectiveness for actual real-world driving situations [1]. This study shows a smart vehicle system with YOLO v9 for real-time object detection, distance estimation, and motion tracking. This system was implemented on a Raspberry Pi 4 Model B, and it identifies pedestrians, vehicles, and obstacles by issuing voice alerts for objects within one meter. It has blind-spot detection capabilities that avoid side collisions on behalf of the driver [1].

The system, unlike most other traditional approaches of distance measurements using LIDAR or ultrasonic sensors, uses a formula that estimates distance based on the focal length of the camera, thus making it an efficient and cost-effective solution. The YOLO v9 model was trained on a customized dataset of 900 images labeled to fit localized driving conditions. This custom dataset, being smaller in size compared to big datasets like COCO, assists the model to perform better by providing region-specific traffic scenarios. The training is done on Roboflow, which allows the preprocessing, annotation, and augmentation of the dataset. The trained model is then run on the Raspberry Pi, processing real-time video input from the Raspberry Pi Camera V2, and sends detection result to the infotainment display for the driver's awareness [9,10]. The deployment of deep learning models on edge devices such as Raspberry Pi brings computation challenges because of the hardware constraints. The study optimizes YOLO v9 for minimal latency and efficient real-time processing. Additionally, the system is integrated with an Arduino Uno controlling all the aspects of car movement, emergency braking, and sensor communication. A Motor Driver (L298N) handles the car motion, while for wireless communication it uses a Bluetooth module for enabling remote control as well as the transmission of data. Infotainment also comes with a viewable display which is in a touchscreen, while alternatively the display can be checked remotely with help of smart devices [2].

The important contributions of the research include

i. The integration of YOLOv9 within a vehicle's infotainment system, enabling real-time distance estimation capabilities [3].

ii. The integration of camera-based methodology for accurate distance estimation.

iii Implementation of AI-powered safety features on a Raspberry Pi platform, demonstrating a reduced-cost edge computing solution.

iv. Development of dataset comprising 900 images customized for localized driving scenarios .

v.Optimizations of the YOLOv9 algorithm to achieve real-time processing on low-power hardware platforms

## 2. Proposed System

### 2.1 Object Detection

YOLOv9 makes use of novel strategies to increase accuracy and reduce inefficiencies, that exists on the object detection models. YOLOv9 has four main parts in its architecture: Backbone, Neck, Head, and Auxiliary branch.

Backbone (GELAN): Extracts essential features from input images with Generalized Efficient Layer Aggregation Network (GELAN). GELAN represents an aggregation of CSPNet and ELAN architectures, strategically designed to optimize gradient flow and accelerate inference [11].

Neck: serves as an intermediary between the Backbone and Head, facilitating seamless feature integration and incorporating Spatial Pyramid Pooling (SPP). This design aims to enhance object detection capabilities across a diverse range of scales, thereby improving the accuracy in detecting both small and large objects.

Head: It is used for forecasting bounding boxes, object classes, and probabilities through a decoupled detection head that separates classification and localization for maximum accuracy.

Auxiliary Branch: An innovative aspect of YOLOv9. It retains essential information, avoids information loss in deep networks, and enhances gradient propagation. The Auxiliary Branch is discarded at test time to ensure efficiency.

The Information Bottleneck Principle exposes a fundamental issue in deep learning. With increasing data movement through consecutive layers of a network, the likelihood of information loss is higher. This process is mathematically described as

$$\boldsymbol{I}(\boldsymbol{X},\boldsymbol{X}) \geq \boldsymbol{I}(\boldsymbol{X},\boldsymbol{f}_{\boldsymbol{theta}(X)}) \geq \boldsymbol{I}\left(\boldsymbol{X},\boldsymbol{g}_{\boldsymbol{phi}(\boldsymbol{f}_{\boldsymbol{theta}(X)})}\right) \quad (1)$$

## 2.2 Custom Dataset

The custom dataset is architected to better detect objects within blind spots with the aim to increase road safety in India. In contrast to general-purpose datasets such as COCO, the custom data set contains only the most appropriate objects appearing on Indian roads, providing superior accuracy in the real-world driving scenario.

