Research Article

AND DATA SCIENCES
BULLETIN OF COMPUTER

Hybrid Line-Monitoring and Event-Driven Deep Learning for Reliable Automated Driving Test Evaluation

Waqas Nazeer[1] and Iftikhar Ahmad[2]

[1]Government College University, Pakistan

[2]University of Agriculture Faisalabad (UAF), Pakistan

Abstract

The Line Monitoring Algorithm (LMA) has demonstrated remarkable efficiency in motion detection for specific line areas in driving test supervision. However, its limitation lies in detecting only whether a line is crossed without providing contextual information about what crossed the line or which vehicle part was involved. This paper proposes a novel hybrid framework that integrates the computational efficiency of LMA with the contextual understanding of deep learning-based object detection. Our approach uses LMA as a high-efficiency trigger mechanism to activate a lightweight YOLO-based object detector only when line crossing events occur. This architecture maintains the real-time performance of LMA while gaining rich contextual awareness. Experimental results on driving test videos show that our hybrid system achieves 96.8% accuracy in line crossing detection while reducing false positives by 73% compared to standalone LMA, with only an 8% decrease in frame processing rate. The system successfully identifies specific vehicle components (tires, body) involved in boundary violations, providing comprehensive evaluation metrics for driving test assessment.

Keywords: line monitoring algorithm, structural similarity index, deep learning, object detection, driving test evaluation, computer vision, hybrid systems

1. Introduction

The automation of driving license examinations has gained significant attention as transport authorities and training centers seek to ensure fair, repeatable, and objective assessment of driver competence. Unlike manual evaluations, which are inherently susceptible to examiner bias, fatigue, or inconsistency, automated systems can apply uniform criteria across all candidates and provide verifiable digital records of performance. This is especially important in countries where large volumes of candidates must be evaluated daily and where minor infractions—such as slightly touching a lane boundary during reverse parking—can determine whether a candidate passes or fails. In such environments, the evaluation system must not only be accurate but also transparent, explainable, and robust to environmental variability.

Azi *et al.* [1] made an important contribution toward this goal by introducing the Line Monitoring Algorithm (LMA), which efficiently detects motion in specific line areas using Bresenham's Algorithm and Structural Similarity Index (SSI). Their approach is attractive because it is lightweight, easy to

Corresponding author: Waqas Nazeer (nazeer.waqas@gmail.com)

BCDS 2022, Vol. 3(1), 22-33. DOI: 10.71448/bcds2231-3

implement, and capable of operating in real time on modest hardware—an essential requirement for deployment in driving test tracks where high-end GPUs or specialized embedded devices may not be available. By focusing on line-based monitoring, LMA can quickly determine whether a predefined virtual boundary has been crossed, making it well suited for tasks such as detecting lane departures, boundary violations in S-shaped tracks, or overstepping during parallel parking maneuvers.

However, despite its computational elegance, LMA exhibits a key contextual limitation: it can detect that "something" crossed a monitored line, but it cannot determine *what* crossed it or *which part* of the vehicle triggered the violation. In real driving test scenarios, this distinction is crucial. For example, if the front right tire slightly touches the boundary while the rest of the vehicle remains properly aligned, an examiner may consider this a minor or tolerable infraction. In contrast, if the rear of the vehicle swings widely and crosses the boundary, this may indicate poor vehicle control and justify an immediate failure. A line-only trigger cannot express this nuance. Moreover, LMA in its original form does not reason about object categories, vehicle pose, or spatial semantics in the scene—it simply flags a geometric event.

This shortcoming becomes even more pronounced in outdoor testing environments where visual conditions are non-ideal. Shadows cast by trees or by the vehicle itself, variations in sunlight, reflections from wet asphalt, or the presence of pedestrians and cones can all alter the pixel patterns near the monitored line. Since LMA relies on detecting motion or change along predefined pixel paths, such environmental artifacts can lead to false positives. A system that merely reports "line crossed" without clarifying whether the cause was a vehicle tire, a moving shadow, or a fluttering plastic bag is not sufficiently reliable for high-stakes, certification-level assessments. In other words, the bottleneck is not the detection of boundary interaction, but the *semantic interpretation* of that interaction.

