applied sciences

MDPI

Article

Traffic Accident Detection from Sudden Vehicle Stops in Surveillance Camera Videos

Yılmaz Dikilitaş [1,*] and Ahmet Sayar [2,3]

[1] Department of Computer Engineering, Faculty of Engineering, Kocaeli University, Kocaeli 41001, Türkiye
[2] Department of Artificial Intelligence Engineering, Bahçesehir University, Istanbul 34353, Türkiye; ahmet.sayar@bau.edu.tr or ahmet.sayar@kocaeli.edu.tr
[3] Kocaeli Üniversitesi Teknopark, Bidist, Kocaeli 41275, Türkiye
* Correspondence: yilmaz.dikilitas@kocaeli.edu.tr

Abstract

Accident detection in surveillance video can shorten emergency response times, but a practical detector must also raise few false alarms in normal traffic. This study evaluates an interpretable pipeline that detects vehicles with YOLOv3, tracks them with MOSSE, and raises an alarm when a pixel-based motion-change score exceeds a fixed threshold. On 15 videos that all contain accidents, the pipeline detected 60% of the accidents. However, on the public TAD benchmark, which also contains normal traffic, its score did not separate accident videos from normal videos (test AUC 0.48), and it raised alarms in 14 of 16 normal videos. Controlled experiments traced this failure mainly to tracking errors and the pixel-based score. We therefore propose a configuration that detects vehicles with YOLOv3, tracks them with ByteTrack, and scores each vehicle by its speed just before it stops suddenly. It raised the test AUC to 0.88 and the F1 score from 0.67 to 0.83. Its ranking performance was statistically indistinguishable from that of a fine-tuned video transformer (AUC 0.84), although it needs no training on accident videos and identifies the vehicle and frame behind each alarm. Because false alarms remain frequent, the method is suited only to pre-screening by human operators.

Keywords: traffic accident detection; traffic surveillance; vehicle tracking; ByteTrack; YOLOv3; sudden stop; video analysis

1. Introduction

Road traffic crashes remain a major global public-health problem. According to the World Health Organization (WHO), approximately 1.16 million people die each year in road traffic crashes, and another 20–50 million people suffer non-fatal injuries. Road traffic injuries are the leading cause of death among children and young adults aged 5–29 years, and approximately 92% of road traffic deaths occur in low- and middle-income countries [1]. Because post-crash care is highly time-sensitive, shortening the time between an accident and its detection can help emergency services respond faster and can improve traffic management.

Intelligent transportation systems combine sensing, communication, and automated data analysis to make traffic safer and more efficient. Traffic surveillance cameras are particularly valuable in this context because they continuously observe road segments, intersections, and other high-risk locations. However, operators who monitor many camera streams manually can easily miss events. An automated video-analysis system can support

Check for updates

Academic Editor: Suchao Xie

Received: 6 October 2026
Revised: 3 November 2026
Accepted: 4 November 2026
Published: 9 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

them by reporting candidate accidents together with the corresponding frame, time, and vehicle. This study focuses on the visual identification of such candidate accidents in video; sending alerts to emergency services or other systems is outside its scope.

Vision-based analysis of traffic accidents covers several related but distinct tasks, which differ in when they act relative to a collision. *Crash-risk prediction* estimates how likely a crash is at a given location and time from traffic conditions or traffic conflicts, before any crash has occurred (Section 2.1); it supports preventive measures but does not observe an actual accident. *Accident anticipation* predicts an imminent collision a few seconds before it happens. *Accident detection* recognizes an accident during or immediately after its occurrence [2]. Finally, the *assessment of consequences*, such as the severity of a crash, injured road users, or blocked lanes, follows detection. Recent reviews show that detection and anticipation remain difficult because accidents are rare, brief, visually diverse, and strongly dependent on interactions among vehicles, pedestrians, and the scene [2]. The present study addresses only accident detection. It neither predicts crash risk nor anticipates collisions, and it does not assess the consequences of an accident: its output is a candidate alarm, linked to a vehicle and a frame, that a human operator verifies.

Existing video-based methods for accident detection fall into three broad groups. The first group classifies frames or short video segments directly from spatial and temporal features. For example, Robles-Serrano et al. used deep learning to distinguish accident from non-accident video sequences [3], and more recent models add temporal learning, recurrent units, and optical flow to represent motion across consecutive frames [4,5]. The second group treats accidents as anomalies that differ from normal driving; the DoTA framework, for instance, detects traffic anomalies in driving videos without supervision [6]. These methods can model complex spatial and temporal relationships, but their performance depends strongly on the size, diversity, and balance of the training data.

The third group uses a modular pipeline of detection, tracking, and analysis. Road users are first detected and tracked, and their trajectories, distances, directions, or changes in motion are then evaluated. Ghahremannezhad et al., for example, combined YOLOv4 detection, Kalman-filter tracking, Hungarian association, and trajectory-conflict analysis to detect accidents at urban intersections [7]. Such pipelines are comparatively interpretable, because the detected vehicles, their trajectories, and the decision rule can be inspected separately. However, errors propagate through them: missed detections, identity switches, fluctuating boxes, and tracker drift directly distort the measured motion.

Reliable detection and tracking are therefore essential for trajectory-based accident analysis. YOLOv3 is a single-stage object detector with a good balance between speed and localization accuracy [8]. The Minimum Output Sum of Squared Error (MOSSE) tracker follows an object from frame to frame with adaptive correlation filters at low computational cost [9]. More recent multi-object trackers, such as ByteTrack, show that associating both high- and low-confidence detections reduces missed objects and fragmented trajectories [10]. These developments also reveal a weakness of simple single-object trackers: once such a tracker loses a vehicle or drifts toward the background, its trajectory no longer describes the motion of the vehicle.

The pipeline examined in this study was originally designed along these lines. It runs YOLOv3 periodically, follows each vehicle with MOSSE between detector runs, matches detections to tracks by the distance between their box centers, and raises an alarm when a motion-change score computed from center displacements exceeds a fixed threshold. It needs no camera-specific background model, and its detector is not trained on the evaluation videos.

An important distinction must be made between the displacement of a box in the image and the physical speed of a vehicle. Box displacement is measured in pixels and

depends on the frame rate, camera perspective, focal length, distance to the camera, and road geometry. It therefore cannot be converted into a physical speed, such as $m\,s^{-1}$ or $km\,h^{-1}$, without camera calibration, a perspective transformation, or known real-world distances [11]. The present study does not estimate metric speed or physical acceleration; all motion quantities are measured in the image.

In an initial evaluation on 15 accident videos, the original pipeline detected 60% of the accidents. Such an evaluation, however, cannot show how often a detector raises alarms in normal traffic, which matters as much for an alerting system as its detection rate. We therefore evaluated the pipeline on TAD, a public benchmark of surveillance videos that contains both accidents and normal traffic [12]. On this benchmark, the original motion-change score did not separate accidents from normal traffic. Controlled experiments traced this failure to the tracker and to the definition of the score. These findings led to a proposed configuration that keeps the modular and interpretable structure but links detections with ByteTrack and scores each track by the speed at which it comes to a sudden stop, measured relative to the apparent size of the vehicle.

The main contributions of this study are as follows:

1. A benchmark evaluation of the original pipeline, which combines periodic detection with correlation-filter tracking, on accident and normal surveillance videos. Recall, precision, specificity, false alarms per hour, and runtime are reported, and they show that its pixel-based score performs at chance level;
2. Controlled experiments that vary the detector, the tracker, the detection interval, and the normalization of the score. They identify tracker drift and the pixel-based, frame-to-frame score as the limiting factors and show that periodic detection mainly affects speed;
3. A scale-normalized sudden-stop score that requires no training on labeled accident videos. Combined with YOLOv3 and ByteTrack, it reaches a test AUC of 0.88 on TAD; its ranking performance is statistically indistinguishable from that of fine-tuned 3D-CNN and video-transformer classifiers, while it identifies the vehicle and frame that trigger each alarm and runs at 67 frames per second; and
4. An analysis of failure modes, of a resolution bias in the benchmark, and of the false alarms, which shows that such detectors are suited to pre-screening by human operators.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the baseline and proposed configurations. Section 4 describes the datasets, the evaluation protocol, and the reference methods. Section 5 presents the results, Section 6 discusses them together with the limitations of the study, and Section 7 concludes the paper.

