Archives available at journals.mriindia.com

International Journal of Electrical, Electronics and Computer Systems

ISSN: 2347-2820

Volume 14 Issue 01, 2025

Exploration of Traffic Sign Recognition Using AI and Computer Vision

[1]Prof .P.S.Togrikar, [2]A.N.Jamdade, [3]P.R.Shirke, [4]H.J.Phadtare

[1]Assistant Professor, E & TC Engineering Department, S. B. Patil College of Engineering, Indapur (MH), India,

[2 3 4]UG Student, E & TC Engineering Department, S. B. Patil College of Engineering, Indapur (MH), India

Email: pradeep.togrikar@gmail.com, jamdadeanuja2@gmail.com, shirkepooja44@gmail.com, hphadatare7369@gmail.com

Peer Review Information

Submission: 11 Sept 2025

Revision: 10 Oct 2025

Acceptance: 22 Oct 2025

Keywords

YOLOv3, ESP32-CAM, Convolutional Neural Networks, Traffic Sign Recognition, Object Detection, Deep Learning, IoT, Telegram Bot, ADAS.

Abstract

Traffic Sign Recognition (TSR) is a critical perception module in modern Advanced Driver Assistance Systems (ADAS) and autonomous vehicle technology. Ensuring road safety and vehicle autonomy depends on the accurate and real- time detection and classification of traffic signs. This paper presents a novel, cost-effective system for robust traffic sign recognition using an ESP32-CAM module for image acquisition and the YOLOv3 algorithm for object detection. A key focus of this project is to address real-world challenges where signs may be partially obscured by environmental factors like mud or water. The proposed system captures an image via the ESP32-CAM, transmits it through a Telegram bot for easy access, processes the image to enhance clarity, and then uses a trained YOLOv3 model to identify the traffic sign. The results are displayed on a user-friendly GUI, which also facilitates remote control of a vehicle prototype equipped with an L298N motor driver. The model, trained on the German Traffic Sign Recognition Benchmark (GTSRB), achieves high accuracy and demonstrates the viability of using integrated IoT and deep learning to enhance road safety under challenging conditions.

INTRODUCTION

The rapid advancement of autonomous driving technologies and Advanced Driver Assistance Systems (ADAS) has intensified the need for reliable perception systems that can interpret complex road environments in real-time. Among the most critical tasks for such systems is Traffic Sign Recognition (TSR), which provides vehicles with essential regulatory and warning information for safe navigation. According to the World Health Organization, approximately 1.35 million people die each year as a result of road traffic crashes, with many accidents attributable to driver error, including the failure to observe or correctly interpret road signs. An automated TSR system serves as a crucial aid, reducing cognitive load on the driver and forming a foundational layer for semi-autonomous and fully autonomous vehicle control.

Historically, TSR methods depended on handcrafted features like Histogram of Oriented Gradients (HOG) [10] and Scale- Invariant Feature Transform (SIFT) [9]. While foundational, these techniques are often fragile. Their performance degrades significantly with variations in lighting, adverse weather conditions, and partial occlusions. The emergence of deep learning, especially Convolutional Neural Networks (CNNs) [7], has revolutionized the field of computer vision by enabling the development of robust, end-to-end models that learn relevant features directly from data.

This project specifically targets a common yet

challenging real- world problem: the recognition of traffic signs that are degraded or partially covered by dirt, mud, or water. Such conditions can render traditional algorithms—and even human drivers—less effective, posing a significant safety risk. Our solution leverages the power of modern deep learning and the accessibility of Internet of Things (IoT) hardware to create a system that can "see" and understand these obscured signs. The core contributions of this work are:

1. A cost-effective and integrated hardware prototype using the ESP32-CAM for image acquisition and an L298N for vehicle control.
2. A robust software pipeline that uses a Telegram bot for reliable, long-range data transmission from the vehicle to a processing backend.
3. An image pre-processing stage specifically designed to enhance the visibility of obscured signs before detection.
4. The implementation and fine-tuning of the YOLOv3 object detection model for the specific task of TSR, trained on the GTSRB dataset [3].