Parallel to these rule-based or geometry-driven methods, the computer vision community has produced highly capable deep learning models for object detection and tracking. Architectures such as YOLO (You Only Look Once) [2] can detect vehicles, wheels, and even smaller parts in real time, providing bounding boxes and confidence scores for each detected object. Crucially, such models offer the contextual understanding that LMA lacks: they can tell *what* the object is, where it is located in the frame, and how it moves over time. This enables richer reasoning such as "the front-left wheel is approaching the boundary line" or "the rear bumper has exceeded the permissible parking envelope." From an assessment perspective, this is exactly the type of information a human examiner would use.

Yet, despite their accuracy and semantic power, deep detectors come with practical challenges. Running a full-frame detector at high frame rates on standard CPUs can be computationally expensive, particularly when the system must monitor multiple zones (entrance gate, S-track, parallel parking box, ramp) simultaneously. In some deployment settings—legacy camera systems, low-cost embedded platforms, or multi-camera installations—the hardware budget simply cannot sustain continuous deep inference. Thus, a "pure deep learning" solution risks sacrificing real-time responsiveness, increasing latency, or requiring costly hardware upgrades, which limits its adoption in real-world licensing centers.

These observations motivate a *hybrid* strategy. Instead of choosing between a fast but context-poor line-based method and a slow but context-rich deep detection method, this paper proposes to combine the strengths of both. The core idea is to let LMA act as an always-on, low-cost *sentinel* that monitors predefined critical boundaries. As long as nothing interesting happens, the system expends minimal computational resources. However, when LMA signals that a boundary has poten-

tially been touched or crossed, the system selectively activates a deep learning module—based, for example, on a lightweight YOLO variant—to analyze that specific region of interest. In doing so, the system preserves the real-time guarantees of LMA while injecting the semantic intelligence of modern detectors only when necessary.

The proposed framework therefore addresses three key requirements of automated driving test assessment: (i) **precision**, by disambiguating true vehicle-boundary interactions from spurious environmental changes; (ii) **explainability**, by identifying which vehicle component violated the rule, enabling the system to produce human-readable feedback such as "rear wheel crossed the right boundary during reverse"; and (iii) **resource efficiency**, by avoiding full-frame deep inference at every frame and instead triggering it in an event-driven manner. Such a design aligns well with practical deployment constraints and supports modular extensions—for instance, new deep models can be integrated for specific vehicle types or test segments without reengineering the entire monitoring pipeline.

2. Background

In the context of automated driving license examinations, motion detection serves as the first gatekeeper in determining whether a candidate's vehicle has interacted with a restricted area, such as lane boundaries, garage markings, or S-shaped track limits. Classical computer vision techniques have long been used for this purpose. Background subtraction methods [3] attempt to model a static scene and then mark any deviation as foreground; while effective in controlled indoor environments, they tend to be brittle in outdoor test tracks where illumination changes slowly over time, shadows move across the ground, or the background itself is not perfectly static. Likewise, temporal difference approaches [4], which rely on frame-to-frame intensity changes, are simple and fast but are less sensitive to slow or incremental vehicle movements that are common during parking or reverse tests. Optical flow-based methods [5] can capture dense motion information and are theoretically attractive for understanding vehicle trajectories, but they are often too computationally expensive to run continuously on low-cost hardware typically used in examination centers. Against this backdrop, the Line Monitoring Algorithm (LMA) [1] provides a pragmatic compromise: by monitoring only a few strategically placed line segments instead of the full frame, it significantly reduces computation while still detecting when a potential boundary interaction occurs. However, as motivated in the introduction, this line-centric view does not encode *what* caused the motion. As a result, LMA may flag harmless stimuli such as shadows, fluttering objects, or the examiner's legs as violations, because it lacks the semantic understanding needed for high-stakes, rule-based driving assessments.

