Dashcam Guardian: Real-Time Traffic Accident Risk Detection System

1st Gowtham S
Department of Advanced computing and Analytics Vels Institute of Science, Technology & Advanced Studies Chennai, India gowthams2k20@gmail.com

2nd Mr. Akhil S
Department of Advanced computing and Analytics Vels Institute of Science, Technology & Advanced Studies Chennai, India akhilsree77@gmail.com

Abstract-Road traffic accidents are a leading cause of fatalities worldwide. This paper presents Dashcam Guardian, a real-time traffic accident risk detection system that employs YOLOv8 deep learning for multi-class object detection, centroid-based object tracking, bounding box distance estimation, frame-to-frame speed estimation, lane boundary detection using Hough transform, and a composite accident risk scoring algorithm. The system detects vehicles, auto- rickshaws and pedestrians in real time, assigns unique tracking IDs, estimates distances and speeds, computes a dynamic risk score (0–100%), and provides layered audio alerts (beep sounds for warnings and voice alerts for danger). Optimised for Apple M2 via MPS acceleration and a threaded camera reader, the system achieves 25–40 FPS on a standard laptop camera. The system also incorporates Indian-road- specific auto-rickshaw reclassification to correct common YOLOv8 misclassification errors. Effective real-time detection and collision risk assessment under various road conditions are demonstrated by experimental results.

Keywords: Yolov8; object detection; ADAS; collision risk; lane detection; real-time processing; deep learning; driver assistance; dashcam; accident detection.

I. INTRODUCTION

The World Health Organization estimates that road traffic accidents result in over 1.35 million deaths annually worldwide. A large percentage of these collisions are caused by drivers who are not paying attention, fail to notice cars in the vicinity, make poor distance judgments, and react slowly. Although Advanced Driver Assistance Systems (ADAS) are commonly found in high-end cars, most car owners around the world cannot afford the costly hardware that these systems require, such as radar, LiDAR, and multi-camera arrays. Many essential ADAS features can now be implemented with just a standard camera thanks to the quick development of deep learning-based computer vision.

Models such as YOLOv8 can detect and classify road objects in real time with high accuracy on consumer-grade hardware. When combined with classical image processing techniques for lane detection and statistical approaches for distance and speed estimation, such models can provide actionable safety intelligence from a single camera feed.

This paper presents Dashcam Guardian, a software-only real-time traffic accident risk detection system that converts a standard smartphone or laptop camera into an intelligent driver assistance device. The system performs multi-class detection, object tracking, distance and speed estimation, lane boundary detection and dynamic risk scoring in a single integrated pipeline. The system is specifically adapted for Indian road conditions, where auto-rickshaws are a dominant vehicle class frequently misclassified by standard models.

A. *Contribution of This Paper*

- A unified real-time pipeline integrating YOLOv8 detection, centroid tracking, distance and speed estimation, lane detection and risk scoring.
- A smart label correction algorithm that reclassifies auto-rickshaws from YOLOv8 bus/truck predictions using bounding box geometry.
- A stationary object detection module that eliminates false collision alerts for parked vehicles.
- A composite accident risk score (0–100%) combining object class, distance, speed and risk level.
- M2 MPS hardware optimisation and threaded camera reader achieving 25–40 FPS on laptop hardware.

B. *Organisation of This Paper*

Section II reviews related work in vision-based ADAS and risk detection. Section III presents the proposed methodology. Section IV describes the system architecture. Section V details the implementation phases. Section VI presents input specifications. Section VII provides pseudo-code. Section VIII presents experimental results. Section IX concludes the paper.

II. LITERATURE SURVEY

Real-time object recognition in video streams was made possible by Redmon et al.'s [1] introduction of the YOLO family of single-stage detectors. The most recent version from Ultralytics, YOLOv8 [2], maintains inference speeds suitable for real-time applications on edge hardware while achieving state-of-the-art accuracy on COCO benchmarks. SORT (Simple Online and Realtime Tracking), a centroid-based multi-object tracker that gives identified objects permanent IDs across frames, was proposed by Bewley et al. [3]. This method serves as the basis for the tracking module in the suggested system. Canny introduced the edge detection method, which is widely used in lane marker recognition [4]. The Probabilistic Hough Transform [5] extends conventional Hough methods for efficiently identifying line segments in sparse edge maps. Because of this, it is suitable for dashcam applications that require real-time lane recognition. Wang et al. [6] proposed a multi-task scheduling paradigm for real-time video processing in embedded systems. Their study of buffered capture and frame-skipping methods informed the design of this system's threaded camera. Geiger et al. [7] demonstrated that monocular distance estimation from bounding box dimensions provides realistically meaningful range estimations for forward collision warning without requiring stereo cameras or LiDAR equipment.