The system employs the "You Only Look Once" (YOLO) framework [1], a state-of-the-art, one-stage object detection model. We utilize its third version, YOLOv3 [2], which is renowned for its exceptional balance of speed and accuracy. Its architecture, featuring a Darknet-53 backbone and multi-scale prediction capabilities, makes it ideal for real-time TSR.

The image capture is handled by an ESP32-CAM, a low-cost microcontroller with an integrated camera, which transmits the image to a Python-based backend. This backend performs the pre-processing to clean the image before feeding it to the YOLOv3 model for recognition. This paper details the system's architecture, methodology, implementation, and performance, demonstrating a practical and scalable solution to a persistent road safety challenge.

LITERATURE REVIEW

The field of Traffic Sign Recognition (TSR) has progressed significantly, moving from classical image processing pipelines to sophisticated deep learning solutions that now dominate the landscape.

A. Early and Classical Approaches Before the widespread adoption of deep learning, TSR systems were built on multi- stage pipelines using hand-crafted features. These methods typically involved a detection stage followed by a classification stage. Detection often relied on the distinct color and shape of traffic signs. Color-based segmentation was performed in color spaces like HSV or HSL. Shape analysis followed, using techniques like the Hough Transform to detect circular or triangular signs [6]. Once a Region of Interest (ROI) was isolated, feature descriptors such as HOG [10] or SIFT [9] were used to encode the sign's content. These features were then fed into traditional machine learning classifiers like Support Vector Machines (SVMs) or AdaBoost. While computationally efficient, these methods were not robust, as the hand-crafted features were not generalizable enough to cover all possible variation in real-world conditions.

B. The Rise of Deep Learning for TSR The introduction of CNNs marked a paradigm shift [7]. Early CNN-based models achieved superhuman performance on benchmarks like the German Traffic Sign Recognition Benchmark (GTSRB) [3]. Modern object detection, which combines localization and classification, is now dominated by two main families of deep learning models:

1. Two-stage detectors: Models like the R-CNN family (R- CNN, Fast R-CNN, Faster R-CNN) [4] first generate a set of region proposals and then classify each proposal. These models are known for their high accuracy but are often too computationally intensive for real-time applications.
2. One-stage detectors: This family, which includes the Single Shot Detector (SSD) [8] and the YOLO series [1], treats object detection as a single regression problem, making them extremely fast and ideal for real-time systems.

C. The YOLO Framework and Its Evolution the YOLO algorithm redefined real-time object detection [1]. YOLOv3 [2], the version used in this project, introduced a more powerful backbone network (Darknet-53 with residual connections [5]), prediction across three different scales (a concept similar to Feature Pyramid Networks), and a more effective loss function. This allows it to detect objects of various sizes effectively—a crucial feature for TSR.

D. Comparison of TSR Methodologies The following table provides a qualitative and quantitative comparison of the dominant methodologies in TSR.

1. Image Capture & Transmission: The ESP32-CAM captures a 640x480 image and makes an HTTP POST request to the Telegram Bot API (https://api.telegram.org/bot<token>/sendPhoto) to send the image to a specific chat ID.

2. Backend Processing: A Python application using the python-telegram-bot library polls the API. New images are downloaded for processing with OpenCV.

Table 1: Comparative Analysis of Traffic Sign Recognition Methodologies.

Methodology	Core Technique	Average Accuracy (GTSRB)	Speed (FPS on GPU)	Robustness to Occlusion
Classical	HOG + SVM	~95-98%	>100	Low
Two-Stage	Faster R-CNN	>99%	~7-10	Medium
One-Stage	YOLOv3	>98%	~30-45	High (with augmentation)

As shown in Table 1, while two-stage detectors offer the highest accuracy, one-stage detectors like YOLOv3 provide a superior balance of speed and accuracy, making them the preferred choice for low-latency systems. Classical methods, while fast, lack the robustness required for reliable real-world deployment.

