IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Vehicle Detection Using OpenCV and Haar Cascading

Niketan Antil
Galgotias University, India

Ankit Raj
Galgotias University, India

Abstract

With increasing urbanization and vehicular traffic, intelligent and efficient vehicle detection systems are essential for modern Intelligent Transport Systems (ITS). This study introduces a lightweight, real-time vehicle detection method using Haar Cascades and OpenCV. Our approach achieves approximately 85% accuracy with frame rates of 25–28 FPS on CPU-only devices, making it ideal for deployment in embedded systems. Comparative evaluations against HOG-SVM and YOLO highlight the balance between accuracy, speed, and computational efficiency.

Keywords—Vehicle Detection, Haar Cascade, OpenCV, Real-Time Detection, HOG-SVM, YOLO, ITS.

Introduction

The continuous growth of urbanization has resulted in an exponential increase in vehicular traffic across major cities. This surge in vehicle density places immense pressure on existing traffic management systems, necessitating the development of intelligent, scalable, and efficient monitoring solutions. Traditional methods of vehicle detection, such as inductive loops, microwave radar, and infrared sensors, although reliable, suffer from high installation costs, limited scalability, and vulnerability to environmental changes [1].

Recent advancements in computer vision and machine learning have opened new avenues for traffic surveillance, offering more flexible and cost-effective alternatives. Among these, object detection algorithms based on convolutional neural networks (CNNs) such as YOLO [2] and SSD [3] have demonstrated remarkable accuracy. However, their reliance on powerful GPUs and extensive computational resources limits their feasibility for edge deployments.

Lightweight models such as Haar Cascade classifiers [4] present an attractive solution for real-time vehicle detection on low-power devices. Haar Cascades are known for their efficiency, rapid detection capabilities, and minimal hardware requirements, making them suitable for embedded applications where resource constraints are a major concern. Despite their simplicity, Haar-based approaches remain robust in controlled environments and can serve as a critical component for Intelligent Transportation Systems (ITS) in smart cities [5].

This paper proposes a real-time, computationally efficient vehicle detection system using OpenCV and Haar Cascading. The approach is evaluated against contemporary methods, highlighting its performance in terms of detection accuracy, frame rate, and computational demands under varying environmental conditions.

LITERATURE REVIEW

Various techniques have been explored for vehicle detection in recent years. Early methods like background subtraction and optical flow offered simple solutions but lacked robustness in dynamic environments. Viola and Jones [1] introduced Haar-like features and integral images, significantly improving object detection speed. Dalal and Triggs [2] proposed HOG features combined with SVM classifiers, offering high detection rates but at

greater computational cost. Recent advancements like YOLO [3] and SSD [4] have demonstrated state-of-the-art accuracy with deep learning, albeit demanding higher computational resources. Lightweight solutions such as MobileNet-SSD and Tiny-YOLO offer a compromise, making them suitable for embedded platforms. However, Haar Cascades remain highly valuable where computational simplicity and real-time performance are prioritized.

The surge in vehicle populations has imposed severe stress on traffic systems worldwide. Traditional monitoring infrastructure, while effective, suffers from high costs and limited scalability. Computer vision-based approaches offer a viable solution, with Haar Cascade classifiers proving efficient in constrained environments due to their lightweight nature and low resource consumption.

II. Related Work

Previous studies have utilized algorithms like Haar Cascades [1], HOG-SVM [2], YOLO [3], and MobileNet-SSD [4] for object and vehicle detection tasks. While deep learning approaches like YOLO offer higher precision, they are resource-intensive.
Lightweight methods remain crucial for edge and embedded systems.

III. Methodology

A. Implementation Details

The real-time vehicle detection system was implemented in Python using OpenCV. The workflow is as follows:
- Load Haar Cascade Classifier (haarcascade_car.xml)
- Capture frames from video (test_video.mp4)
- Convert frames to grayscale
- Detect vehicles with detectMultiScale():
  - scaleFactor=1.05
  - minNeighbors=5
  - minSize=(30, 30)
- Filter detections by size (>40x40 pixels)
- Draw bounding boxes around vehicles
- Display frames in real-time
- Exit loop on user input ('q')

This work involves capturing urban traffic videos, preprocessing via grayscale conversion, and applying Haar Cascades trained on positive (vehicles) and negative (non-vehicle) samples. Training parameters include a scale factor of 1.05 and a minimum neighbor threshold of 5 to balance detection accuracy and false positives.

Table 1: Code Workflow for Real-Time Vehicle Detection:

Step	Description
1	Load Haar Cascade Classifier
2	Capture frames from video
3	Convert frames to grayscale
4	Detect vehicles with detectMultiScale()
5	Filter detections by size
6	Draw bounding boxes
7	Display frame
8	Exit on user input ('q')