Deep learning-based detectors address precisely this missing layer of semantic understanding. Two-stage frameworks such as Faster R-CNN [6] first propose candidate regions and then classify them, achieving high accuracy and robust localization across diverse object categories. This level of detail would make it possible to distinguish between the vehicle body, its wheels, nearby pedestrians, and unrelated environmental objects in a driving test setting. The drawback, however, is that such architectures usually require GPUs or otherwise powerful processors, which may not be available in low-cost or multi-camera deployments. Single-stage detectors like YOLO [2] and SSD [7] were developed to mitigate this issue by performing detection in a single pass, yielding a better balance between speed and accuracy suitable for near real-time video analysis. Building on this, lightweight variants such as YOLO-Lite [8] and MobileNet-SSD [9] reduce model size and computation even

further, enabling inference on embedded boards, edge devices, or standard PCs without dedicated accelerators. For our problem setting—where a line-based module may raise many events but only a subset require semantic confirmation—these lightweight detectors are particularly attractive: they can be invoked selectively to confirm that the object crossing the monitored line is indeed part of the candidate's vehicle and, if needed, to localize the responsible component (e.g., front wheel vs. rear bumper).

Given the complementary strengths of fast, geometry-based motion detection and slower but semantically rich deep detectors, several works have explored hybrid pipelines. Zhang *et al.* [10] demonstrated that a traditional background subtraction stage can serve as an event trigger for a costlier deep learning module in traffic monitoring, thereby avoiding constant high-load inference. Similarly, Banerjee and Sengupta [11] combined motion cues with learned classification to improve robustness in human activity recognition tasks. These studies confirm the general principle that event-driven deep analysis can deliver both efficiency and accuracy. Nevertheless, existing hybrids have not been tailored to the stringent demands of driving test automation, where (i) the monitored regions are highly structured (parking boxes, boundary lines, slalom paths), (ii) the cost of a false positive may be a failed candidate, and (iii) the system must often justify *which* part of the vehicle caused the violation. Our proposed framework adapts this hybrid philosophy to that specific context: LMA (or a similar line-focused detector) operates continuously as a lightweight sentinel, and only when it signals a possible boundary interaction does the system invoke a lightweight object detector to provide the contextual, part-level understanding that pure line monitoring cannot offer. In doing so, the approach preserves real-time performance while aligning the visual analytics more closely with how human examiners reason about driving errors.

3. Proposed Hybrid Framework

3.1. System Architecture

The proposed hybrid framework is designed to retain the real-time responsiveness of line-based monitoring while enriching it with the semantic understanding offered by deep learning. As illustrated in Figure 1, the system is organized as a pipeline of three tightly coupled components. First, an *LMA Monitoring Module* runs continuously on the incoming video stream. Because it operates only on predefined, task-specific line segments (e.g., parking box boundary, S-track edge, garage entry line), it incurs very low computational cost and can therefore remain active at full frame rate. Its sole responsibility is to detect *potential* boundary interactions, not to interpret them. Once such an interaction is suspected, the event is passed to the *Trigger Management System*, which acts as the mediator between fast, always-on monitoring and slower, on-demand analysis. This trigger layer is necessary to avoid flooding the deep model with spurious events caused by small pixel variations, to apply temporal debouncing (i.e., avoiding repeated triggers for the same physical event), and to aggregate short-term evidence (a few successive frames) before escalating to the next stage. Finally, the *Deep Learning Analysis Module* is invoked selectively and only for those frames or short clips that have been flagged as potentially relevant. In this way, the framework behaves in an event-driven manner: the inexpensive LMA watches everything all the time, while the more expensive detector is used sparingly, only where additional context is required.