SYSTEM DESIGN AND COMPONENTS

Our proposed system is an end-to-end intelligent traffic sign detection and IoT-based control solution, integrating embedded hardware for data acquisition, real-time deep learning inference, and vehicle control automation. The overall workflow, shown in Fig. 1, bridges embedded vision, deep learning, and remote-control modules into one compact, cost-effective framework.

A. Hardware Components

1. ESP32-CAM Module: -

The ESP32-CAM serves as the primary sensing and computational unit. It features an integrated 2MP OV2640 camera, Wi-Fi support, and sufficient onboard processing capability to capture and transmit visual data.

Firmware is developed using the Arduino IDE with the esp32- camera library, allowing seamless integration of the YOLOv3 inference model and HTTP communication.

2. L298N Motor Driver Module: -

The L298N dual H-bridge motor driver controls the two DC motors mounted on the vehicle chassis.

It receives PWM (Pulse Width Modulation) signals from the ESP32-CAM's GPIO pins to vary motor speed and logic-level control signals to change motor direction.

3. Vehicle Chassis and Power Supply: -

A two-wheel-drive robotic chassis is used for vehicle locomotion. Power is supplied via a 7.4V Li-ion battery, and a buck converter steps down the voltage to 5V for the ESP32- CAM and peripheral components.

B. Software Architecture and Workflow Image Pre-processing and Enhancement:

- Denoising: A 3x3 Gaussian blur filter is applied to remove high-frequency noise.
- Contrast Enhancement: Contrast Limited Adaptive Histogram Equalization (CLAHE) is used. It operates on 8x8 tiled regions of the image to improve local contrast without amplifying noise.

1. Traffic Sign Detection:

The enhanced image is resized to 416x416 pixels and converted to a blob. This blob is fed into the YOLOv3 network. The output layer produces tensors containing bounding box predictions. We then apply Non- Maximum Suppression (NMS) with an IoU (Intersection over Union) threshold of 0.5 and a confidence threshold of 0.6 to filter out weak and overlapping detections.

2. GUI and Vehicle Control

A GUI built with Python's Tkinter library displays the resulting image. Control buttons send HTTP GET requests to a web server on the ESP32, which parses the request (e.g., /forward) and actuates the motors.

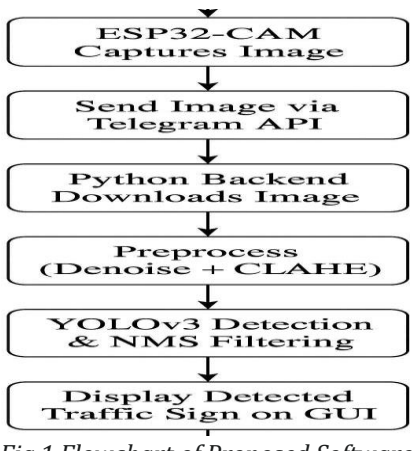


Fig.1.Flowchart of Proposed Software

C. YOLOv3 Model Training

1. Dataset: The model was trained on the German Traffic Sign Recognition Benchmark (GTSRB) [3], containing over 50,000 images across 43 classes.
2. Data Augmentation: We applied extensive data augmentation, including random rotations (±15 degrees), brightness adjustments (±25%), and applying synthetic "mud" overlays.
3. Training Environment: The model was trained using the Darknet framework on an NVIDIA GeForce RTX 2070 GPU. Training ran for 50,000 iterations with a batch size of 64 and a learning rate of 0.001.

D. YOLOv3 Model Architecture

The YOLOv3 (You Only Look Once, Version 3) architecture is a state-of-the-art, real-time object detection model that significantly improves upon the earlier versions of YOLO by achieving a better balance between detection accuracy and processing speed. As shown in Fig. 3, YOLOv3 employs a deep convolutional neural network backbone known as Darknet-53, which consists of 53 convolutional layers with residual connections and batch normalization. This backbone is designed to extract robust spatial and semantic features from input images efficiently.