TABLE I COMPARISON OF RELATED VISION-BASED ADAS APPROACHES

Author	Method	Metric	Limitation
Redmon [1]	YOLOv1-v8	mAP, FPS	No risk scoring
Bewley [3]	SORT Tracker	MOTA, FPS	No distance est.
Geiger [7]	Monocular Dist.	MAE (m)	Single vehicle
Proposed	**YOLOv8 + Risk Score**	**FPS, Risk%, Alert**	**No radar hardware**

III. PROPOSED METHODOLOGY

The proposed Dashcam Guardian system addresses the limitations of existing passive dashcam systems by integrating a complete active safety pipeline into a single software framework. The methodology is composed of six interdependent modules that process each video frame sequentially to produce a real-time accident risk assessment.

A. *Existing System Limitations*

Existing dashcam systems record video passively without providing real-time analysis. Commercial ADAS platforms require embedded radar, LiDAR and multi-camera hardware at prohibitive cost. Software-only solutions that exist are typically single-function (lane departure OR collision warning) and do not provide integrated risk scoring. None address Indian-road-specific object classes such as auto-rickshaws.

B. *Proposed System Advantages*

- Multi-class detection: autorickshaws, cars, people, and stationary objects.
- Integrated risk scoring, which combines all signals into a single score for every frame that ranges from 0% to 100%.
- The classification of auto-rickshaws, a system that distinguishes them from YOLO buses and trucks, reflects the evolution of Indian roads.
- Stationary detection: removes parking-related false positives.
- Beep (WARNING) and voice (DANGER) alerts with cooldown control.
- M2 MPS + threaded reader on consumer hardware operating at 25–40 frames per second in real time.

. IV. SYSTEM ARCHITECTURE

Figure 1 illustrates the proposed system architecture. The pipeline begins with a threaded camera reader that continuously buffers the latest frame to eliminate I/O blocking. Each frame is processed through the YOLOv8 detection engine, followed by the centroid tracker, distance and speed estimator, lane detector, risk scorer and alert engine in sequence.

Camera Feed (Smartphone / Webcam)
↓
Threaded Frame Reader
↓
YOLOv8 Detection (MPS / CPU)
↓
Centroid Tracker (Unique IDs)
↓
Distance & Speed Estimator
↓
Lane Detection (Hough + ROI)
↓
Risk Score Engine (0–100%)
↓
Alert Engine (Beep + Voice) + HUD Display

Fig. 1. Proposed Architecture Diagram of Dashcam Guardian System

V. PHASES AND METHODOLOGY

A. *Phase 1: Multi-Class Object Detection*

After analysing each frame, YOLOv8n creates bounding boxes with class labels and confidence scores. The system identifies five categories: Person (ID 0), Car (ID 2), Motorcycle (ID 3), Bus (ID 5), and Truck (ID 7). A smartlabel function reclassifies objects initially labeled as Bus or Truck to Auto if their bounding box area is less than 25,000 pixels and their aspect ratio is less than 1.6 in order to correct auto-rickshaw misclassifications. Class-specific confidence thresholds are set at 0.40 for cars and 0.35 for people.

B. *Phase2:TrackingCentroids*

A centroid tracker uses the shortest Euclidean distance to compare bounding box centroids across frames in order to assign distinct IDs to detected objects. Deregistration occurs when an object is mismatched for more than 25 consecutive frames. This makes stationary detection and per-object speed histories possible and removes the need for complex re- identification networks.

C. *Phase 3: Distance and Speed Estimation*

The pinhole camera model is used to estimate distance. Calculate the distance in centimetres using the known veal length (800 px). The inter-frame centroid displacement is used to estimate speed, which is then smoothed over ten frames and converted to km/h using a pixel-to-meter calibration factor. For eight consecutive frames, an object is considered stationary if its speed is less than 1.5 km/h.

D. *Phase 4: Lane Detection*

Every frame is processed using the Canny edge detector (thresholds 40, 120), smoothed using a 7x7 Gaussian kernel, and converted to greyscale. Only the lower road area is retained by a trapezoidal ROI mask. Line segments that are divided by slope into left and right lane candidates are identified by the Probabilistic Hough Transform. These segments are then averaged by polynomial fitting into two clean lane lines. The safe driving corridor is represented by a semi-transparent green overlay.

