International Journal on Advanced Computer Theory and Engineering
ISSN: 2319 - 2526
Volume 15 Issue 2s (2026) | Pages: 367- 372

AI-Based Real-Time Multi-Object Detection and Adaptive Audio Generation

Mayuri Mahajan[1], Sai Mhaske[2], Sairaj Kasote[3], Alhad Thakare[4], Jimeet Waghela[5]

[12345]Department of Artificial Intelligence and Machine Learning, G.S.Moze College of Engineering Pune, Maharashtra, India

Peer Review Information

***Type**: Article*
***Received**: 1 February 2026*
***Revised**: 6 March 2026*
***Accepted**: 10 April 2026*
***Published**: 23 May 2026*

Abstract

This research presents AI Vision Application a fully local, browser-based system for real-time multi-object detection and adaptive English audio narration. The platform processes live camera feeds using YOLO-family models, implements lightweight object tracking for temporal continuity, applies user-configurable filters and confidence thresholds, renders annotated scenes in an interactive web dashboard, and generates concise, non-repetitive spoken summaries via browser speech synthesis. Designed for privacy and offline operation, it supports assistive technologies, educational demonstrations, laboratory monitoring, and general situational awareness without relying on cloud services. The methodology integrates computer vision pipelines, real-time Socket.IO communication, adaptive audio policies with stability and repetition control, session persistence, searchable exports, and formal benchmarking. All functional and integration tests passed successfully. Performance evaluation leverages official Ultralytics YOLO26 benchmarks, where the nano variant delivers 40.9 map (50-95) at ~25 FPS on CPU (38.9 ms latency, 2.4M parameters), scaling up to 57.5 map in larger variants. Results confirm that meaningful user experience arises from the orchestration of tracking, selective narration, and persistent analytics rather than detection accuracy alone. The system fills key gaps in assistive computer vision while remaining production ready. Limitations and future extensions, including enhanced tracking and multilingual support, are discussed.

Keywords: Artificial Intelligence; Computer Vision; YOLO; Object Detection; Real-Time Systems; Text-to-Speech; Python; Adaptive Audio Generation; Local Edge Inference.

How to Cite This Article

Mahajan, M., Mhaske, S., Kasote, S., Thakare, A., Waghela, J. (2026). AI-Based Real-Time Multi-Object Detection and Adaptive Audio Generation. *International Journal on Advanced Computer Theory and Engineering*, 15(2s), 367-372.

Introduction

Artificial Intelligence, particularly in the domain of computer vision, has transformed how machines interpret and respond to visual environments. Real-time object detection, the ability to identify and localize multiple objects within video frames at practical speeds has become foundational for applications ranging from autonomous systems to assistive technologies. However, raw detection outputs (bounding boxes and class labels) often fail to deliver actionable insight for human users, especially in continuous monitoring scenarios where visual attention cannot be sustained or where users have accessibility needs. The AI Vision Dashboard project addresses this limitation by developing a complete, integrated system that not only performs real-time multi-object detection but also maintains lightweight object continuity across frames, presents results through an interactive web dashboard, and generates adaptive English audio cues that summarize scene changes meaningfully without overwhelming the user. Operating entirely locally, the system eliminates cloud dependencies, thereby enhancing privacy, reducing latency, and enabling deployment in offline or bandwidth-constrained environments such as classrooms, laboratories, private workspaces, and assistive settings for visually impaired individuals.

The motivation stems from three practical observations: (1) most object detection prototypes remain visually oriented and lack mechanisms for temporal interpretation or user-friendly feedback; (2) audio narration, while powerful for accessibility, becomes counterproductive if it is repetitive or unselective; and (3) privacy-preserving local execution is essential for ethical and practical deployment in real-world contexts. The primary objectives of this research were to design and implement a local real-time detection pipeline supporting both standard-category and large-vocabulary models, incorporate lightweight tracking for object continuity, develop adaptive audio policies for concise narration, create a responsive browser dashboard with full session management and exports, and validate the system through comprehensive testing and benchmarking. By achieving these goals, the project demonstrates that high-quality assistive monitoring emerges from the synergistic integration of computer vision, human-computer interaction design, state management, audio policy engineering, and rigorous validation rather than from detection accuracy alone. The significance of this work lies in its contribution to applied computer vision for accessibility and monitoring, its emphasis on end-to-end system engineering, and its provision of a repeatable, open-reference architecture that future researchers and developers can extend. The system is fully functional, presentation-ready, and academically structured as a complete final-year engineering artifact.