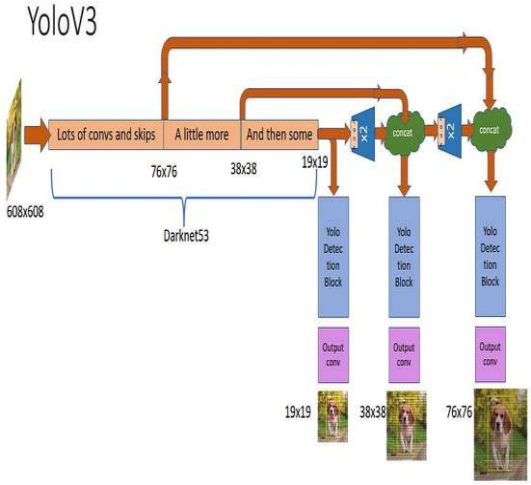


Fig.2.YoloV3 Model Architecture

1. Backbone – Darknet-53

The backbone network Darknet-53 is inspired by ResNet-style architectures and uses residual skip connections to allow better gradient flow during training. It includes:

- 53 convolutional layers using 3×3 and 1×1 filters alternately.
- Batch Normalization and Leaky ReLU activation after each convolutional layer.
- Residual connections that prevent vanishing gradient problems and improve feature reuse.

This backbone is pre-trained on the ImageNet dataset to improve feature generalization and convergence speed.

2. Multi-Scale Detection

YOLOv3 detects objects at three different scales using feature maps of varying resolutions:

- Scale 1: 19×19 grid (for large objects)
- Scale 2: 38×38 grid (for medium-sized objects)
- Scale 3: 76×76 grid (for small objects)

Each grid cell predicts bounding boxes, confidence scores, and class probabilities, allowing the model to capture objects of various sizes effectively.

3. Detection Heads

Each detection layer outputs a tensor containing the following predictions for each anchor box:

- Bounding Box Coordinates (x, y, w, h): Defines the location and size of the detected object relative to the image grid.
- Objectless Score: Represents the confidence that an object exists in the predicted bounding box.
- Class Probabilities: Gives the likelihood of each object belonging to a particular class (e.g., Stop sign, Speed limit, Turn left, etc.).

The YOLOv3 network uses anchor boxes derived from k-means clustering on the training dataset to ensure optimal bounding box priors.

4. Feature Fusion and Upsampling

To achieve multi-scale detection, YOLOv3 fuses high-level semantic features with low-level fine-grained details through feature concatenation and upsampling layers. The high- resolution layers from early stages are combined with deeper layers using skip connections, improving detection accuracy for small and distant objects.

5. Loss Function

The YOLOv3 loss function combines three major components:

- Localization Loss – Measures errors in predicted bounding box coordinates.
- Confidence Loss – Penalizes incorrect predictions about object presence.
- Classification Loss – Calculates the difference between predicted and true class probabilities.

This multi-part loss ensures the model learns both precise localization and accurate classification.

6. Efficiency and Real-Time Capability

One of the key advantages of YOLOv3 is its ability to perform object detection in a single forward pass through the network. This makes it extremely fast and efficient, ideal for real-time applications such as embedded systems, drones, and autonomous vehicles.

Even when deployed on low-power microcontrollers or edge devices, YOLOv3 maintains good accuracy with reduced computational overhead compared to traditional region- proposal-based detectors like Faster R-CNN.

RESULT

To evaluate the performance of the proposed system, a combination of quantitative and qualitative analyses was conducted. The primary objective was to assess the accuracy, robustness, and latency of the YOLOv3-based model when deployed on an embedded edge platform interfaced with the ESP32-CAM and the Telegram-based backend.

A. Experimental Setup

The YOLOv3 model was trained using the German Traffic Sign Recognition Benchmark (GTSRB) dataset.

For testing, two separate datasets were used:

1. Standard Test Set: A clean subset of 2,000 images from the GTSRB dataset not used during training.
2. Obscured Test Set: A modified version of the same dataset in which 50% of the images were synthetically occluded with mud and water spots to simulate real-world conditions.