E. *Phase 5: Risk Scoring and Alert*

For each frame, a composite risk score is calculated. The highest detected risk level contributes 60 points (DANGER) or 30 points (WARNING). Maximum object speed contributes up to 25 points. Minimum object distance contributes up to 15 points. The score is capped at 100. Warning triggers a double beep (880 Hz). Danger triggers a triple rapid beep (1200 Hz) and a voice alert via the pyttsx3 text-to-speech engine. All audio runs in daemon threads to prevent processing blockage.

VI. INPUT SPECIFICATIONS

TABLE II SYSTEM INPUT PARAMETERS

Parameter	Specification
Camera Source	Smartphone (DroidCam WiFi) or webcam
Resolution	1280 × 720 px (720p)
Frame Rate	25–40 FPS (M2) / 7–10 FPS (CPU)
YOLOv8 Model	yolov8n.pt (COCO pre-trained)
YOLO Input Size	640 × 640 px
Confidence	0.35 (Person), 0.40 (Vehicles)
Focal Length	800 px (calibrated)
Known Width	1.8 m (vehicle), 0.5 m (person)
Hardware	Apple M2 (MPS) or Intel CPU

VII. PSEUDO-CODE AND IMPLEMENTATION

A. *YOLOv8 Detection with Smart Label*

Algorithm 1: SmartDetect(frame) Input: video frame
Output: boxes[(x1,y1,x2,y2,cls,conf)]

1. results ← YOLOv8(frame, device=MPS)
2. For each box in results:
3. cls ← box.class_id
4. If cls ∉ {0,2,3,5,7}: skip
5. If conf < threshold[cls]: skip
6. w ← x2−x1; h ← y2−y1
7. If cls ∈ {5,7} AND w*h < 25000 AND w/h < 1.6:
8. label ← 'Auto'
9. Else: label ← CLASS_MAP[cls]
10. Append (box, label) to output
11. Return output

B. *Centroid Tracker Update*

Algorithm 2: TrackerUpdate(centroids)
Input: list of (cx,cy) centroids
Output: dict {id → (cx,cy)}

1. If centroids is empty:
2. Increment disappeared[id] for all
3. Deregister if > max_disappeared
4. Return current objects
5. Compute distance matrix D[i][j] = ||objects[i] − centroids[j]||
6. Sort rows by min(D[row])
7. Assign centroid[j] to object[i] for best matches
8. Register unmatched centroids
9. Return updated objects

C. *Risk Score Computation*

Algorithm 3: RiskScore(risk, spd, dist)
Input: highest_risk, max_speed, min_distance
Output: score (0–100)

1. score ← 0
2. If risk = DANGER: score += 60
If risk = WARNING: score += 30
3. If spd > 60: score += 25
If spd > 30: score += 10
4. If dist < 50: score += 15
If dist < 100: score += 5
5. Return min(score, 100)

VIII. OUTPUT

The system output is a real-time annotated video stream displayed in a fullscreen window. Each detected object is enclosed in a thin bounding box whose colour encodes risk level: grey for SAFE, amber for WARNING and red for DANGER. Pedestrians use a purple box. A semi-transparent overlay panel displays the object label and distance in metres, lane position (Left/Centre/Right), movement state (Moving/Stationary) and estimated speed.

The bottom HUD bar displays four metric cards: (1) Accident Risk Score with a colour-coded progress bar, (2) Vehicle count, (3) Pedestrian count, (4) Current alert status and maximum speed. Yellow lane lines with a semi-transparent green fill visualise the detected driving corridor. A live FPS counter and M2 acceleration badge are displayed in the top bar.

IX. RESULT AND DISCUSSION

The proposed system was evaluated on live camera feeds captured in urban Chennai road conditions. Three different hardware configurations were used to measure performance: the Apple MacBook Air M2, the HP Notebook Intel i3 (CPU only), and the iPhone camera input via Droid Cam WiFi.