2. Related Work

Vision-based analysis of traffic accidents draws on four closely related lines of research: motion- and trajectory-based accident detection, end-to-end spatio-temporal deep learning, vehicle detection and multi-object tracking, and benchmark datasets recorded by surveillance cameras. These lines differ in their input, their annotation requirements, their computational cost, and their interpretability. This section reviews each of them and positions the present study.

2.1. Motion- and Trajectory-Based Accident Detection

Early camera-based systems estimated vehicle motion directly from image measurements. Ibrahim et al. proposed the Speed Detection Camera System (SDCS), which detected moving objects by combining adaptive background subtraction with frame differencing,

tracked their centers, and computed their speed from the number of frames that an object needed to cross the scene [13]. Its reliability depended on stable camera conditions, successful foreground segmentation, and a known relationship between image coordinates and real-world distances. Xia et al. divided each frame into non-overlapping blocks weighted by the average motion observed during training, modeled normal traffic scenes with a low-rank matrix approximation, and detected an accident when the approximation error increased [14]. Because the model of normal motion is learned from non-accident videos of the target road, this approach must be adapted to each camera.

Object-level pipelines replace global motion with the trajectories of individual road users. Wang et al. enhanced low-visibility frames with the Retinex algorithm, detected road users and post-crash states (e.g., fallen road users and overturned vehicles) with a trained YOLOv3 detector, and identified crashes with a decision model built on the detector outputs [15]. They reported a detection rate of 92.5% and a false-alarm rate of 7.5% under their experimental conditions, but the approach required specialized object categories and scenario-dependent rules. Ghahremannezhad et al. combined YOLOv4 detection, Kalman-filter tracking, Hungarian association, and trajectory-conflict analysis to detect accidents at urban intersections [7]. Yang et al. used multi-vehicle trajectory data and a deep convolutional network to distinguish accident trajectories from normal motion and to classify six types of freeway accidents [16]. A related line of work estimates crash *risk* instead of detecting an observed collision: Yu et al. modeled crash probability from traffic conditions with convolutional networks [17], and Zheng and Sayed estimated the crash risk of each signal cycle at signalized intersections from traffic conflicts observed by cameras, using extreme-value models [18].

Trajectory-based methods are attractive because every intermediate result—detections, tracks, and motion features—can be inspected. Their main weakness is error propagation: missed detections, identity switches, box jitter, and tracker drift directly distort the motion features on which the decision is based. The experiments in Section 4 quantify this effect for the pipelines studied here.

2.2. End-to-End Spatio-Temporal Deep Learning

A second family of methods learns accident representations directly from the pixels. Robles-Serrano et al. classified accident and non-accident video sequences with deep spatio-temporal features [3], and later models used temporal learning, recurrent units, optical flow, and ensembles to represent motion across consecutive frames [4,5]. Lu et al. fused appearance and motion features to balance detection speed and accuracy [19]. These methods build on general video-recognition networks. R3D applies three-dimensional convolutions over space and time, and R(2+1)D factorizes them into separate spatial and temporal convolutions [20]; SlowFast networks process a video at two temporal rates [21]; and multiscale video transformers such as MViTv2 model long-range spatio-temporal dependencies with pooling attention [22]. Such networks are usually pretrained on large action-recognition datasets such as Kinetics [23] and then fine-tuned for accident detection.

Because accidents are rare and precise temporal annotations are expensive, several studies use weak supervision. Sultani et al. formulated anomaly detection in surveillance video as multiple-instance learning (MIL) over video segments, which requires only one label per video [24]. Unsupervised methods model accidents as deviations from normal traffic, as in the DoTA framework for driving videos [6]. Recent surveys emphasize that, despite these advances, accident detection remains difficult because accidents are brief, visually diverse, and strongly dependent on interactions among road users and on the scene [2]. End-to-end models are generally more accurate than rule-based pipelines,

but they require labeled accident videos, are hard to interpret, and are computationally expensive.

2.3. Vehicle Detection and Multi-Object Tracking

Modular pipelines are only as good as their detection and tracking components. YOLOv3 is a single-stage detector with a good trade-off between speed and localization accuracy [8], and later YOLO versions, such as YOLO11 [25], are more accurate at a similar inference cost. The MOSSE tracker follows an object from frame to frame with adaptive correlation filters at very high speed [9]. As a single-object tracker based only on appearance, however, it has no motion model and can drift toward the background when the vehicle is occluded, changes scale, or moves abruptly. Tracking-by-detection methods overcome these problems by linking detections across frames with a motion model: SORT combines a Kalman filter with Hungarian assignment [26], and ByteTrack also uses low-confidence detections to reduce missed objects and fragmented trajectories [10]. Previous studies by our group addressed real-time vehicle detection and classification in surveillance video streams [27] and the detection of traffic-rule violations in video streams [28].

2.4. Surveillance-Based Accident Benchmarks

Accident detectors have been difficult to evaluate because few public surveillance datasets contain both accidents and normal traffic. UCF-Crime contains a road-accident category among its thirteen anomaly classes, together with normal surveillance videos [24]. CADP provides videos from CCTV traffic cameras for accident analysis [29]. TAD is a recent large benchmark recorded by surveillance cameras; it contains accident and normal videos with one label per video and provides baseline results for image classification, object detection, and video classification [12]. Because its normal videos were recorded under comparable conditions, TAD makes it possible to measure false alarms and specificity, which are essential for practical alerting systems and cannot be measured on a dataset that contains only accidents.

2.5. Positioning of the Present Study

The present study belongs to the modular, trajectory-based family. It does not propose a new detector or tracker, and it does not claim to be the first to use motion or trajectory changes for accident detection. Instead, it (i) evaluates an interpretable pipeline based on periodic detection, correlation-filter tracking, and a pixel-based motion-change rule on a public surveillance benchmark that contains both accident and normal videos; (ii) quantifies, in controlled experiments, how the detector, the tracker, the detection interval, and the score definition affect accuracy, false alarms, and speed; and (iii) proposes a scale-normalized sudden-stop score that requires no training on accident videos and compares it with modern tracking-by-detection variants and with end-to-end 3D-CNN and video-transformer classifiers. Because neither camera calibration nor real-world distances are used, all motion quantities are measured in the image and must not be interpreted as physical vehicle speed or acceleration.

3. Materials and Methods

3.1. System Architecture

The framework processes a traffic video frame by frame and reports candidate accidents, each linked to an individual vehicle track. As shown in Figure 1, it consists of three stages: vehicle detection, vehicle tracking and association, and a motion-based score. We study two configurations of these stages. The *baseline* configuration is the originally proposed pipeline: YOLOv3 runs every N_d frames, a MOSSE tracker follows each vehi-

cle between detector runs, and a pixel-based motion-change score is compared with a threshold (Sections 3.3–3.6). The *proposed* configuration keeps the modular structure and interpretability of the baseline, but it replaces the single-object correlation-filter tracker with tracking-by-detection and the pixel-based score with a scale-normalized sudden-stop score (Sections 3.7 and 3.8). In both configurations, the output is a candidate accident alarm with the corresponding frame, time, and track identifier; it is not proof that a collision has occurred.