Implementing an accurate model involves gathering of a diverse dataset by capturing images from vehicle-mounted cameras, ensuring various angles, lighting conditions (day/night), and weather scenarios (fog, rain, clear). The dataset includes the important objects such as motorcycles, pedestrians, auto-rickshaws, stray animals, trucks, and bicycles, which are common in blind-spot incidents. The data was utilized in Roboflow and the bounding boxes for each object was annotated manually, identifying them with appropriate class names. Annotations should be maintained uniformly in order to increase the performance of models. Finally, Roboflow's pre-processing options like image enhancement (rotation, illumination modification, injection of noise), was utilized to get more resilient against environmental conditions of datasets.

## 2.3 Data Collection

The training data used for object detection model construction were created from a mix of sources to maximize relevance and variety. Real-time video was recorded from cameras fitted onto a Bluetooth-driven prototype vehicle to mimic road conditions similar to the sample shown in Figure 1. Additionally, open-source datasets like BDD100K and the India Driving Dataset were also added, featuring an extensive mix of real-world driving scenarios on roads. Moreover, YouTube videos and dashcam videos of Indian roads were employed to simulate actual traffic conditions and behavior. All images and videos were recorded at a resolution of 720p and above for efficient object detection.

**Figure 1.** Sample Dataset [12]

## 2.4 Class Selection

In order to make the detection model suitable for Indian road conditions, 11 to 12 classes of objects were chosen on the basis of occurrence and relevance to daily traffic. These were pedestrians, motorcyclists, cyclists, cars, buses, trucks, and auto rickshaws. The dataset also had animals like stray dogs and cows, which are found on Indian roads. Traffic lights and traffic signs were also marked. Optionally, other classes like speed breakers and potholes were also taken into consideration to offer additional utility ensuring a safe driving assistance.

## 2.5 Data Annotation

The video frames and the collected images were annotated through software such as LabelImg, Roboflow, and CVAT. The annotation was in the YOLO format that can be supported by the YOLOv9 architecture. The text file accompanying each image consisted of bounding box coordinates in the following format: <class_id> <x_center> <y_center> <width> <height>, with values being normalized against the dimensions of the image. This structure is effective for real-time object detection and enables the model to efficiently recognize and localize objects at inference

## 2.6 Data Augmentation

To increase the diversity of the dataset and facilitate model generalization, various data augmentation methods were utilized. Geometric augmentations comprised horizontal flipping, random cropping, and rotation to mimic multiple viewpoints and orientations. Photometric augmentations like brightness and contrast modification, Gaussian blur, and adding noise assisted in mimicking different lighting. Synthetic weather effects like rain and fog were optionally applied to train the model for harsh environments. Also, state-of-the-art augmentation methods such as CutMix and Mosaic were used to merge several images into one, a process supported by YOLOv9 for improved detection of small objects and greater robustness.

## 2.7 Dataset Splitting

The last annotated and enriched dataset was split into three sets to enable effective training and testing. The dataset was split with seventy percent for training, twenty percent for validation, and the rest ten percent for testing. Caution was exercised to ensure class balance in all splits and to include each subset of a mix of scenes under different environmental and light conditions. This distribution facilitated stable model training while maintaining performance to be testable and verifiable under a variety of conditions.

## 2.8 Model summary

Table 1. Illustrates the model summary of the Yolov9 Model

**Table 1.** Model Summary

| Layer Name | Output Shape | Parameters | Remarks |
|---|---|---|---|
| Input | (3, 640, 640) | 0 | RGB image input |
| Backbone (C2f-D) | Varies | ~5.5M | Dense connections for feature learning |
| Neck (PAFPN-E) | Varies | ~3.2M | Multi-scale feature fusion |
| Detection Head | (N x 3 x S x S x C) | ~2.1M | Task-aligned dynamic head (3 scales) |
| **Total Parameters** | **~10.8M** | | For YOLOv9-Small |
| GFLOPs | ~17.0 | | Efficient for real-time applications |
| Inference Speed | ~5–10 FPS (Raspberry Pi 4) | | Varies with input size and hardware |

## 3. Hardware Implementation

The object detection model is used in a prototype Bluetooth controlled car to simulate and analyze the outcome of the model as is shown in Figure 3. Figure 2 illustrates the block diagram of the proposed.