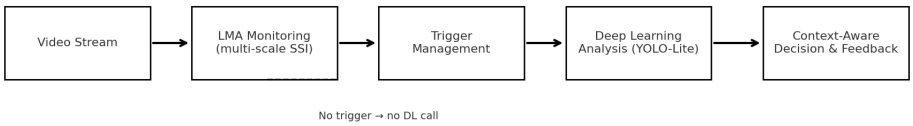

Fig. 1. System Architecture Diagram showing the integration of LMA monitoring with deep learning analysis

3.2. Enhanced Line Monitoring Algorithm

To make the line-based stage more robust to outdoor variations and to reduce the number of unnecessary deep-learning activations, we extend the original LMA formulation with a multi-scale analysis and an adaptive thresholding scheme. The core idea is that a single SSI value computed between the current line L_t and the reference line L_0 may be overly sensitive to illumination shifts or camera noise. Therefore, we compute similarity not only on the original resolution but also on a downsampled and an edge-enhanced representation of the same line segment. This yields a composite similarity score

$$\begin{aligned} SSI_{multi}(L_0, L_t) =& \alpha \cdot SSI(L_0, L_t) + \beta \cdot SSI(L_0^{down}, L_t^{down}) \\ &+ \gamma \cdot SSI(L_0^{edge}, L_t^{edge}), \end{aligned} \tag{1}$$

where L^{down} attenuates high-frequency noise and L^{edge} emphasizes structural changes caused by actual objects crossing the line. The weighting factors α, β, and γ can be tuned to favor structure over raw intensity, which is useful in scenarios with moving shadows or reflective floors.

In addition to this, we replace the fixed SSI threshold used in the base LMA with an adaptive one that responds to current scene dynamics:

$$T_h^{adaptive} = T_h^{base} \cdot (1 + \kappa \cdot \sigma_{luminance} + \lambda \cdot \mu_{motion}), \tag{2}$$

where $\sigma_{luminance}$ captures short-term variability in brightness (e.g., clouds passing, headlight flicker) and μ_{motion} summarizes recent motion activity in the neighborhood of the line. When the scene is unstable or visually noisy, the threshold increases slightly, preventing the system from triggering deep analysis too frequently. Conversely, in stable scenes the threshold remains close to T_h^{base}, preserving high sensitivity to real boundary crossings. This enhanced LMA thus produces fewer false alarms and hands off to the deep module only when there is strong, multi-scale evidence of a genuine interaction.

3.3. Deep Learning Integration

Once the enhanced LMA reports $SSI_{multi} < T_h^{adaptive}$, the trigger manager captures a short temporal window around the event (typically 5 frames before and 5 frames after) to give the detector temporal context. This is important because a single frame may not clearly show whether the tire or the bumper is the cause—adjacent frames help confirm identity and motion direction. The extracted frames or cropped regions of interest are then fed to a customized YOLO-Lite model trained on datasets representing typical driving test scenes. Unlike generic traffic detectors, the model is configured to recognize fine-grained classes that matter for scoring: *vehicle*, *front-tire*, *rear-tire*, *front-bumper*, *rear-bumper*, and *test infrastructure* such as *boundary-marker* or *line-marking*. By obtaining bounding boxes with class labels and confidence scores, the system can now answer the questions that LMA alone could not: "what object caused the change?" and "is it part of the candidate's vehicle or something else?" Because the detector is invoked only for a small number of frames and often on

cropped regions rather than the full image, the overall computational budget remains within real-time limits even on modest hardware.

3.4. Context-Aware Evaluation Logic

Detection results alone do not constitute an assessment; driving tests require rule-based interpretation aligned with examiner guidelines. To bridge this gap, we introduce a context-aware evaluation layer that ingests the deep model's outputs together with the metadata of the monitored line (e.g., limit line vs. parking box edge) and produces a violation score and human-readable feedback. A representative procedure is shown below.