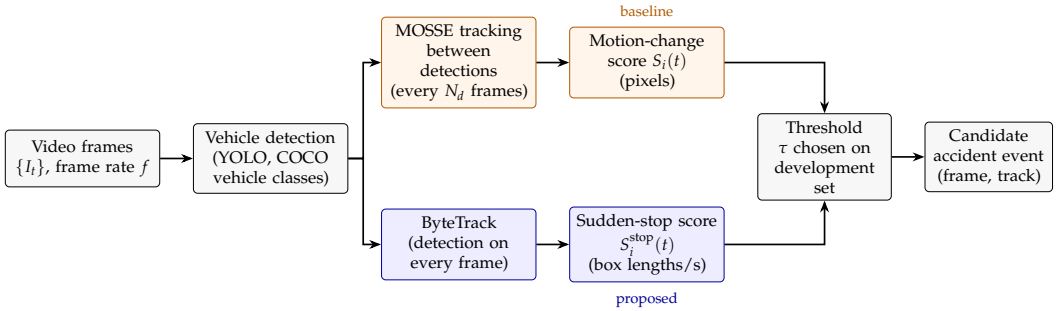


Figure 1. Overall workflow of the framework. The baseline configuration (orange) uses periodic detection, MOSSE tracking, and a pixel-based motion-change score. The proposed configuration (blue) uses tracking-by-detection with ByteTrack and a scale-normalized sudden-stop score. The decision threshold is chosen using only the development set of TAD, not the test set (Section 4.1).

3.2. Implementation and Parameters

The baseline pipeline was originally implemented in Python 3.6.8 with TensorFlow 2.1.0, OpenCV (Version 4.1.2) , and NumPy (Version 1.18.1) and it was used for the preliminary experiments on 15 accident videos (Section 5.1). For the benchmark experiments, all pipelines were re-implemented in Python with PyTorch (Version 2.11.0) and torchvision as provided by the Google Colaboratory runtime of September 2026, the Ultralytics library [25] (release available in September 2026), OpenCV-contrib 4.10.0.84 (MOSSE tracker), and the `supervision` 0.25.1 implementation of ByteTrack [10]. Both detectors use the COCO-pretrained weights distributed by Ultralytics. The YOLOv3 weights (`yolov3u.pt`) belong to the Ultralytics port of YOLOv3, which keeps the Darknet-53 backbone and multi-scale neck of YOLOv3 [8] but uses an anchor-free detection head; they are applied at the original input size of 416 × 416 pixels. YOLO11m (`yolo11m.pt`, 640 × 640 input) [25] serves as a modern detector for comparison. No detector was fine-tuned on the evaluation videos. The end-to-end reference models use the Kinetics-400 weights of torchvision (`R3D_18_Weights.KINETICS400_V1` and `MViT_V2_S_Weights.KINETICS400_V1`). All random operations use the seed 42. All fixed parameters are listed in Table 1; the parameters of the sudden-stop score were selected on the development set only (Section 3.8).

The choice of components follows the aim of the study: to find out how far an interpretable decision rule that needs no training on accident videos can go on top of generic vehicle detection and tracking, and which components limit its performance. YOLOv3 and MOSSE are kept because they form the originally proposed pipeline, whose behavior on accident and normal traffic had not been measured. Rather than replacing them silently, we treat them as a baseline and isolate their effects by replacing one component at a time: the detector (YOLOv3 versus YOLO11), the tracker (MOSSE versus ByteTrack versus detection-only association), the detection interval N_d, and the score definition. End-to-end spatio-temporal classifiers, which require labeled training videos, are evaluated separately as reference methods (Section 4.3).

Table 1. Components and fixed parameters of the baseline and proposed configurations.

Component/Parameter	Setting	Function
Detection (both configurations)		
Detector	YOLOv3 (`yolov3u`), 416×416; YOLO11m, 640×640	COCO-pretrained vehicle localization and classification.
Vehicle classes	car, motorcycle, bus, truck	COCO classes kept after detection.
Confidence/NMS IoU	0.50/0.45	Filtering of detections (ByteTrack also uses detections with confidence ≥ 0.10).
Baseline configuration		
Detection interval	$N_d = 5$ frames	YOLO runs on frames $t \equiv 0 \pmod{N_d}$.
Tracker	MOSSE	Frame-to-frame update of each vehicle box.
Bounding-box expansion	$\gamma = 1.25$	Enlargement of a detection box when a MOSSE tracker is initialized.
Area-change threshold	$\tau_A = 0.50$	Refresh of the tracked box dimensions.
Motion window	$K = 5$ frames	Frames per displacement window.
Event threshold	$\tau_S = 11$ pixels	Original threshold of the motion-change score.
Proposed configuration		
Tracker	ByteTrack	High/low confidence 0.25/0.10, matching threshold 0.8, track buffer f frames (1 s).
Gap filling/smoothing	$\leq$0.5 s/0.2 s	Linear interpolation of short track gaps; moving-average smoothing of centers.
Pre-event window	$T_b = 0.5$ s	Speed before a candidate stop.
Guard interval	$T_g = 0.2$ s	Time allowed for the deceleration.
Post-event window	$T_a = 1.0$ s	Time during which the vehicle must remain (almost) stationary.
Stationarity threshold	$\epsilon_v = 0.2$ box lengths/s	Maximum median speed after the stop.
Event threshold	τ_{stop}	70th percentile of the scores of normal development videos (Section 3.9).

3.3. Vehicle Detection

Each frame passed to the detector is letterboxed to the input size of the network. For every detected object, YOLO outputs a class, a confidence score, and a bounding box. Detections below the confidence threshold are removed, non-maximum suppression is applied, and only the vehicle classes listed in Table 1 are kept. The box coordinates are then mapped back to the original video resolution.

A bounding box for vehicle i in frame t is represented as

$$B_i^Y(t) = \left[x_i^Y(t), y_i^Y(t), w_i^Y(t), h_i^Y(t)\right], \tag{1}$$

where $x_i^Y(t)$ and $y_i^Y(t)$ are the coordinates of the upper-left corner, and $w_i^Y(t)$ and $h_i^Y(t)$ are the width and height of the box.

3.4. Baseline Tracking with MOSSE

In the baseline configuration, YOLO runs on the first frame and then every N_d frames. Between two detector runs, each vehicle is followed by its own MOSSE correlation-filter tracker [9]. This reduces the number of detector runs and yields a continuous sequence of center points. For a bounding box

$$B_i(t) = [x_i(t), y_i(t), w_i(t), h_i(t)], \tag{2}$$

the center point $\mathbf{c}_i(t)$is

$$\mathbf{c}_i(t) = \big(c_{x,i}(t), c_{y,i}(t)\big) = \left(x_i(t) + \frac{w_i(t)}{2}, y_i(t) + \frac{h_i(t)}{2}\right). \tag{3}$$

Figure 2 shows a bounding box and its center point. The center coordinates are used both to match detections to tracks and to analyze motion.

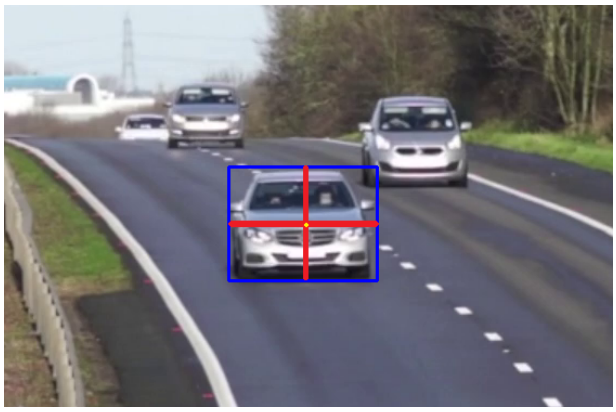


Figure 2. A vehicle bounding box and its center point in the image.

When a MOSSE tracker is initialized, the detection box is enlarged around its center by the factor $\gamma = 1.25$:

$$B_i^{\gamma}(t) = \left[c_{x,i}(t) - \frac{\gamma w_i(t)}{2}, c_{y,i}(t) - \frac{\gamma h_i(t)}{2}, \gamma w_i(t), \gamma h_i(t)\right]. \tag{4}$$

The enlargement gives the tracker some context around the vehicle. However, a MOSSE tracker initialized at one scale becomes inaccurate when the vehicle approaches or moves away from the camera, and, because it has no motion model, it can drift toward the background when the vehicle is occluded or moves abruptly. The periodic detections are therefore also used to refresh the size of the tracked boxes, and a track is ended when the MOSSE update fails or the center of the box leaves the image.