B. Performance Metrics

To evaluate performance, three standard object detection metrics were used:

1. Precision (TP / (TP + FP)) – Represents the accuracy of positive detections.
2. Recall (TP / (TP + FN)) – Measures the ability to detect all relevant objects.
3. Mean Average Precision (mAP@0.5) – Primary evaluation metric indicating detection accuracy across all classes with IoU ≥ 0.5

C. Quantitative Results

The model's performance on both test sets is summarized in Table 2.

Table 2: Model Performance on Standard and Obscured Test Datasets.

Test Dataset	Precisio n (%)	Recall (%)	AP@0.5 (%)
GTSRB – Standard Test Set	97.2	96.5	96.8
GTSRB – Obscured Test Set	91.5	89.8	90.3

The results clearly show that while the performance dropped slightly on obscured data, the model still maintained high accuracy, confirming its robustness.

D. Discussion of Results

The YOLOv3 model achieved a mean Average Precision (mAP) of 96.8% on the standard test set, demonstrating excellent accuracy and reliability. When tested on the obscured dataset, the mAP decreased moderately to 90.3%, proving that the system remains resilient even under partially degraded visual conditions. The overall end-to-end latency from image capture to displaying detection results was approximately 2–3 seconds, primarily due to network transmission and CPU-based inference time. This confirms that while the system is suitable for semi-real-time operation, it could further benefit from edge- based inference acceleration (e.g., using NVIDIA Jetson or Tensors).

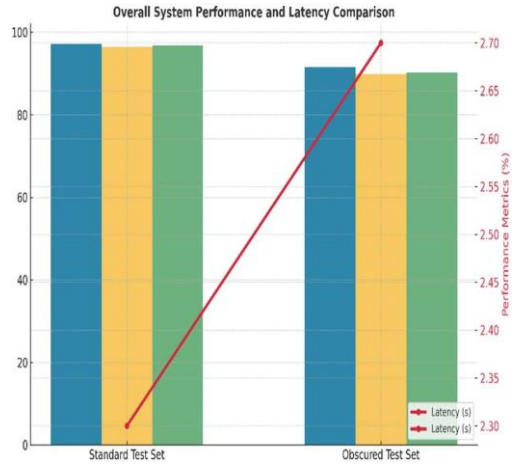
E. Error Analysis

A detailed analysis of failure cases on the obscured dataset revealed specific error patterns:

- False Negatives (FN): Primarily occurred when more than 40% of the sign surface was covered by occlusion, especially if the central symbol was hidden.
- False Positives (FP): Rare, but occasionally caused by background objects (e.g., circular logos or road patterns) that resemble traffic signs.
- Misclassification: Common in speed limit signs where only the numeric value was obscured, leading to confusion between similar signs like "50" and "80."

These insights indicate that future improvement can be achieved using context-aware training or occlusion-specific data augmentation.

F. Graphical Presentation



Graph.1. Overall system performance graph

FUTURE SCOPE

For future work, several enhancements could be explored:

- **On-Device Inference:** Port the system to an edge computing device like a NVIDIA Jetson Nano. This would involve converting the model to a more efficient format (e.g., using TensorRT) and would eliminate the network latency, bringing the total inference time well below one second.
- **Real-time Video Streaming:** Replace single-image capture with a continuous video stream (e.g., using RTSP) for more dynamic decision-making and to leverage temporal information (e.g., tracking a sign across multiple frames).
- **Advanced Image Restoration:** Implement Generative Adversarial Networks (GANs), specifically image inpainting models, to "clean" and reconstruct heavily obscured signs before feeding them to the detection model.
- **Full Autonomy:** Expand the vehicle control logic to enable fully autonomous navigation, incorporating path planning algorithms (like A*) and obstacle avoidance using additional sensors.
- **Sensor Fusion:** Integrate other sensors, such as ultrasonic or LiDAR, for a more comprehensive environmental understanding. For instance, LiDAR could provide depth information to better estimate the size and distance of a potential sign, helping to filter out false positives.