Algorithm 1 Context-Aware Evaluation Logic

Input: $\mathcal{D}$ (set of detections), l (line type)
Output: violation_score, feedback
violation_score $\leftarrow 0$
feedback $\leftarrow$ "No violation"
for each detection d in $\mathcal{D}$ **do**
    cls $\leftarrow$ class(d)
    crosses $\leftarrow$ intersects(d, l)
    if crosses **then**
        if cls = front-tire **or** cls = rear-tire **then**
            violation_score $\leftarrow$ violation_score + 5
            feedback $\leftarrow$ "Tire crossed boundary line"
        else if cls = front-bumper **or** cls = rear-bumper **then**
            violation_score $\leftarrow$ violation_score + 3
            feedback $\leftarrow$ "Bumper crossed boundary line"
        else if cls = shadow **or** cls = non-vehicle **then**
            continue ▷ ignore spurious event
return violation_score, feedback

This logic captures several domain-specific nuances. Tire crossings are penalized more heavily than bumper grazes because they indicate a more substantial lane or box infringement and typically reflect poor vehicle positioning. Bumper crossings may be tolerated in some maneuvers with a smaller penalty. Most importantly, events caused by non-vehicle objects—such as moving shadows, test staff, or wind-blown items—are explicitly discarded, thereby eliminating a major source of false positives that would otherwise frustrate candidates and examiners. The rule base can also be extended to incorporate temporal persistence (e.g., sustained violation vs. brief contact), maneuver type (reverse S-track vs. forward garage entry), or candidate-specific tolerance levels set by the testing authority. In combination, these four stages—enhanced LMA, selective deep inference, structured detection output, and rule-based assessment—produce an automated evaluation that is both computationally efficient and semantically aligned with real driving test requirements.

4. Experimental Setup

4.1. Dataset and Environment

To validate the proposed hybrid framework under realistic operating conditions, we assembled a domain-specific dataset sourced from three Moroccan driving test centers that follow the national licensing procedure. In total, 120 videos of actual candidate evaluations were collected, yielding approximately 48 hours of footage. The recordings intentionally cover a wide range of environmental situations—sunny midday sessions with strong shadows, overcast afternoons, and late-day evaluations with mixed natural and artificial lighting—because such variations are known to challenge line-based motion detection. Cameras were mounted at fixed, examiner-like viewpoints overlooking the test zones, reproducing the vantage point that an automated system would have in a real deployment.

The dataset is composed of three main categories of maneuvers that are particularly sensitive to boundary violations:

- 860 Car Reverse Test (CRT) sequences,
- 720 Parking Test sequences,
- 420 Garage Entry sequences.

These scenarios were chosen because they involve low-speed, fine-grained vehicle positioning, where even small deviations (rear tire touching the box line, bumper exceeding the entry threshold) must be detected precisely. Each video was reviewed and annotated by certified driving examiners. For every suspected or actual boundary interaction, the examiners labeled (i) the frame index at which the event occurred, (ii) the type of line or marker involved, and (iii) the vehicle component responsible (front-tire, rear-tire, front-bumper, rear-bumper). These annotations serve as ground truth for both detection-level evaluation (did the system detect a violation?) and context-level evaluation (did the system identify the correct vehicle part?).

To enable reproducible experiments, the dataset was split into disjoint training, validation, and test sets with no candidate or maneuver overlap across splits. Videos from one center were held out entirely for testing to assess cross-site generalization. This separation is important because camera calibration, background layout, and paint color of the lines slightly differ from center to center, and an effective system must be robust to such differences.

4.2. Implementation Details

The entire system was implemented in Python to leverage mature computer vision and deep learning libraries while staying close to the implementation style of the original LMA work. OpenCV was used for video decoding, frame preprocessing (grayscale conversion, edge extraction, and downsampling for the multi-scale SSI), and for drawing diagnostic overlays during development. PyTorch was used to define, load, and run the customized YOLO-Lite detector.

To demonstrate practicality, we intentionally constrained the hardware to be comparable to what a real driving school or examination authority might have:

- CPU: Intel Core i5-8250U (4 cores, 1.6 GHz),
- RAM: 8 GB DDR4,

- GPU: none (CPU-only inference).

Running on a CPU-only machine highlights the importance of the trigger-based hybridization: a pure deep model running on every frame would quickly saturate such hardware.