3.5. Detection–Tracking Association

Whenever the detector runs, the detected boxes are matched to the active MOSSE tracks. Let $\mathbf{c}_i^Y(t)$ be the center of detection i and $\mathbf{c}_j^M(t)$ the center of MOSSE track j. Their Euclidean distance is

$$d_{ij}(t) = \left\| \mathbf{c}_i^Y(t) - \mathbf{c}_j^M(t) \right\|_2. \tag{5}$$

The matching procedure repeatedly selects the unmatched detection–track pair with the smallest center distance and removes both from the candidate sets. This produces a one-to-one greedy matching without a maximum distance. Every matched track keeps its identifier. Detector and tracker boxes can differ in area. Let

$$A_i^Y(t) = w_i^Y(t) h_i^Y(t) \quad \text{and} \quad A_j^M(t) = w_j^M(t) h_j^M(t) \tag{6}$$

be the areas of a matched detection box and MOSSE box, respectively. Their relative area change is

$$\Delta A_{ij}(t) = \frac{\left| A_i^Y(t) - A_j^M(t) \right|}{\max\left(A_j^M(t), \varepsilon\right)}, \tag{7}$$

where ε is a small positive constant. If $\Delta A_{ij}(t) \geq \tau_A$, the MOSSE box keeps its center but takes the width and height of the detection:

$$\widetilde{B}_j^M(t) = \left[c_{x,j}^M(t) - \frac{w_i^Y(t)}{2}, c_{y,j}^M(t) - \frac{h_i^Y(t)}{2}, w_i^Y(t), h_i^Y(t) \right]. \tag{8}$$

Keeping the tracker center prevents a sudden jump of the center caused only by the change in box size. Unmatched detections start new MOSSE trackers. Algorithm 1 summarizes the procedure.

Algorithm 1 Baseline detection–tracking association and bounding-box refresh

Require: Detections $\mathcal{D}_t$, active MOSSE tracks $\mathcal{T}_{t-1}$, expansion factor γ, area threshold τ_A
Ensure: Updated track set $\mathcal{T}_t$
Update every track in $\mathcal{T}_{t-1}$ with MOSSE; end tracks whose update fails or whose center leaves the image
Compute the center and area of every detection and track
$\mathcal{D} \leftarrow \mathcal{D}_t$; $\mathcal{T} \leftarrow \mathcal{T}_{t-1}$; $\mathcal{M} \leftarrow \emptyset$
while $\mathcal{D} \neq \emptyset$ **and** $\mathcal{T} \neq \emptyset$ **do**
    $(i^\star, j^\star) \leftarrow \arg\min_{i \in \mathcal{D}, j \in \mathcal{T}} d_{ij}(t)$
    $\mathcal{M} \leftarrow \mathcal{M} \cup \{(i^\star, j^\star)\}$; remove $i^\star$ from $\mathcal{D}$ and $j^\star$ from $\mathcal{T}$
end while
for all $(i, j) \in \mathcal{M}$ **do**
    Keep the identifier of track j
    if $\Delta A_{ij}(t) \geq \tau_A$ (Equation (7)) **then**
        Refresh the box using Equation (8) and reinitialize MOSSE
    end if
end for
for all unmatched detection $i \in \mathcal{D}$ **do**
    Enlarge $B_i^Y(t)$ using Equation (4); start a new MOSSE tracker and track ID
end for
return $\mathcal{T}_t$ (unmatched tracks keep their MOSSE prediction)

3.6. Baseline Motion-Change Score

The baseline decision stage examines the history of the center points of each track. These are image coordinates in pixels; the stage therefore does not estimate physical speed or acceleration, and we call the computed quantity the *motion-change score*. Table 2 lists the information stored for each track.

Table 2. Information stored for each active vehicle track.

No.	Field	Description
1	`track_id`	Persistent identifier of the vehicle track.
2	`bounding_box`	Current position and size of the vehicle bounding box.
3	`frame_age`	Number of consecutive frames in which the vehicle has been tracked.
4	`center_history`	Ordered history of recent center points.
5	`matched`	Whether the track was matched to a detection in the current detector frame.
6	`displacement`	Displacement of the center over the current window.
7	`score`	Motion-change score (baseline) or sudden-stop score (proposed).
8	`candidate_flag`	Whether the score exceeds the event threshold.

Let $K = 5$ be the number of frames in a motion window. The displacement of vehicle i over the window that ends at frame t is

$$D_i(t) = \|\mathbf{c}_i(t) - \mathbf{c}_i(t-K+1)\|_2, \tag{9}$$

and the displacement over the window shifted back by one frame is $D_i(t-1) = \|\mathbf{c}_i(t-1) - \mathbf{c}_i(t-K)\|_2$. The two windows share $K-1$ frames, so $K+1$ consecutive center points are needed. The motion-change score is the absolute difference between the two displacements:

$$S_i(t) = |D_i(t) - D_i(t-1)|. \tag{10}$$

Figure 3 illustrates the two overlapping windows. A vehicle is flagged when $S_i(t) > \tau_S$, with $\tau_S = 11$ pixels in the original implementation.

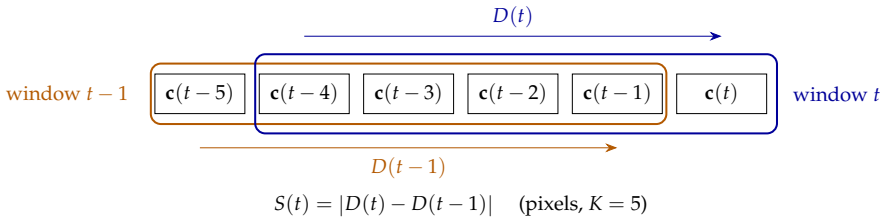


Figure 3. The two overlapping five-frame windows of the baseline motion-change score. For a vehicle that moves at a constant velocity in the image, $D(t) = D(t-1)$ and the score is zero.

This score has three structural limitations. First, it is measured in pixels, so the same physical event produces larger scores in higher-resolution images and for vehicles closer to the camera. Second, the fixed window of K frames lasts $\Delta T_K = (K-1)/f$, which depends on the frame rate f. Third, because the two windows differ by only one frame, $S_i(t)$ is dominated by frame-to-frame jitter of the tracked box, and any tracker drift or box re-initialization directly produces a score peak. To isolate the first two effects, we also evaluate two normalized variants: a frame-rate-normalized score, in which K spans a fixed duration of 4/25 s and $S_i(t)$ is multiplied by $f/25$, and a scale-normalized score, in which $S_i(t)$ is divided by $\sqrt{w_i(t)h_i(t)}$.

3.7. Proposed Tracking with ByteTrack

In the proposed configuration, the detector runs on every frame and ByteTrack links the detections into tracks [10]. ByteTrack predicts each track with a Kalman filter and matches detections in two steps: high-confidence detections are matched first by their intersection over union, and the remaining tracks are then matched to low-confidence detections. This second step recovers partially occluded vehicles without creating new identities. Compared with MOSSE, the motion model prevents drift toward the background, and the matching criterion prevents a track from jumping to a distant vehicle. As a further variant, a detection-only association links the detections of consecutive frames by greedy center matching with a maximum distance proportional to the box size, without any visual tracker.

3.8. Scale-Normalized Sudden-Stop Score

In collisions recorded by surveillance cameras, one or more vehicles usually come to rest within a fraction of a second, whereas stops in normal traffic, at signals or in queues, are gradual. The proposed score therefore targets *sudden stops*. It measures speed relative to the apparent size of the vehicle, which removes the dependence on image resolution and reduces the dependence on the distance to the camera.