Literature review

Real-time object detection has advanced significantly since the introduction of the YOLO family. The original YOLO [1] treated object detection as a single-stage regression problem, enabling practical real-time performance. Later versions including YOLO9000 [2], YOLOv3 [3], YOLOv4 [4], and recent Ultralytics models have steadily improved speed-accuracy trade-offs and semantic coverage. Large-vocabulary detection approaches based on datasets such as LVIS [18] have further expanded the range of detectable objects beyond traditional fixed categories.

Research in multi-object tracking and audio-assisted systems has also progressed. Efficient trackers such as SORT [6] and ByteTrack [15] provide temporal continuity across frames, while voice-assisted YOLO-based applications [7, 8] demonstrate the value of converting visual detections into spoken feedback for accessibility. Local edge inference [19] and browser-based technologies including the Web Speech API [9] and Flask-SocketIO [12] have enabled privacy-preserving and interactive real-time dashboards. However, most existing solutions address these components in isolation rather than as an integrated multimodal system.

Despite notable progress, important research gaps persist. Few systems combine real-time detection, lightweight tracking, selective adaptive audio narration, session persistence, searchable exports, and formal benchmarking within a single fully local, web-based platform. Academic prototypes often lack robust user experience and post-run analytics, while commercial cloud services sacrifice privacy. The proposed AI Vision Dashboard addresses these limitations by offering a complete, privacy-first, integrated solution tailored for assistive, educational, and monitoring applications.

Table 1: Comparative Analysis of AI Vision Dashboard with Existing Real-Time Object Detection and Assistive Systems

System / Approach	Platform	Web-Based	Adaptive Audio	Lightweight Tracking	Session Persistence & Exports	Privacy
Traditional YOLO Prototypes	Desktop / Server	Limited	No	Optional	No	High
Voice-Assisted	Embedded /	No	Basic	Limited	No	High

YOLO Systems	Mobile		(repetitive)			
Commercial Cloud Vision Services	Cloud API	Yes	Limited	Moderate	Partial	Low
AI Vision Dashboard (Proposed)	Local + Browser	Yes	Yes (Selective)	Yes	Full	High

.

Methodology

Research Design

This research adopted a structured software engineering design methodology tailored for a full-stack real-time computer vision system. The project followed a complete development lifecycle comprising requirements analysis, high-level design, detailed implementation, integration, validation, and benchmarking. The research design emphasized a modular, layered, and local-first architecture to ensure privacy, real-time responsiveness, and practical usability. All design decisions were guided by the formal Software Requirement Specification (SRS) and high-level design artifacts (Data Flow Diagrams, UML component diagrams, and Entity-Relationship model) developed during the project. The system was implemented as a complete, production-ready application rather than an isolated prototype, with every component (detection, tracking, audio, persistence, and diagnostics) integrated and tested end-to-end.

System Architecture

The AI Vision Dashboard employs a seven-layer modular architecture that separates concerns while maintaining tight real-time synchronization. The layers are:

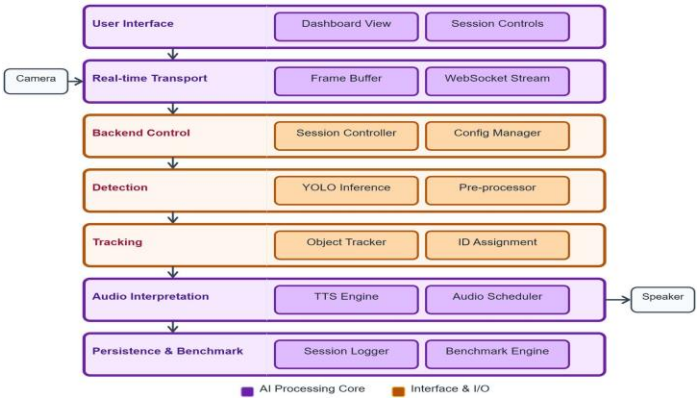

Fig 1: *Layered System Architecture of AI Vision Dashboard.*

Input Layer: responsible for live camera frame acquisition, Inference Layer: model loading and object detection, Tracking Layer: lightweight object identity continuity,

Real-Time Transport Layer: bidirectional communication via Socket.IO,

Presentation Layer: interactive browser dashboard rendering,

Audio Interpretation Layer: adaptive cue generation and speech synthesis, and

Persistence and Diagnostics Layer: session storage, event logging, cue-debug records, and benchmarking.