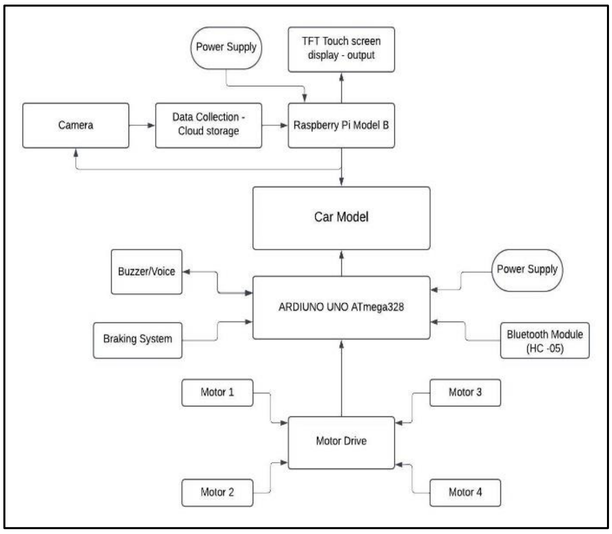


**Figure 2.** Block Diagram

**Figure 3.** Prototype

Figure 2 illustrates the proposed system integrates various components as stated in the figure to facilitate object detection and distance measurement for autonomous vehicles. At the core of the system is a Raspberry Pi 4 Model B, which processes input from a camera running the YOLO v9 object detection model [4]. The model detects objects in real-time and calculates the distance of these objects using the focal length formula. The focal length (pre-calibrated based on the camera specifications) and the height of the detected object in the image will be used for the estimation of the real distance.

$$Distance = \frac{Focal\ length \times Known\ Width}{Pixel\ Width} \tag{2}$$

The system observes and trains from numerous datasets which are stored in cloud. These inputs are fed into the Raspberry Pi, which interfaces with an Arduino UNO ATmega328 to control various actuators and decision-making components, such as the braking system, buzzer and voice alerts, and motor drive. The motor drive controls the vehicle's four motors, enabling autonomous movement based on real-time object detection. Additionally, the system's status and detected objects are shown on a TFT touchscreen, while a Bluetooth module (HC-05) is used for remote control and monitoring through a mobile device. The data related to object detection and vehicle performance are continuously stored in cloud storage to be analyzed and improved in the future for detection algorithms [5], [7], [8].

pyttsx3, a text-to-speech (TTS) engine based on Python, is employed to produce real-time audio warnings for an automobile infotainment system. The system is intended to increase driver alertness by giving voice warnings regarding objects detected, including pedestrians, cars, and blind-spot obstacles. This application is incorporated into the prototype [7].