For each track, gaps of at most 0.5 s are filled by linear interpolation, and the center trajectory $\mathbf{c}_i(t)$ is smoothed with a 0.2 s moving average. Let $s_i = \text{median}_t \sqrt{w_i(t)h_i(t)}$

be the characteristic size of the vehicle in pixels, computed over each continuous track segment. The normalized speed, in vehicle lengths per second, is

$$v_i(t) = \frac{f}{s_i} \left\| \frac{\mathbf{c}_i(t+1) - \mathbf{c}_i(t-1)}{2} \right\|_2. \tag{11}$$

The characteristic size s_i is the geometric mean of the box width and height, that is, the side length of a square with the same area as the box. The reason for this normalization is geometric. Under a pinhole camera model, a vehicle of physical length L at distance Z from the camera appears with an image size of approximately $f_c L/Z$, and a lateral displacement ΔX produces an image displacement of approximately $f_c \Delta X/Z$, where f_c is the focal length in pixels. The ratio of the two, $\Delta X/L$, depends neither on f_c nor on the image resolution nor on Z. Dividing the speed in pixels by a size in pixels therefore gives an approximate speed in vehicle lengths per second that is comparable across cameras and distances. We use the geometric mean rather than the width or height alone because it is less sensitive to changes in the shape of the box, for example when a vehicle turns, and the median over the track segment suppresses frame-to-frame fluctuations of the box size.

This normalization is approximate, and three effects remain. (i) Camera perspective: motion along the viewing direction produces a much smaller image displacement than lateral motion at the same speed, so vehicles that approach or move away from the camera receive lower normalized speeds. (ii) Changing distance: because s_i is a median over the segment, a vehicle whose distance to the camera changes substantially is normalized by an average size. (iii) Vehicle type and orientation: the box of a car seen from the side is longer than that of the same car seen from the front, and a truck has a larger box than a car, so the unit "vehicle length" differs between vehicles. These effects mainly change the speed before the stop, that is, the magnitude of the score. The stationarity test after the stop is hardly affected, because a stopped vehicle has a speed close to zero under any normalization. The normalization removes the dependence on image resolution (Section 6.2), but the effects of perspective and orientation were not quantified in this study; camera calibration would remove them (Section 6.5).

With $n_b = \mathrm{round}(T_b f)$, $n_g = \mathrm{round}(T_g f)$, and $n_a = \mathrm{round}(T_a f)$ frames, the speeds before and after a possible stop at frame t are the medians

$$\begin{aligned} \bar{v}_i^-(t) &= \mathrm{median}\{v_i(t-n_b), \ldots, v_i(t-1)\}, \\ \bar{v}_i^+(t) &= \mathrm{median}\{v_i(t+n_g), \ldots, v_i(t+n_g+n_a-1)\}, \end{aligned} \tag{12}$$

and the sudden-stop score is

$$S_i^{\mathrm{stop}}(t) = \begin{cases} \bar{v}_i^-(t), & \bar{v}_i^+(t) < \epsilon_v, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

In words, a vehicle receives a positive score at time t only if it becomes (almost) stationary within T_g and remains so for at least T_a. The score equals the speed the vehicle had just before, so fast vehicles that stop abruptly receive the highest scores. The medians make the score robust to isolated box jitter, and the requirement of a sustained stop suppresses short fluctuations. Because the window after the stop ends $T_g + T_a = 1.2$ s after t, the score becomes available with an inherent delay of 1.2 s. The four parameters $(T_b, T_g, T_a, \epsilon_v)$ were selected by a grid search on the development set only ($T_b \in \{0.5, 1, 2\}$ s, $T_g \in \{0.2, 0.5\}$ s, $T_a \in \{1, 2, 3\}$ s, $\epsilon_v \in \{0.1, 0.2, 0.3, 0.5\}$). The ten best settings reached development AUCs within 0.03 of the selected one, so the score is not sensitive to their exact values. Other cues—raw deceleration, change of heading, distance between vehicles, and long stationary periods—were also examined on the development set; they did not improve

on the sudden-stop score alone and were therefore not used. Algorithm 2 summarizes the computation.

Algorithm 2 Scale-normalized sudden-stop score

Require: Track i with frames, centers $\mathbf{c}_i(t)$, and sizes $w_i(t), h_i(t)$; frame rate f; parameters $T_b, T_g, T_a, \epsilon_v$

Ensure: Score sequence $S_i^{\text{stop}}(t)$

1: Split the track at gaps > 0.5 s; interpolate shorter gaps linearly; smooth the centers with a 0.2 s moving average
2: $s_i \leftarrow \text{median}_t \sqrt{w_i(t) h_i(t)}$; compute $v_i(t)$ using Equation (11)
3: **for** each frame t with complete windows before and after t **do**
4: Compute $\bar{v}_i^-(t)$ and $\bar{v}_i^+(t)$ using Equation (12)
5: $S_i^{\text{stop}}(t) \leftarrow \bar{v}_i^-(t)$ **if** $\bar{v}_i^+(t) < \epsilon_v$ **else** 0
6: **end for**
7: **return** $S_i^{\text{stop}}(t)$

3.9. Event-Level Decision and Alarm Aggregation

For both scores, the score of a frame is the maximum over all active tracks, $S(t) = \max_i S_i(t)$, and the score of a video is $\max_t S(t)$. A video is classified as an accident video when its score exceeds the threshold τ. In the benchmark experiments, τ is never tuned on the test set. For the proposed score, it is fixed in advance as the 70th percentile of the scores of the normal videos in the development set, which corresponds to a nominal specificity of 0.70 on the development data. For the baseline score, we report results both for the original $\tau_S = 11$ pixels and for a threshold selected on the development set by maximizing Youden's index. To count false alarms in continuous operation, consecutive frames with $S(t) > \tau$ are grouped into alarm events, and alarms less than 2 s apart are merged into a single event.

Figure 4 shows an example output of the original baseline implementation, with vehicle boxes, center points, motion vectors, and the motion-change score of each track (labeled *ivme*, the on-screen name of the score).

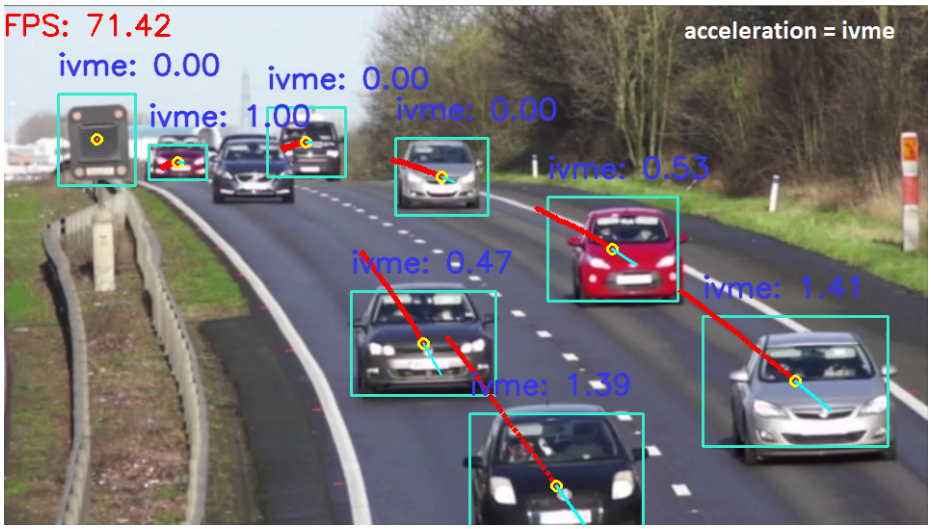


Figure 4. Example output of the original baseline implementation with the tracked vehicles and their motion information in the image. The on-screen label *ivme* is the former name of the motion-change score and does not denote physical acceleration.

4. Experimental Setup

The pipelines were evaluated in two stages. The original baseline was first evaluated on a small corpus that contains only accident videos, so only its detection rate could be

measured (Section 5.1). To overcome this limitation, all configurations were then evaluated on a public surveillance benchmark that contains both accident and normal videos, with thresholds chosen on the development set and performance reported on a separate test set that was not used for any tuning (Section 4.1).

4.1. Datasets

Preliminary accident-only corpus. Fifteen videos that contain accidents were obtained from the KGYS camera-image portal of the Traffic Department of the Turkish General Directorate of Security [30] and from publicly available YouTube videos. The videos differ in resolution, frame rate, illumination, viewpoint, compression quality, and traffic density. Four of them were also used for a manual analysis of tracking. Because the corpus contains no normal traffic, it cannot be used to estimate precision, specificity, or false alarms.