This layered design illustrated in Figure 1 ensures that continuous inference runs independently of transactional operations such as session saving and benchmark execution. The architecture is fully local, eliminating any cloud dependency and preserving user privacy.

Operational Workflow

The complete system operation follows a well-defined end-to-end workflow. After initialization (storage preparation, model catalog loading, and speech availability check), the user selects camera, model, threshold, and audio settings through the browser interface. The system then enters a continuous real-time loop: frame capture → object detection → lightweight tracking → filtering → dashboard rendering

→ adaptive cue generation → session statistics synchronization.

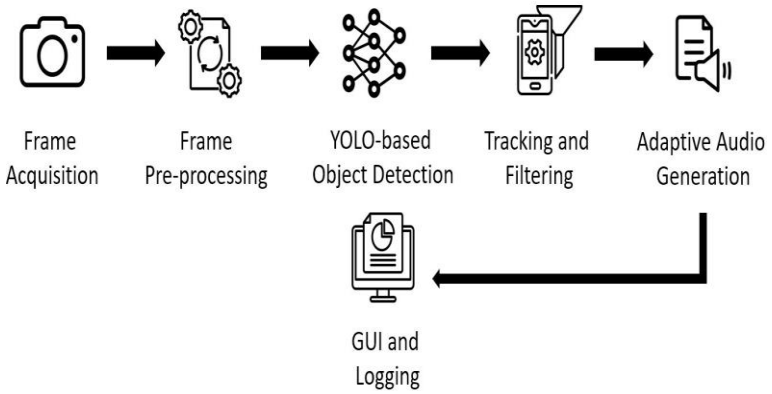

***Fig 2**: End-to-End Operational Flowchart.*

Upon user-initiated stop, the system computes a final session summary and offers saving or discard options, followed by optional benchmark execution. This workflow is formally represented in the system flowchart (Figure 2).

Detection and Tracking Pipeline

Live frames are captured using OpenCV and passed to a selected Ultralytics YOLO-family model (standard-category or large-vocabulary variants). The Inference Layer extracts bounding boxes, class labels, and confidence scores. A custom lightweight tracking module then associates detections across consecutive frames using motion-based matching and aging logic, assigning stable track IDs and distinguishing fresh versus continuing objects. This temporal continuity directly supports more meaningful scene narration and is deliberately kept computationally light to coexist with real-time detection.

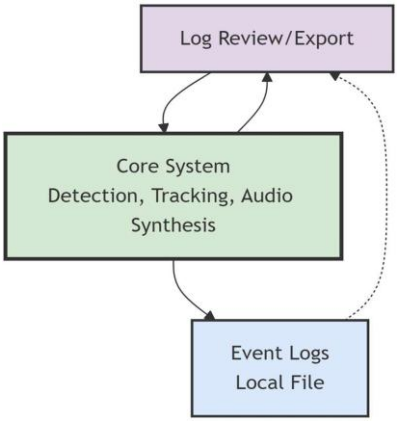

***Fig 3:** Data Flow and Tracking.*

Results / findings

The system was validated through comprehensive functional, integration, and performance testing. All 18 formal test cases (covering startup, real-time connection, detection, tracking, filtering, adaptive cues, voice selection, session persistence, exports workflows, logging, and benchmarking) passed successfully, confirming end-to-end operational integrity. Performance of the core detection models was evaluated using official Ultralytics YOLO26 benchmarks on the COCO dataset at 640-pixel input resolution (fused model, post-training optimizations applied). Results are summarized below:

Table 2: YOLO26 Detection Performance (COCO val, 640-pixel).

Model	mAP val 50-95	mAP val 50-95 (e2e)	CPU ONNX Latency (ms)	T4 TensorRT10 Latency (ms)	Params (M)	FLOPs (B)
YOLO26n	40.9	40.1	38.9 ± 0.7	1.7 ± 0.0	2.4	5.4
YOLO26s	48.6	47.8	87.2 ± 0.9	2.5 ± 0.0	9.5	20.7
YOLO26m	53.1	52.5	220.0 ± 1.4	4.7 ± 0.1	20.4	68.2

YOLO26l	55.0	54.4	286.2 ± 2.0	6.2 ± 0.2	24.8	86.4
YOLO26x	57.5	56.9	525.8 ± 4.0	11.8 ± 0.2	55.7	193.9