## 4. Results and Discussions

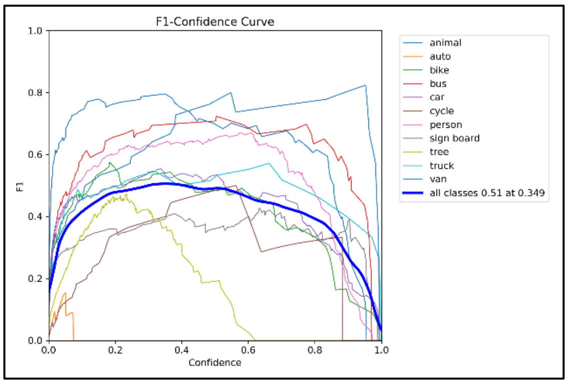


**Figure 4.** F1-Confidence Curve

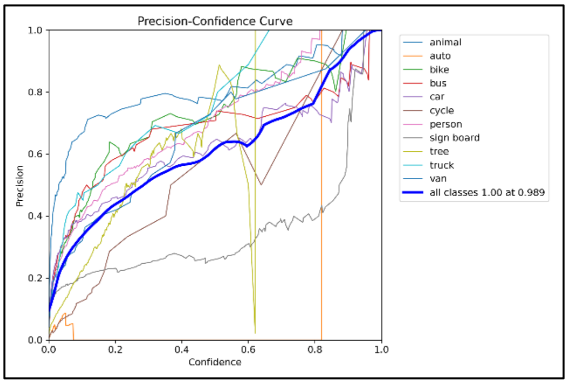


**Figure 5.** Precision-Confidence Curve

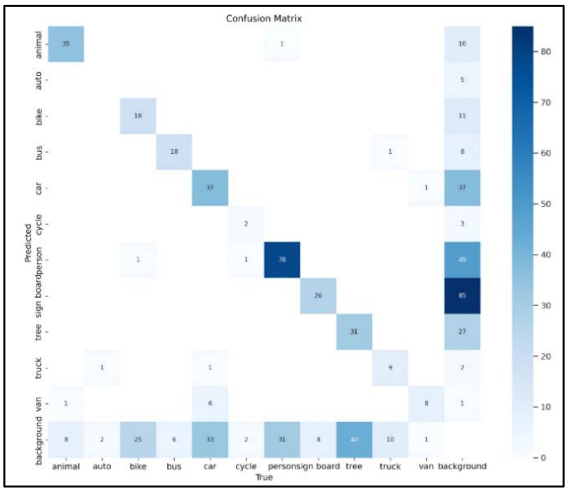


**Figure 6.** Confusion Matrix

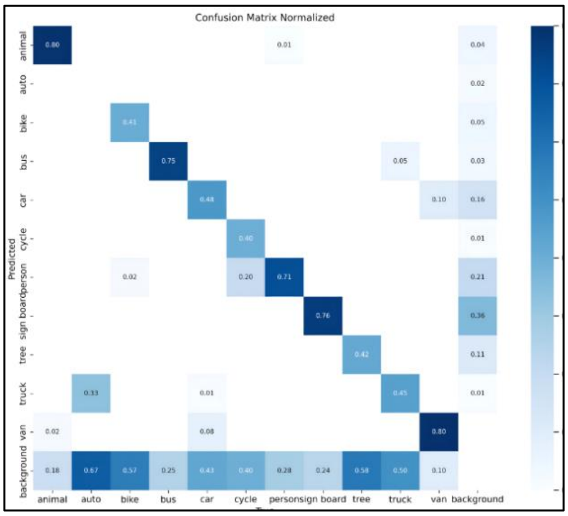


**Figure 7.** Normalized CNF

Figure 4 shows the F1-confidence curve which is the precision-recall tradeoff for various confidence scores. It can be observed from the results of the custom-trained model that F1 scores are different for various classes. Maximum F1 scores are obtained for frequently detected classes such as animals, buses, and vans, while other classes like trees and auto-rickshaws have low F1 scores. The average peak F1 score of all the classes is 0.51 with 0.349 confidence, showing that it is a good balanced model with decent precision-recall tradeoff.

From Figure 5, precision indicates the number of correct objects that are detected. The model has high precision for some classes such as cars, buses, and persons, while other classes such as trees and traffic signs have minimal variability. The general model achieves 100% precision at 0.989 confidence, which indicates that when the model is very confident, its predictions are nearly always accurate. At lower confidence, there is a greater likelihood of false positives.

Figure 6 shows the Confusion matrix which gives insight into detection accuracy class-wise. The best-detected classes in the custom dataset are persons (78 correct detections), cars (37 correct detections), and animals (35 correct detections).

Once the confusion matrix has been normalized, it can be observed from Figure 7 that the model is performing at an 80% accuracy for cars and animals, 76% for persons, and 75% for buses, reflecting robust detection performance. The confusion matrix was utilized to assess the classification accuracy of the YOLOv9 model by comparing the true positives, false positives, and misclassifications over all object classes. The performance assessment assisted in confirming the model's capacity to accurately differentiate between visually close objects like cars, bikes, and buses, even under practical limitations. Importantly, high normalized values on the diagonal suggest precise and sensitive recall for principal classes, which resonates with the aim of enhancing road safety by accurately detecting objects in blind spots and other important areas in real-time.

When the custom-trained YOLOv9 model is compared with a COCO-trained model, it can be seen that the custom model is more optimized for real-time object detection in Indian driving scenarios, especially for blind spot hazard detection. The COCO dataset is meant for generic object detection on a broad range of scenes, but it has poor representation of region-specific traffic items such as auto-rickshaws, local traffic signals, and stray animals, which are very essential for Indian roads. The proposed model is more precise for road-oriented objects such as buses, cars, pedestrians, and animals with a maximum F1 score of 0.51 at 0.349 confidence and 100% precision at 0.989 confidence. Confusion matrix is drawing out excellent detection for principal traffic items but does present some misclassifying among closely similar items, such as  trucks and vans. More generalized models that have been COCO-trained could possibly struggle with these localised vehicle categories as well as with signage, rendering the special-purpose model preferable to deployment on the streets in India for safety measures. Additional enhancements can be made by adjusting the confidence levels and boosting dataset diversity to improve detection performance for underrepresented classes such as trees and traffic signs.