TAD benchmark. The main evaluation uses TAD, a public benchmark of traffic videos recorded by surveillance cameras [12]. The version used here was downloaded from the public release linked from the authors' repository (https://github.com/UnicomAI/UnicomBenchmark/tree/main/TADBench, accessed on 4 September 2006). It contains 404 videos, 277 with an accident and 127 with normal traffic, each with one label per video. TAD is officially divided into two parts, which we use without change. The *development set* (372 videos: 261 accident and 111 normal; 1.76 h in total) was used for all design decisions, that is, to select the parameters and decision thresholds of every method. The *test set* (32 videos: 16 accident and 16 normal; 0.18 h) was used only once, for the final evaluation, and never for any selection. This separation ensures that the reported test results are not biased by tuning. The frame rate ranges from 14 to 54 fps (median 25 fps), the resolution from 228×138 to 2688×1520 pixels (median 1920×1080), and the duration from 1.6 to 89.3 s (median 13.5 s).

The development set has an important property. Most of its normal videos have a high resolution (88 of 111 have 1080 lines), whereas most of its accident videos have a lower resolution (152 of 261 have fewer than 1080 lines). As a result, image resolution alone separates the two classes of the development set with an area under the ROC curve (AUC) of 0.82, whereas it carries no information on the test set (AUC 0.50; 30 of the 32 test videos have 1920×1080 pixels). Any score measured in pixels is affected by this bias. We therefore also report development results on the resolution-matched subset of 197 development videos with 1080 lines (88 normal and 109 accident videos).

Three further subsets were derived from the official division: the resolution-matched subset described above; the training (80%) and validation (20%) subsets of the development set used for the end-to-end models (Section 4.3); and the three accident test videos in which an accident onset could be marked, which form the latency subset (Section 4.2).

4.2. Evaluation Protocol and Metrics

All benchmark results are reported at the video level. A video is predicted to contain an accident if its score exceeds the decision threshold (Section 3.9). Thresholds, score parameters, and model checkpoints were selected on the development set only, and the test set was used once for the final evaluation. We report the following measures on the test set: recall (the detection rate of accident videos), precision, specificity, F1 score, AUC, and average precision (AP). AUC and AP do not depend on a threshold; they measure how well a score ranks accident videos above normal ones. Because the test set is small, 95% confidence intervals (CIs) are estimated with 2000 bootstrap resamples that preserve the number of videos of each class. To describe continuous operation, the false-alarm rate is computed as the number of alarm events per hour of normal video, with alarms less than 2 s apart merged into one event. The normal test videos last only 5.8 min in total, so a

single alarm event changes the test false-alarm rate by approximately 10 events per hour. Test false-alarm rates are therefore coarse, and development values are also given.

Detection latency was measured on the 16 accident videos of the test set. For each video, the first author marked the first frame in which the colliding vehicles touch each other (or a vehicle touches an obstacle), without seeing the output of any method; we call this frame the accident onset. In 13 of the 16 videos, the collision had already happened before the video starts, so no onset could be marked; these videos were excluded from the latency analysis. Because each onset was marked by a single annotator, inter-annotator agreement could not be assessed. The latency of a video is the time from the onset to the first alarm within the interval $[t_{\text{onset}} - 1\,\text{s},\ t_{\text{onset}} + 10\,\text{s}]$; alarms more than 1 s before the onset are counted as early (false) alarms.

4.3. End-to-End Reference Methods

Two end-to-end spatio-temporal classifiers were trained as reference methods: R3D-18, a 3D convolutional network [20], and MViTv2-S, a multiscale video transformer [22]. Both were initialized with Kinetics-400 weights [23]. Because TAD provides only one label per video, the models were trained as multiple-instance classifiers [24]. Each video was represented by eight clips of 16 frames (temporal stride 4) evenly distributed over its duration; the clip outputs (logits) were combined into a video output by taking their maximum, and a class-weighted binary cross-entropy loss was minimized. To train these models, we randomly divided the development set into a training subset (80% of the videos, 297 videos), on which the models were trained, and a validation subset (20%, 75 videos), on which the best checkpoint and the decision threshold were selected. Both subsets contain the same proportion of accident videos as the whole development set. The models were trained for 12 epochs with AdamW (learning rate 10^{-4} for R3D-18 and 3×10^{-5} for MViTv2-S, weight decay 10^{-4}) in mixed precision. The checkpoint with the highest validation AUC was kept, and the decision threshold was selected on the validation subset by maximizing Youden's index. The input frames were resized to 112×112 pixels for R3D-18 and 224×224 pixels for MViTv2-S.

4.4. Experimental Environment

The preliminary experiments were run on a computer with an AMD Ryzen 5 3500 processor, 16 GB of RAM (Advanced Micro Devices, Santa Clara, CA, USA) and an NVIDIA GeForce GTX 1660 SUPER GPU under Windows 10 Pro (NVIDIA Corporation, Santa Clara, CA, USA). The benchmark experiments, including all runtime measurements, were run on Google Colaboratory with an NVIDIA A100-SXM4-40GB GPU. Detector runtimes were measured with a batch size of one, which corresponds to frame-by-frame streaming operation.

5. Results

5.1. Preliminary Evaluation on Accident-Only Videos

Tracking continuity. The proportion of correctly tracked vehicles was determined by manually inspecting one frame every three seconds in four accident videos:

$$R_{\text{track}}(v,t) = \frac{N_{\text{correct}}(v,t)}{N_{\text{visible}}(v,t)} \times 100, \tag{14}$$

where $N_{\text{visible}}(v,t)$ is the number of visible vehicles and $N_{\text{correct}}(v,t)$ the number of vehicles whose MOSSE track was still on the correct vehicle. Table 3 lists the results.

Over the 26 available observations, the mean tracking ratio was 67.73% (54.43%, 83.60%, 67.71%, and 69.71% for Videos 1–4, respectively), and it decreased over time as

occlusion, scale changes, and drift accumulated. These values describe tracking continuity, not the detection accuracy of YOLOv3.

Table 3. Percentage of correctly tracked vehicles (baseline, MOSSE) at three-second intervals in four accident videos of the preliminary corpus.

Time	Video 1	Video 2	Video 3	Video 4	Mean
0 s	87	100	87	80	88.50
3 s	62	87	37	69	63.75
6 s	43	77	100	69	72.25
9 s	66	77	83	65	72.75
12 s	57	77	57	60	62.75
15 s	33	–	55	65	51.00
18 s	33	–	55	80	56.00

The values are percentages. "–" indicates that no observation was available at that time. Each mean is calculated from the available observations in the row.

Event-level detection rate. With $\tau_S = 11$ pixels, nine of the 15 accidents were detected, a detection rate of 60% (95% Wilson CI: 36–80%). Table 4 shows five representative cases. The detected cases reached maximum motion-change scores of 14–18 pixels, whereas several missed cases stayed between 8 and 10 pixels. Missed detections were associated with lost tracks, drifting boxes, too small a change in displacement, partial occlusion, low image quality, and unstable frame rates. Because this corpus contains only accident videos, it cannot show how often the pipeline raises false alarms in normal traffic. False alarms are therefore evaluated on the TAD benchmark in Section 5.2.

Table 4. Representative cases of the preliminary corpus and their maximum baseline motion-change scores.

Case	Outcome	Max. Motion-Change Score (Pixels)	Representative Frame
Case 1	Detected	18	
Case 2	Missed	8	
Case 3	Detected	17	
Case 4	Detected	14	
Case 5	Detected	14	

5.2. Benchmark Comparison

Table 5 compares the baseline, the proposed configuration, and the end-to-end reference methods on the TAD test set. With the original threshold $\tau_S = 11$ pixels, the baseline reaches a high recall (0.938), but it raises alarms in 14 of the 16 normal videos (specificity 0.125), and its AUC of 0.477 shows that the motion-change score does not rank accident videos above normal ones. Selecting the threshold on the development set gives almost the same value ($\tau_S = 12.25$ pixels) and the same test performance, and normalizing the score by frame rate and vehicle size does not help (AUC 0.383). The problem therefore lies in the score and in the tracks on which it is computed, not in the particular threshold.