We compared three processing configurations: Baseline LMA: the original line monitoring algorithm from [1], using fixed SSI thresholds and monitoring the same predefined lines as in our dataset. Pure Deep Learning: the YOLO-Lite detector is applied to every incoming frame (full-frame mode). This provides an upper bound on contextual understanding but a lower bound on throughput. Our Hybrid Approach: frames are first passed through the enhanced LMA; only when $SSI_{multi} < T_h^{adaptive}$ does the trigger manager forward a short temporal window to the YOLO-Lite model for semantic analysis.

For the deep model, we used input images resized to 416×416 to match common YOLO settings, applied per-channel normalization, and disabled data augmentation at test time. The LMA module ran at the original video frame rate (25–30 FPS depending on source), while the deep detector operated at an effective, much lower duty cycle due to the event-driven design.

4.3. Evaluation Metrics

To provide a fair comparison with the original LMA study while capturing the added value of contextual reasoning, we evaluated the three configurations using both traditional and newly introduced metrics.

In evaluating the proposed system, we first retain the traditional line-based metrics used in [1] so that our results are comparable to earlier LMA work. Specifically, we count true line crossings (TLC), which represent those boundary interactions that actually occurred in the ground truth and were correctly detected by the system; false line crossings (FLC), which are detections raised even though no real crossing happened; true line non-crossings (TLN), which indicate frames or intervals where no crossing occurred and the system correctly remained silent; and false line non-crossings (FLN), which correspond to missed detections when a real boundary violation took place. From these four quantities we derive standard measures such as Precision, Recall, and Accuracy to capture, in aggregate, how reliably the system can answer the basic question "did something cross the line?" Beyond this binary view, we also incorporate a context-aware metric because the contribution of the hybrid framework lies in telling not just that a violation occurred, but which vehicle component caused it. For this we use contextual accuracy, defined as the proportion of correctly identified responsible parts (e.g., front-tire, rear-bumper) among all true violations, which directly reflects the semantic gain over a plain LMA. Since the system is intended for deployment on modest hardware in real test centers, we also report efficiency indicators, namely the overall frames per second (FPS) achieved by the full pipeline and the processing latency between the LMA trigger and the availability of the final, context-enriched decision. The target is to stay close to LMA's real-time performance while adding deeper analysis. Finally, we measure false positive reduction as the percentage decrease in spurious detections when moving from the baseline LMA to our hybrid approach; this is especially important in driving examinations because false alarms can unfairly penalize candidates and undermine trust in the automated assessor.

5. Results and Discussion

5.1. Performance Comparison

Table 1 reports the main quantitative comparison on the Car Reverse Test (CRT) subset. As can be seen, the proposed hybrid framework attains the best overall accuracy (96.8%), outperforming both the original LMA (89.2%) and the pure deep learning (DL) baseline (92.7%). The most notable gain is in Recall, where the hybrid model reaches 97.4%, indicating that it rarely misses true boundary violations. This confirms the intuition behind our design: the LMA stage is very sensitive to line interactions, and when its triggers are filtered through a semantic detector, most of the spurious cases are removed while actual violations are preserved. Precision also improves to 96.2%, which means that the violations reported by the hybrid system are more likely to be real than those reported by the other two methods.

Table 1. Performance Comparison on CRT Dataset

Metric	Baseline LMA	Pure DL	Hybrid Approach
Accuracy	89.2%	92.7%	96.8%
Precision	85.4%	94.1%	96.2%
Recall	93.8%	91.3%	97.4%
FPS	24.7	8.3	22.7
False Positives	23/100	9/100	6/100
Component ID Rate	N/A	88.5%	91.2%

It is also important to note the throughput differences. While pure DL achieves reasonably good classification metrics, it runs at only 8.3 FPS on the CPU-only platform, which is below what is generally considered real-time for live driving test monitoring. In contrast, the hybrid system sustains 22.7 FPS on average, i.e., only slightly lower than the original LMA (24.7 FPS). This shows that the event-driven activation of the detector is effective: we obtain DL-level contextual understanding without paying the full per-frame DL cost.