## 5. Conclusion

This work introduced an intelligent car infotainment system with real-time object detection and distance estimation through the YOLOv9 model on a Raspberry Pi platform. The system efficiently improves road safety by giving real-time alerts for obstacles, pedestrians, and vehicles, including blind-spot vehicles, through the use of deep learning, edge computing, and low-cost hardware. The YOLOv9 model was selected specifically for its higher accuracy, faster processing, and better feature extraction compared to its previous generations, making it extremely well-suited for real-time, resource-limited platforms such as Raspberry Pi. Its precision in detecting multiple classes even in region-specific and dense traffic conditions directly helps to  address the intended goal of enhancing driver awareness and reducing blind spot hazards. The proposed model displays significant accuracy in the recognition of objects and estimating distance, superior to general-purpose models under region-specific driving environments. The  approach presents a sustainable option for classical LIDAR and ultrasonic distance estimation technology applied to advanced driver-assistance systems, confirmed using simulation and development of the prototype. Future development will  involve an expansion of the dataset;nd will attempt to speed up the real-time processing and add more AI-based safety features towards further improved vehicle automation and crash avoidance. This work demonstrates the promise of deep learning in car safety and sets the stage for smarter, more efficient, and safer driving experiences.

## References

[1] Khan, Sajjad Ahmad, Hyun Jun Lee, and Huhnkuk Lim. "Enhancing object detection in self-driving cars using a hybrid approach." Electronics 12, no. 13 (2023): 2768.

[2] Prakash, M., M. Janarthanan, and D. Devi. "Multiple Objects Identification for Autonomous Car using YOLO and CNN." In 2023 7th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 597-601. IEEE, 2023..

[3] Pujara, Abhijeet, and Mamta Bhamare. "DeepSORT: real Time & multi-object detection and tracking with YOLO and TensorFlow." In 2022 International conference on augmented intelligence and sustainable systems (ICAISS), pp. 456-460. IEEE, 2022.

[4] Leibe, Bastian, Konrad Schindler, Nico Cornelis, and Luc Van Gool. "Coupled object detection and tracking from static cameras and moving vehicles." IEEE transactions on pattern analysis and machine intelligence 30, no. 10 (2008): 1683-1698.

[5] A. R. Kondelwar et al., "Enhancing road traffic safety with YOLO V7 object detection," Int. J. Novel Res. Dev. (IJNRD), vol. 9, no. 4, pp. a434–a438, Apr. 2024.

[6] Cao, Danyang, Zhixin Chen, and Lei Gao. "An improved object detection algorithm based on multi-scaled and deformable convolutional neural networks." Human-centric Computing and Information Sciences 10, no. 1 (2020): 14.

[7] Singh, Pooja, Pinki Nayak, Arpita Datta, Depanshu Sani, Garima Raghav, and Rahul Tejpal. "Voice control device using raspberry pi." In 2019 Amity International Conference on Artificial Intelligence (AICAI), pp. 723-728. IEEE, 2019.

[8] A. Y. Mahmoud, J. M. Smith, and R. K. Patel, "Using YOLOv9 deep learning model for automatic detection and classification of VHF signals," IEEE Trans. Signal Process., vol. 71, no. 4, pp. 567–578, 2024.

[9] Vashistha, Piyush, Juginder Pal Singh, Pranav Jain, and Jitendra Kumar. "Raspberry Pi based voice-operated personal assistant (Neobot)." In 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA), pp. 974-978. IEEE, 2019.

[10] Maharaja, Amrita, and Namrata Ansari. "Voice controlled automation using raspberry pi." In 2nd International Conference on Advances in Science & Technology (ICAST). 2019.

[11] Buda, Andreea Maria, Delia Bianca Deaconescu, David Vatamanu, and Simona Miclaus. "Using YOLOv9 Deep Learning Model for Automatic Detection and Classification of VHF Signals." In 2024 International Symposium on Electronics and Telecommunications (ISETC), pp. 1-5. IEEE, 2024.

[12] https://www.moonstone.co.za/new-driving-demerit-system-what-will-the-impact-be-on-insurance/