The proposed configuration raises the test AUC to 0.875 (95% CI: 0.734–0.984) and the F1 score from 0.667 to 0.833. At the threshold fixed on the development set, it detects 15 of the 16 accident videos and correctly rejects 11 of the 16 normal videos, and it reduces the false-alarm rate from 391 to 124 events per hour of normal video. On the development set, where the threshold was chosen, its AUC is 0.828 overall and 0.840 on the resolution-matched subset, which shows that the score is not driven by the resolution bias described in Section 4.1. The end-to-end classifiers reach test AUCs of 0.770 (R3D-18) and 0.844 (MViTv2-S). MViTv2-S reaches the highest specificity (0.812) at a lower recall (0.750).

The two groups of methods were obtained under different training conditions. R3D-18 and MViTv2-S were fine-tuned on 297 labeled development videos and validated on the remaining 75 (Section 4.3). The proposed configuration was not trained on accident videos: its detector and tracker are generic, the development labels were used only to choose the few parameters of the score (Section 3.8), and its threshold depends only on the scores of normal development videos. The numerical differences between the proposed configuration and MViTv2-S should not be over-interpreted. Their AUCs differ by only 0.031 (0.875 versus 0.844), whereas each of their 95% CIs is about 0.25 wide and the two overlap almost completely (0.73–0.98 versus 0.70–0.96); the difference is therefore statistically inconclusive. With 16 videos per class, the test set cannot establish whether one method ranks accidents better than the other; the comparison shows only that the configuration without accident training reaches a similar level. Likewise, the false-alarm rate of 124 events per hour is based on 12 alarm events in 5.8 min of normal video; treating these events as Poisson counts, its 95% interval is approximately 64–216 events per hour. The practical advantages of the proposed configuration therefore lie less in its accuracy figures than in its properties: it does not require labeled accident videos, every alarm is linked to a specific vehicle track and frame, and each intermediate decision can be inspected.

Table 5. Video-level accident detection on the TAD test set (16 accident and 16 normal videos). Thresholds were selected on the development set (end-to-end models: on its validation subset). FA/h: alarm events per hour of normal video. Values in brackets are 95% bootstrap CIs.

Method	Recall	Precision	Specificity	F1	AUC	AP	FA/h
Baseline: YOLOv3 ($N_d = 5$) + MOSSE + motion-change score							
$\tau_S = 11$ pixels (original)	0.938	0.517	0.125	0.667	0.477 [0.28, 0.69]	0.471	391
τ_S from development set	0.938	0.517	0.125	0.667	0.477	0.471	391
fps- and scale-normalized score	0.875	0.538	0.250	0.667	0.383	0.440	381
Proposed: YOLOv3 + ByteTrack + sudden-stop score (no training on accident videos)							
τ_{stop} from development set	0.938	0.750	0.688	**0.833**	**0.875** [0.73, 0.98]	0.826	124
End-to-end reference methods (fine-tuned on labeled development videos)							
R3D-18 [20]	0.750	0.632	0.562	0.686	0.770 [0.60, 0.92]	0.717	–
MViTv2-S [22]	0.750	0.800	0.812	0.774	0.844 [0.70, 0.96]	0.835	–

Bold values indicate the highest F1 score and the highest AUC.

5.3. Ablation of Detector, Tracker, and Detection Interval

Table 6 isolates the contribution of each component and leads to four observations. First, with the baseline score, the detection interval has little effect on accuracy: running the detector on every frame ($N_d = 1$) instead of every fifth frame leaves the test AUC almost unchanged (0.480 versus 0.477) but reduces the throughput from 78 to 42 frames per second. Periodic detection therefore saves computation, but it does not explain the low accuracy. Second, the tracker is the decisive component. Replacing MOSSE with ByteTrack improves the baseline score from 0.477 to 0.660 (YOLOv3) and from 0.527 to 0.688 (YOLO11), and it reduces false alarms by up to a factor of three. With the sudden-stop score, MOSSE tracks reach only 0.48–0.58, whereas ByteTrack and detection-only tracks reach 0.77–0.88; the drift and re-initialization jumps of MOSSE described in Section 3.6 produce false stops and speed peaks that no score definition can remove. Third, the newer detector does not improve the result once a robust tracker is used (YOLO11 + ByteTrack: 0.773 versus YOLOv3 + ByteTrack: 0.875 with the sudden-stop score), so YOLOv3 detections are sufficient for this task. Fourth, with ByteTrack or detection-only tracks, the sudden-stop score is clearly better than the baseline score (0.77–0.88 versus 0.61–0.69), whereas with MOSSE tracks both scores remain close to chance. Both the tracker and the score definition therefore had to be changed.

Table 6. Ablation on the TAD benchmark: AUC of the baseline motion-change score (test set) and of the sudden-stop score (development and test sets). FPS: end-to-end throughput, including decoding, detection, and tracking, on an A100 GPU (batch size one).

			Baseline Score	**Sudden-Stop Score**			
Detector	**Tracker**	N_d	**AUC Test**	**AUC Dev.**	**AUC Test**	**FA/h Test** †	**FPS**
YOLOv3	MOSSE	1	0.480	0.691	0.523	185	42.5
YOLOv3	MOSSE	5	0.477	0.681	0.582	124	78.1
YOLOv3	MOSSE	10	0.418	–	–	–	92.5
YOLOv3	MOSSE	20	0.469	–	–	–	103.6
YOLOv3	detection-only	1	0.605	0.810	0.824	93	73.0
YOLOv3	ByteTrack	1	0.660	**0.828**	**0.875**	124	66.8
YOLO11m	MOSSE	5	0.527	0.716	0.477	185	71.3
YOLO11m	ByteTrack	1	0.688	0.804	0.773	93	54.6

† Sudden-stop score at the threshold fixed on the development set (70th percentile of the scores of normal development videos). Bold values indicate the highest AUC of the sudden-stop score on the development and test sets.

5.4. Threshold Sensitivity

Figure 5 shows how the operating point depends on the threshold. For the baseline score (Figure 5a), increasing τ_S from 5 to 30 pixels lowers the recall from 1.00 to 0.75 but never raises the specificity above 0.19: normal videos produce motion-change scores as large as accident videos, so no pixel threshold separates the two classes. For the sudden-stop score (Figure 5b), recall and specificity trade off smoothly, and a broad range of thresholds, from approximately 0.10 to 0.55 box lengths per second, gives test F1 scores between 0.76 and 0.84. Table 7 lists the operating points obtained when the threshold is set to different percentiles of the scores of normal development videos. Because the score is expressed in box lengths per second, the same threshold applies to videos of different resolutions and frame rates. Stricter thresholds reduce false alarms to 16–34 events per hour on the development set, at the cost of recall.

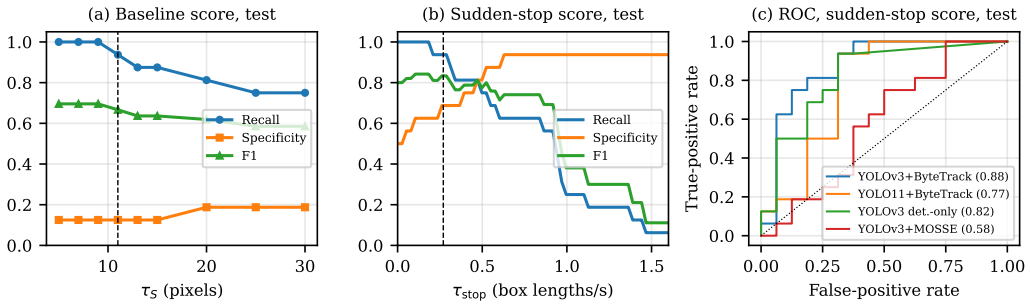


Figure 5. Threshold sensitivity and ranking performance on the TAD test set. (**a**) Baseline motion-change score for $\tau_S \in [5, 30]$ pixels (dashed line: original $\tau_S = 11$). (**b**) Sudden-stop score of the proposed configuration (dashed line: threshold fixed on the development set). (**c**) ROC curves of the sudden-stop score for different detector–tracker combinations (AUC in parentheses; dotted diagonal: chance level).