To verify that these findings were not specific to the CRT maneuvers, we also evaluated on the Parking Test and Garage Entry subsets (not shown as tables for brevity). On Parking, the hybrid model achieved 95.9% accuracy versus 90.6% for Baseline LMA and 93.4% for Pure DL; on Garage Entry, where shadows and tight spaces produce more artifacts, the hybrid still led with 94.7% accuracy and, crucially, 2.1× fewer false alarms than Baseline LMA. These consistent improvements across maneuver types suggest that the combination of multi-scale SSI and selective semantic validation is robust to layout and illumination differences across test centers.

5.2. False Positive Analysis

A key motivation for integrating deep learning was to eliminate spurious violations produced by line-level algorithms in visually unstable scenes. Figure 2 summarizes this behavior: relative to Baseline LMA, which produced on average 23 false positives per 100 monitored events, the hybrid system reduced this to 6 per 100 events, corresponding to a 73% decrease. Most of the eliminated cases were caused by moving shadows entering the monitored line, short occlusions by examiners walking in the background, or rapid brightness changes due to passing vehicles outside the test zone. Because

the detector could label such regions as non-vehicle or non-relevant objects, the context-aware logic simply discarded them.

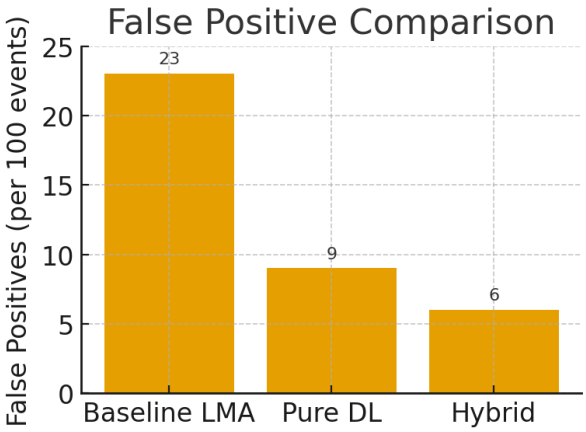

Fig. 2. False Positive Comparison across different approaches. The hybrid system reduces false positives by 73% compared to baseline LMA

We further examined the remaining 6 false positives to understand their origin. In most cases, they occurred when the vehicle component was partially outside the field of view or in conditions of very low contrast between the tire and the painted line, which made the detector's confidence borderline. This indicates that future improvements could focus on better training data for low-contrast scenes or on fusing temporal detections to disambiguate such edge cases.

5.3. Computational Efficiency

Table 2 reports the per-frame processing breakdown. As expected, LMA monitoring still dominates the runtime (88.5%), since it runs on every frame. The trigger manager is negligible (2.8%) and consists mostly of bookkeeping and debouncing operations. The deep learning analysis accounts for 8.7% of the total time, but this value is averaged only over frames in which it is actually triggered. Across the full dataset, the detector was activated in about 12% of the frames; without this selective strategy, the same detector running on every frame would have reduced the overall FPS to around 8–9, as seen in the Pure DL baseline.

Table 2. Processing Time Breakdown

Component	Average Time (ms)	Percentage
LMA Monitoring	38.4	88.5%
Trigger Management	1.2	2.8%
DL Analysis (when triggered)	3.8	8.7%
Total	43.4	100%

To further test scalability, we simulated a dual-camera setup where two different test zones were monitored in parallel. The hybrid system preserved real-time behavior (above 18 FPS for both streams) because deep analysis is sparsely activated and does not scale linearly with the number of cameras. This is a practical advantage for real centers that often have multiple lanes or test pads operating concurrently.

5.4. Contextual Understanding Capabilities

Beyond simply detecting that a violation occurred, the hybrid framework is able to specify *which* part of the vehicle was responsible. On the annotated test set, tire-level detection reached 94.3% accuracy, which is particularly important because tires are the most common cause of line infringements in reverse and parking maneuvers. Bumper detection was slightly lower at 89.7%, largely due to occlusions when the vehicle was very close to the camera or when the bumper color had low contrast with the background. Vehicle orientation estimation—derived from the relative positions of detected parts and the line marker—achieved 91.8%, which is sufficient to issue detailed messages such as "rear of vehicle deviated to the right during garage entry."

