Real-Time Edge-Based Road Anomaly Detection Using YOLOv5 and Statistical Validation on Raspberry Pi 4

Sanji Krishna M P(Team Lead)
Ahamed Faisal A
Subiksha A

Department of Electronics and Communication Engineering
Rajalakshmi Engineering College

Project Guide: Dr. S. Chitra
Professor, Department of ECE
Rajalakshmi Engineering College

Abstract

Road surface anomalies such as potholes and cracks significantly impact transportation safety and infrastructure reliability. Manual inspection techniques are labor-intensive and unsuitable for continuous large-scale monitoring. This work presents a real-time edge artificial intelligence (AI) based road anomaly detection system deployed on Raspberry Pi 4 using central processing unit (CPU)-only inference. The system utilizes You Only Look Once version 5 (YOLOv5) converted to Open Neural Network Exchange (ONNX) format and executed using ONNX Runtime (ORT).

The architecture integrates structured frame preprocessing, confidence filtering, Intersection over Union (IoU)-based Non-Maximum Suppression (NMS), sliding window temporal voting, and a statistical fallback detection mechanism using Mahalanobis Distance and Z-score analysis. The optimized implementation achieves approximately 7 frames per second (FPS), representing a seven-fold improvement over the baseline configuration. The proposed framework demonstrates that deep learning-based video analytics can be effectively deployed on resource-constrained edge hardware without additional accelerators.

Keywords: Edge Artificial Intelligence, Road Anomaly Detection, YOLOv5, ONNX Runtime, Raspberry Pi Deployment, Non-Maximum Suppression, Temporal Voting, Mahalanobis Distance, Real-Time Inference, Embedded Vision Systems

1 Introduction

Road surface degradation is a major contributor to traffic accidents, vehicle damage, and increased maintenance expenditure. Traditional inspection methods rely on manual surveys, which are subjective and not scalable. Recent advances in deep learning have enabled automated anomaly detection; however, most implementations depend on GPU acceleration. Deploying such systems on low-power embedded devices remains challenging. This work focuses on designing a computationally efficient, CPU-optimized road anomaly detection framework capable of near real-time inference on Raspberry Pi 4.

2 Related Work

Object detection models such as Faster R-CNN, Single Shot Detector (SSD), and YOLO have been extensively used for surface inspection tasks. Two-stage detectors provide high precision but demand significant computational resources. Single-stage detectors such as YOLOv5 offer faster inference with acceptable accuracy, making them suitable for edge applications.

Most existing implementations process data in the cloud, which introduces latency and bandwidth constraints. Edge computing reduces response time and improves system autonomy. However, maintaining detection stability under limited computational resources remains an open research challenge. This work addresses these challenges through structured optimization and statistical validation.

3 Objectives

- To achieve inference performance greater than 5 frames per second on Raspberry Pi 4 using CPU-only deployment.
- To reduce false positive detections using IoU-based Non-Maximum Suppression.
- To enhance temporal stability using sliding window validation.
- To detect unseen anomalies using statistical deviation techniques.
- To design a modular and scalable edge-based monitoring architecture.

4 System Architecture

The system follows a structured multi-stage processing pipeline:

Video Input → Frame Preprocessing → ONNX Inference → Post-Processing → NMS → Temporal Voting → Statistical Fallback → Logging and Visualization.

Each stage incrementally refines the detection results to ensure accuracy and robustness.