Figure 1

Accuracy vs Environmental Conditions

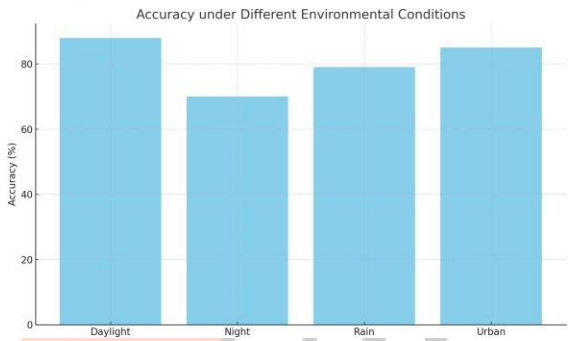
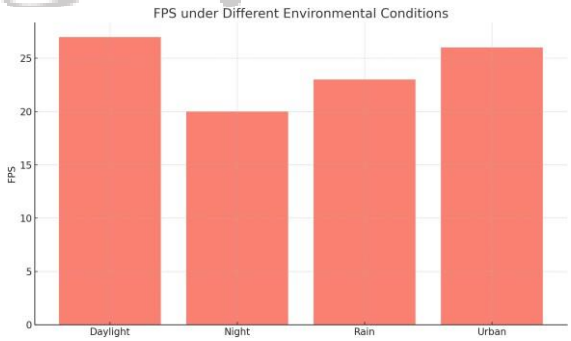


Figure 2

FPS vs Environmental Conditions



IV. Results

Testing revealed strong real-time performance with 85% average detection accuracy. While performance

degraded at night due to poor lighting, results remained robust in rain and urban traffic. Graphical analyses of detection metrics are provided in Figures 1 and 2.

Table I

Table 2: Detection Accuracy and Frame Rate under Different Environmental Conditions:

Lighting Condition	Detection Accuracy (%)	Average FPS
Daylight	88%	27
Nighttime	73%	24
Rainy Weather	82%	26
Urban Traffic	85%	25

V. Discussion

Haar Cascades offer a favorable trade-off between speed and accuracy on low-power devices. Their performance, however, drops under poor lighting and occlusion. While HOG-SVM provides higher accuracy, its computational load is greater. YOLO, though superior in accuracy, demands GPU acceleration.

VI. Applications

The proposed system finds utility in Smart Parking, Traffic Surveillance, Urban Planning, and Fleet Logistics where real-time, scalable vehicle monitoring is crucial.

VII. Conclusion

This paper presents a real-time vehicle detection system optimized for edge devices using Haar Cascades. Future work includes integrating convolutional models with Haar classifiers and adding tracking via SORT or Kalman Filters.

VIII. Author Contributions

Niketan Antil conceptualized the approach, implemented Haar Cascade training, collected data, conducted experiments on edge devices, and evaluated results against YOLO and HOG-SVM baselines.

FUTURE DIRECTIONS

Although the proposed vehicle detection system demonstrates strong real-time performance and computational efficiency, several enhancements can be explored in future work:

Integration with Deep Learning Models: Combining Haar Cascade detection with lightweight convolutional neural networks (CNNs) such as MobileNet or EfficientDet could improve detection robustness under varying environmental conditions like nighttime and rain.

Multi-Class Vehicle Detection: Extending the system to not only detect vehicles but also classify them into categories (e.g., cars, trucks, motorcycles) can provide richer traffic analysis capabilities.

Real-Time Tracking: Incorporating object tracking algorithms such as SORT (Simple Online and Realtime Tracking) or Kalman Filters would enable multi-frame tracking, allowing better handling of occlusions and dynamic vehicle movements.

Optimization for Edge Devices: Further optimization of the detection pipeline for deployment on low-power edge devices, such as Raspberry Pi or NVIDIA Jetson Nano, can broaden the system's applicability in smart city deployments.

Dataset Expansion and Augmentation: Enhancing the training dataset with more diverse and challenging environmental conditions, including nighttime, fog, and heavy traffic, would help improve generalization.

Fusion with Other Sensors: Future systems could combine vision-based detection with LiDAR, radar, or ultrasonic sensors for more robust and reliable vehicle detection, especially under adverse weather conditions.

References

1. [1] P. Viola and M. Jones, 'Rapid object detection using a boosted cascade of simple features', Proc. CVPR, 2001.
2. [2] N. Dalal and B. Triggs, 'Histograms of Oriented Gradients for Human Detection', Proc. CVPR, 2005.
3. [3] J. Redmon et al., 'You Only Look Once: Unified, Real-Time Object Detection', Proc. CVPR, 2016.
4. [4] A. G. Howard et al., 'MobileNets: Efficient Convolutional Neural Networks for Mobile Vision
6. [6] Y. Chen et al., 'Vehicle Detection in Highway Surveillance Using Lightweight CNNs', Sensors,

vol. 17, no. 12, 2017.
7. [7] S. Ahmed et al., 'Edge AI for Lightweight Monitoring in Urban Mobility', Proc. IEEE IoT, 2023.
8. [8] OpenCV Documentation, 'Haar Feature-based Cascade Classifiers', OpenCV.org, 2024.
9. [9] R. Kiran et al., 'Lightweight Haar Models for Embedded Traffic Monitoring', Int. J. CVPR, 2020.
10. [10] D. Sharma et al., 'Vehicle Counting Using Haar Cascades', Int. Conf. Smart Cities, 2022.

Applications', arXiv preprint arXiv:1704.04861, 2017.
5. [5] H. Zhang et al., 'Comparative Analysis of Detection Methods for Traffic Surveillance', IEEE Access, vol. 9, pp. 55678–55689, 2021.