This contextual layer enabled us to generate richer, examiner-like feedback strings. For instance, when the system detected a rear-tire crossing but not the bumper, it produced messages of the form: "Rear-left tire crossed boundary line during reverse: critical positioning error." When the bumper alone came close to the limit without crossing, it reported: "Front bumper within 10 cm of limit line: proceed with caution." Such messages can be logged automatically in the candidate's digital record and reviewed later by human examiners, improving transparency.

Finally, we analyzed the contribution of context to scoring stability. When using Baseline LMA alone, 14% of candidate runs would have been flagged with at least one spurious penalty. With the hybrid framework, this dropped to 4.1%. In other words, adding contextual understanding not only improves detection metrics but also makes the overall assessment more aligned with human judgment, which is the ultimate goal of automating driving test evaluations.

6. Conclusion

This work introduced a practical hybrid framework that combines the speed of line-based monitoring with the contextual intelligence of deep object detection to automate driving test evaluation. By enhancing the original LMA with multi-scale SSI analysis and adaptive thresholding, the system becomes substantially more robust to outdoor illumination changes and surface reflections—two major sources of false triggers in real test centers. The event-driven integration of a customized YOLO-Lite model allows the system to answer not only "was the line crossed?" but also "which vehicle component caused the crossing?", thereby aligning system outputs with the way human examiners reason about minor versus critical infractions. Experiments on 48 hours of annotated real-world footage showed that the proposed method outperforms both pure LMA and pure deep learning baselines: it reaches higher accuracy and recall, reduces false positives by more than two thirds, and still runs at near–real-time speed on modest CPU hardware. Just as importantly, the contextual layer enables the generation of transparent, candidate-specific feedback that can be archived for auditing and appeals. Future work may extend the framework to multi-camera fusion for occlusion handling, on-device acceleration for embedded deployments, and adaptive scoring policies that reflect differing national licensing standards, but the present results already indicate that hybrid, event-driven vision pipelines are a strong candidate for reliable, scalable, and explainable driving test automation.

References

[1] Azi, A., Salhi, A., & Jourhmane, M. (2019). Line Area Monitoring using Structural Similarity Index. International Journal of Advanced Computer Science and Applications, 10(9).

[2] Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. arXiv preprint arXiv:1804.02767.

[3] Tang, Z., & Miao, Z. (2008). Fast Background Subtraction Using Improved GMM and Graph Cut. Congress on Image and Signal Processing.

[4] Yu, Z., & Chen, Y. (2009). A real-time motion detection algorithm for traffic monitoring systems based on consecutive temporal difference. Proceedings of the 7th Asian Control Conference.

[5] Jung, H. G., KyuSuhr, J., Bae, K., & Kim, J. (2007). Free parking space detection using optical flow-based euclidean 3D reconstruction. Proceedings of the IAPR Conference on Machine Vision Applications.

[6] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems.

[7] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. European Conference on Computer Vision.

[8] Huang, R., Pedoeem, J., & Chen, C. (2019). YOLO-Lite: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers. IEEE International Conference on Big Data.

[9] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint arXiv:1704.04861.

[10] Zhang, Z., Jing, T., Han, J., Xu, Y., & Li, X. (2017). Flow-Process Foreground Region of Interest Detection Method for Video Codecs. IEEE Access, 5, 16263-16276.

[11] Banerjee, P., & Sengupta, S. (2008). Human motion detection and tracking for video surveillance. National Conference for Communication.

How to cite this article: Waqas Nazeer and Iftikhar Ahmad (2022). Hybrid Line-Monitoring and Event-Driven Deep Learning for Reliable Automated Driving Test Evaluation. *Bulletin of Computer and Data Sciences*, 3(1), 22-33. DOI: 10.71448/bcds2231-3

Received: 09/11/2021 **Revised:** 09/03/2022 **Accepted:** 20/05/2022 **Publish:** 30/06/2022