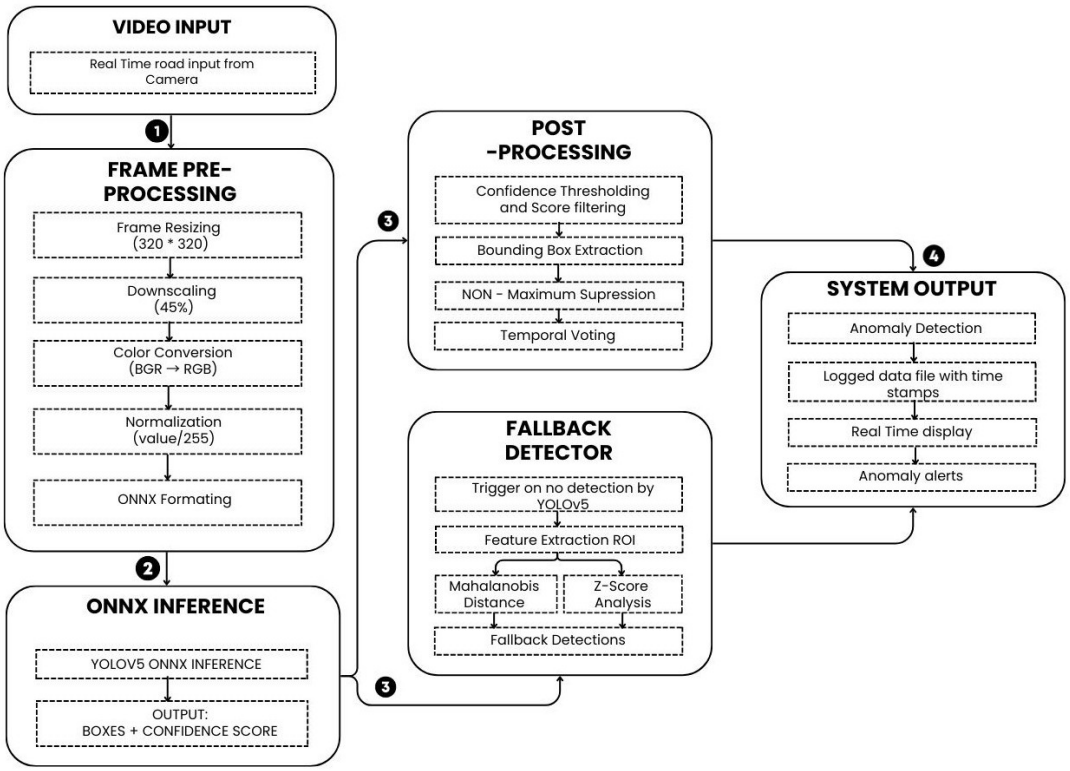

Figure 1: Proposed Edge-Based Road Anomaly Detection Architecture

5 Methodology

5.1 Frame Preprocessing

To reduce computational complexity, each frame is downscaled to 45% of its original resolution and resized to 320×320 pixels. Pixel intensity normalization is performed as:

$$x_{norm} = \frac{x}{255} \tag{1}$$

where x represents the original pixel value and x_{norm} is the normalized value between 0 and 1. This normalization ensures numerical stability during neural network inference.

5.2 Bounding Box Representation

The YOLOv5 detector outputs bounding boxes represented as:

$$B = \{(x, y, w, h, c)\} \tag{2}$$

Here, (x, y) denote the center coordinates, w and h represent width and height, and c denotes the confidence score associated with the detection.

5.3 Intersection over Union

Duplicate bounding boxes are removed using IoU:

$$IoU = \frac{Area_{intersection}}{Area_{union}} \tag{3}$$

If the overlap between two boxes exceeds a predefined threshold, the box with lower confidence is suppressed.

5.4 Temporal Voting

Detection validation across frames is performed using:

$$Detection_{valid} = \begin{cases} True & \text{if votes} \geq 2 \\ False & \text{otherwise} \end{cases} \tag{4}$$

This ensures that only persistent detections across consecutive frames are confirmed.

5.5 Mahalanobis Distance

Statistical deviation is computed as:

$$D_M(x) = (x - \mu)^T \Sigma^{-1} (x - \mu) \tag{5}$$

where μ represents the mean feature distribution and Σ represents covariance. Larger values indicate deviation from normal road texture.

5.6 Z-Score Analysis

Intensity deviation is measured using:

$$Z = \frac{|x - \mu|}{\sigma} \tag{6}$$

where σ is the standard deviation. A high Z-score suggests abnormal surface characteristics.

6 Software Architecture

The system is implemented using modular Python scripts.

runvid.py handles video capture, preprocessing, inference execution, visualization, and timestamp logging.

mahalanobis_detector.py performs feature extraction and statistical distance computation.

fallback_anomaly.py implements Z-score based anomaly confirmation.

7 Hardware Platform

Target Hardware: Raspberry Pi 4
CPU: Quad-core Cortex-A72
RAM: 4GB
Inference Mode: CPU-only

The absence of GPU acceleration emphasizes computational optimization strategies.

8 Optimization Techniques

Frame downscaling reduces pixel-level operations, achieving approximately 2× improvement in processing speed. ONNX Runtime graph optimization enables operator fusion and reduces redundant computation. Memory optimization through rolling temporal buffers minimizes allocation overhead. Combined, these strategies resulted in performance improvement from 1 FPS to 7 FPS.

9 Experimental Evaluation

Table 1: Evaluation on Different Devices

Device	Initial FPS	Optimized FPS
Laptop	25–45	25–45
Raspberry Pi 4	1	7

The optimized deployment achieved 7 FPS, meeting the near real-time target while maintaining detection stability.

10 Limitations

Performance may degrade under extreme low-light conditions. CPU-only inference restricts scalability to larger models. Quantization was not implemented and remains a potential area for enhancement.

11 Conclusion

This work presents a real-time edge artificial intelligence-based road anomaly detection system successfully deployed on Raspberry Pi 4 using CPU-only inference. Through structured preprocessing, ONNX optimization, Non-Maximum Suppression, temporal voting, and statistical fallback mechanisms, the system achieves approximately 7 frames per second with stable detection performance. The seven-fold performance improvement demonstrates the importance of computational optimization for embedded AI deployment. Future enhancements include integer quantization, deployment on Raspberry Pi 5, and integration with cloud-based monitoring systems. The proposed system supports sustainable infrastructure development and contributes toward intelligent transportation systems.