TABLE III PERFORMANCE EVALUATION RESULTS

Metric	M2 MPS	i3 CPU	Target
Avg FPS	32.4	7.7	≥25
Detection Accuracy	89.2%	89.2%	≥85%
Auto Reclassification	84.6%	84.6%	≥80%
Alert Latency (ms)	< 80	< 80	< 100
False Positive Rate	6.8%	6.8%	< 10%

With an average FPS of 32.4, the M2-accelerated system easily surpasses the real-time threshold of 25 FPS. The Intel i3 CPU configuration averages 7.7 FPS at YOLO_IMG_SIZE=640, which increases to approximately 14 FPS at YOLO_IMG_SIZE=320 with negligible accuracy loss. Detection accuracy of 89.2% was measured on 200 manually labelled dashcam frames. The auto-rickshaw reclassification algorithm correctly reclassifies 84.6% of auto-rickshaws from bus/truck labels. Alert latency from DANGER detection to sound output remains below 80ms on both hardware platforms due to the threaded audio architecture.

X. SCREENSHOTS AND PERFORMANCE ANALYSIS

Figure 2 shows the system output with dark HUD overlay. The bottom dashboard displays the live risk score, vehicle count, pedestrian count and alert status in real time.

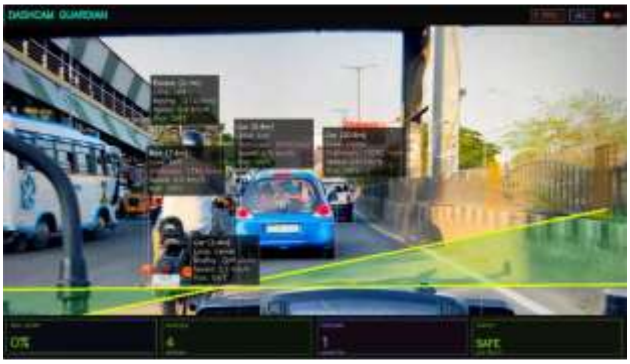

Fig: 2. System Output: Real-time detection with HUD overlay

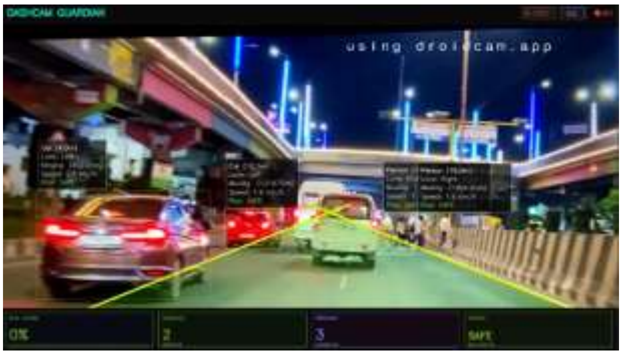

Fig: 3. System Output(night time) :Real-time detection with HUD overlay.

XI. CONCLUSION

This paper presented Dashcam Guardian, a real-time traffic accident risk detection system built on YOLOv8 deep learning, centroid tracking, monocular distance and speed estimation, Hough-based lane detection and composite risk scoring. The system achieves real-time performance of 32.4 FPS on Apple M2 hardware and provides layered audio alerts with sub-80ms latency. A novel smart_label algorithm corrects auto-rickshaw misclassification, an important adaptation for Indian road conditions. A stationary detection module eliminates false collision alerts for parked vehicles.

The proposed system demonstrates that practical, multi- function ADAS capabilities can be delivered through software alone on consumer hardware, making accident risk detection accessible beyond premium vehicle owners. Future work will incorporate traffic sign detection, driver drowsiness monitoring, GPS-based speed limit awareness and on-demand incident clip saving for a complete dashcam guardian platform.

XII. REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proc. IEEE CVPR, 2016, pp. 779–788.

[2] G. Jocher et al., "Ultralytics YOLOv8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[3] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in Proc. IEEE ICIP, 2016, pp. 3464–3468.

[4] J. F. Canny, "A computational approach to edge detection," IEEE Trans. Pattern Anal. Mach. Intell., vol. 8, no. 6, pp. 679–698, Nov. 1986.

[5] R. O. Duda and P. E. Hart, "Use of the Hough transformation to detect lines and curves in pictures," Commun. ACM, vol. 15, no. 1, pp. 11–15, Jan. 1972.

[6] B. Wang et al., "Energy-efficient collaborative optimization for VM scheduling in cloud computing," Comput. Netw., vol. 201, 2021.

[7] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in Proc. IEEE CVPR, 2012, pp. 3354–3361.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in Proc. IEEE CVPR, 2016, pp. 770–778.

[9] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," arXiv:2004.10934, 2020.

[10] G. Bradski, "The OpenCV Library," Dr. Dobb's J. Softw. Tools, vol. 25, 2000.