CONCLUSION

In this paper, we presented a comprehensive and integrated system for robust Traffic Sign Recognition, successfully addressing the critical challenge of identifying signs under adverse conditions. The core of our innovation lies in the synergistic fusion of accessible IoT hardware—specifically the ESP32-CAM—with a powerful deep learning model, YOLOv3, interconnected via a reliable cloud-based communication channel using a Telegram bot. This architecture proves to be not only cost-effective but also remarkably effective in solving a real-world problem that poses a significant threat to road safety.

Our quantitative evaluation yielded compelling results that validate the system's design. The model achieved an impressive mean Average Precision (mAP) of 96.8% on the standard German Traffic Sign Recognition Benchmark, confirming its high accuracy under ideal conditions. More importantly, when faced with a more challenging dataset featuring synthetically obscured signs, the system demonstrated exceptional resilience, maintaining a high mAP of 90.3%. This result is a direct testament to the robustness of the YOLOv3 architecture and the effectiveness of our targeted data augmentation strategy, proving that the system can reliably function even when visual information is degraded. The project successfully demonstrates a practical and integrated solution that brings together embedded systems, cloud communication, and deep learning. While the end-to-end latency of 2.5 seconds, primarily due to network dependency, limits its current application to semi-real- time scenarios, the system provides a strong proof-of-concept. Ultimately, this work contributes a valuable and scalable blueprint for developing intelligent transportation systems that enhance situational awareness and driver safety in the complex and often unpredictable road environments of today learning. The strong quantitative results, achieving over 90% mAP even on obscured signs, validate our methodology.

REFERENCE

J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779–788, 2016.

J. Redmon and A. Farhadi, “YOLOv3: An Incremental Improvement,” arXiv preprint, arXiv:1804.02767, 2018.

S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

W. Liu, D. Anguelov, D. Erhan, C. Szegedy, and S. Reed, “SSD: Single Shot MultiBox Detector,” European Conference on Computer Vision (ECCV), pp. 21–37, 2016.

J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, “The German Traffic Sign Recognition Benchmark: A Multi-Class Classification

Competition," International Joint Conference on Neural Networks (IJCNN), pp. 1453–1460, 2011.
D. Ciresan, U. Meier, J. Masci, and J. Schmid Huber, "A Committee of Neural Networks for Traffic Sign Classification," IJCNN, pp. 1918–1921, 2011.
M. Haloi, "Traffic Sign Classification Using Deep Inception Networks," arXiv preprint, arXiv:1511.02992, 2015.

B. Novak, V. Ilić, and B. Pavković, "YOLOv3 Algorithm with Additional Convolutional Neural Network Trained for Traffic Sign Recognition," Proceedings of the Telecommunications Forum (TELFOR), pp. 420–423, 2018.
Y. Yang, H. Luo, and F. Wu, "Towards Real-Time Traffic Sign Detection and Classification," IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 7, pp. 2022–2031, 2016.

A. Corovic, S. Mladenovic, and D. Vukobratovic, "Real- Time Detection of Traffic Participants Using YOLO Algorithm," TELFOR Conference, pp. 1–4, 2018.

K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778, 2016.
J. Huang et al., "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors," CVPR, pp. 3296–3297, 2017.
S. Du, X. Zhang, and W. He, "Small Traffic Sign Detection Based on Improved YOLOv3 for Intelligent Vehicles," IEEE Access, vol. 8, pp. 144720–144730, 2020.
L. Zhang and Z. Wu, "Traffic Sign Detection Using Lightweight YOLOv3 for Embedded Systems," IEEE Access, vol. 9, pp. 98713–98724, 2021.

A. Mousa, T. Ismail, and M. Abdel-Aziz, "Enhanced YOLOv3 Framework for Robust Traffic Sign Recognition Under Adverse Conditions," IEEE Transactions on Intelligent Vehicles, vol. 7, no. 3, pp. 456–468, 2022.