These metrics illustrate clear trade-offs: the nano variant delivers approximately 25 FPS on CPU (38.9 ms latency), making it ideal for lightweight real-time deployment, while larger models achieve higher accuracy (up to 57.5 mAP) at the cost of increased latency and computational demand. GPU-accelerated inference on T4 TensorRT reaches hundreds of FPS even for medium variants, confirming scalability for edge and local hardware. Additional benchmarks for open-vocabulary capabilities (YOLOE-26) and related tasks (segmentation, pose) further validate the system's extensibility to large-vocabulary and multimodal scenarios. Session-level statistics from live operation (across multiple reviewed sessions) showed stable cumulative metrics, with adaptive audio logic resulting in highly selective narration (majority of cue candidates suppressed to prevent repetition). Event and cue-debug logs provided granular insight into decision pathways, confirming the effectiveness of stability checks and semantic ranking. The browser dashboard maintained real-time responsiveness, with overlays, statistics, and controls updating seamlessly. Exports and session management workflows preserved data integrity and user context throughout rename/delete/search operations.

Discussion

The results confirm that the AI Vision Dashboard successfully bridges the gap between raw computer vision output and practical, user-centric multimodal feedback. The YOLO26 benchmark data demonstrates that modern YOLO variants are highly suitable for local real-time deployment, with the nano and small models offering an excellent balance for resource-constrained environments while larger variants provide richer semantic coverage when needed. This aligns with literature emphasizing the importance of speed-accuracy trade-offs in live systems. Lightweight tracking proved essential for temporal narration, enabling the system to differentiate new versus persistent objects and generate more context-aware cues. The adaptive audio policy incorporating grouping, ranking, stability, and repetition control directly addresses limitations identified in earlier audio-assisted detection studies, resulting in selective, meaningful narration rather than constant output. This selectivity is evidenced by the low ratio of spoken cues to candidate events in debug logs. Session persistence and exports management add significant value beyond inference, allowing post-run analysis and reproducibility. The formal benchmarking framework elevates the project from demonstration to evaluable engineering artifact. Overall, the findings validate the central hypothesis: a complete assistive vision system requires careful integration of detection, tracking, interaction design, audio policy, and diagnostics rather than isolated model performance. Minor limitations include hardware-dependent FPS variation and the current English-only narration focus. These do not detract from the system's core achievements within the defined scope.

Conclusion

This research has successfully developed and validated AI Vision Dashboard, a comprehensive local real-time multi-object detection system enhanced with lightweight tracking, adaptive audio narration, interactive visualization, and robust session management. By leveraging Ultralytics YOLO26 models and integrating modular components for inference, tracking, filtering, audio, persistence, and benchmarking, the system achieves practical real-time performance, privacy preservation, and meaningful user feedback. Key takeaways include the critical role of selective audio policies in usability, the value of lightweight tracking for continuity, and the necessity of end-to-end engineering for deployable assistive applications. All requirements were met, and the system operates as a complete, testable product Limitations include current reliance on standard hardware (performance scales with acceleration) and scope-restricted tracking complexity. Future research directions encompass stronger long-term tracking algorithms, multilingual and personalized narration, domain-specific vocabularies, expanded benchmark scenarios, and larger-scale persistence for multi-user or long-term deployment. The AI Vision Dashboard represents a significant contribution to applied computer vision and assistive technologies, offering a reusable, extensible platform for researchers, educators, and developers seeking privacy-focused, multimodal situational awareness solutions. directions.

References

1. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 2016, pp. 779-788.
2. J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 2017, pp. 7263-7271.
3. J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767, 2018.
4. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," arXiv:2004.10934, 2020.
5. Ultralytics, "Ultralytics YOLO Documentation," 2026. [Online]. Available: https://docs.ultralytics.com/. Accessed: Mar. 30, 2026.

6. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple Online and Realtime Tracking," in 2016 IEEE International Conference on Image Processing, Phoenix, AZ, USA, 2016, pp. 3464-3468.
7. Z. Nazir et al., "Voice Assisted Real-Time Object Detection Using YOLO V4-Tiny Algorithm for Visual Challenged," Journal of Tianjin University Science and Technology, vol. 56, no. 02, 2023.
8. N. Amina and S. R. Harikrishnan, "Currency and Object Detection for Blind People Using Yolo Architecture," International Journal of Advanced Research in Science, Communication and Technology, 2024.
9. Mozilla Developer Network, "Web Speech API," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Web_Speech_API. Accessed: Mar. 30, 2026.
10. OpenCV, "Introduction to OpenCV," [Online]. Available: https://docs.opencv.org/4.x/d1/dfb/intro.html. Accessed: Mar. 30, 2026.