Table 7. Operating points of the proposed configuration (YOLOv3 + ByteTrack + sudden-stop score) for thresholds set at percentiles of the scores of normal development videos.

		Development Set				Test Set			
Percentile	τ_{stop}	**Recall**	**Spec.**	**F1**	**FA/h**	**Recall**	**Spec.**	**F1**	**FA/h**
60th	0.137	0.847	0.604	0.840	129	1.000	0.625	0.842	113
70th *	0.270	0.778	0.703	0.817	107	0.938	0.688	0.833	124
80th	0.336	0.739	0.802	0.811	85	0.812	0.688	0.765	124
90th	0.514	0.605	0.901	0.735	33	0.750	0.875	0.800	51
95th	0.652	0.521	0.946	0.675	16	0.625	0.938	0.741	31

* Operating point fixed in advance and used in Table 5.

5.5. Detection Latency

The latency analysis was limited by the structure of the benchmark. In 13 of the 16 accident test videos, the collision had already happened before the first frame, and only its aftermath (stationary or damaged vehicles, slowing traffic) was visible; an onset could be marked in three videos. Table 8 reports the latency for these three videos. For the proposed configuration, the sudden stop was detected 2.4–4.2 s after the first contact (median 4.0 s). Adding the 1.2 s needed to confirm that the vehicle remains stationary ($T_g + T_a$), the alarm became available 3.6–5.4 s after the onset (median 5.2 s). No alarm was raised before the onset in any of the three videos. The delay reflects the physics of the event, not the processing time: after the first contact, the vehicles involved typically skid or roll for several seconds before they come to rest, and the score deliberately waits for this stop. In the 13 videos without a visible onset, the proposed configuration raised an alarm in 12, which shows that vehicles coming to rest after the collision, such as following traffic, also trigger the score. With only three marked onsets, the latency results have very limited statistical power: they show the order of magnitude of the delay and its physical cause, but they do not allow a reliable estimate of its distribution. In addition, each onset was marked by a single annotator, so its uncertainty of a few frames could not be quantified by inter-annotator agreement; this uncertainty is small, however, compared with the measured latencies. A benchmark with temporal annotations of accident onsets would be required for a reliable latency estimate.

Table 8. Detection latency on the three accident test videos with a visible collision onset. Detected stop: time from the marked first contact to the frame at which the sudden stop begins; alarm available: detected stop plus the confirmation interval $T_g + T_a = 1.2$ s. No early alarms occurred.

Configuration	Video T2	Video T5	Video T8	Median
Detected stop (s)				
YOLOv3 + ByteTrack (proposed)	4.2	2.4	4.0	4.0
YOLOv3 + detection-only	0.0	3.1	0.8	0.8
YOLO11m + ByteTrack	missed	2.5	5.4	–
Alarm available (s)				
YOLOv3 + ByteTrack (proposed)	5.4	3.6	5.2	5.2

5.6. Computational Cost

Table 9 reports the mean processing time per frame of each stage. For the proposed configuration, the five stages of the pipeline take the following times on the A100 GPU: video decoding 1.53 ms, vehicle detection 11.98 ms, tracking with ByteTrack 1.46 ms, motion analysis (computation of the sudden-stop score) 0.03 ms, and alarm generation, which only compares the score with the threshold and merges nearby alarms, a negligible time that was not measured separately. The total of 15.0 ms per frame corresponds to 67 frames per second, well above the typical 25 fps of surveillance video. Detection thus accounts for 80% of the computation, whereas tracking, motion analysis, and alarm generation together account for about 10%. Periodic detection with MOSSE reduces the detector cost by a factor of N_d but adds 7–10 ms per frame for the correlation-filter updates of all tracked vehicles.

These processing times must be distinguished from the inherent delay of the score. An alarm can be raised only after the window after the stop has elapsed, that is, $T_g + T_a = 1.2$ s after the stop begins (Section 3.8). This delay is a property of the method and does not depend on the hardware (Section 5.5).

For the end-to-end models, one forward pass over a 16-frame clip takes 3.4 ms (R3D-18) and 57.7 ms (MViTv2-S), excluding decoding and resizing. Their cost per video depends on how densely clips are sampled, and they provide a decision per clip or per video without identifying the vehicles involved. All times were measured on a data-center GPU; the expected performance on embedded hardware is discussed in Section 6.4.

Table 9. Mean processing time per frame (ms) on an NVIDIA A100 GPU with batch size one. Motion analysis (sudden-stop score, 0.03 ms) and alarm generation are not included; see the text.

Detector	Tracker	N_d	Decoding	Detection	Tracking	Total (FPS)
YOLOv3	MOSSE	5	1.53	2.44	8.83	12.81 (78.1)
YOLOv3	MOSSE	1	1.53	11.98	10.04	23.56 (42.5)
YOLOv3	detection-only	1	1.53	11.98	0.19	13.70 (73.0)
YOLOv3	ByteTrack	1	1.53	11.98	1.46	14.97 (66.8)
YOLO11m	ByteTrack	1	1.58	15.10	1.64	18.32 (54.6)

6. Discussion

6.1. Why the Original Motion-Change Rule Fails

The preliminary evaluation suggested that the baseline rule detects most accidents, and on the benchmark its recall was indeed high (0.938). The benchmark shows, however, that this recall was obtained by raising alarms in almost every video: normal traffic produced motion-change scores as large as accidents, and no pixel threshold separated the two classes (Figure 5a). Three mechanisms explain this result. First, the score compares two windows that differ by a single frame, so it measures the frame-to-frame variation of the tracked box rather than a sustained change in vehicle motion; every detector refresh, box

re-initialization, or small localization error produces a score peak. Second, MOSSE has no motion model and no matching criterion, so its tracks can drift to the background or jump between neighboring vehicles, which creates large apparent displacements in dense traffic. The ablation confirms that the tracker is the dominant factor: keeping the same score but replacing MOSSE with ByteTrack or with gated detection-only association raised the AUC from 0.48 to 0.61–0.69, whereas neither more frequent detection ($N_d = 1$) nor a newer detector changed the result as long as MOSSE was used. Third, the score is measured in pixels, so it grows with image resolution and proximity to the camera. On TAD, this effect coincides with a property of the data—the normal development videos mostly have a high resolution—and would have biased any threshold learned from pixel scores. Normalizing by frame rate and box size removed the dependence on units but not the jitter, and therefore did not help. The preliminary evaluation, which contained only accidents, could not reveal any of these problems; this shows why normal videos are indispensable for evaluating accident detectors.

6.2. What the Sudden-Stop Score Captures

The sudden-stop score encodes a simple physical regularity of collisions observed by surveillance cameras: at least one vehicle involved decelerates from normal traffic speed to standstill within a fraction of a second and stays there. Three design choices make this cue measurable from imperfect tracks. Speed is expressed in vehicle lengths per second, which makes the score independent of image resolution and largely independent of the distance to the camera; accordingly, the development AUC on the resolution-matched subset (0.840) is as high as on the full development set (0.828). Medians over the windows before and after the stop suppress isolated box jitter. Finally, the requirement that the vehicle remain stationary for 1 s rejects short slowdowns. On the development set, 52% of the normal videos contain no sudden stop at all and receive a score of zero, compared with only 12% of the accident videos.

It is important to note what the score does and does not measure. It does not classify a stop as an accident; it reports an abrupt stop of a tracked vehicle as a *candidate* event. Its behavior in a given scenario therefore follows directly from its definition: it fires only if a tracked vehicle moves at a noticeable speed and then comes to rest within the clip. Table 10 summarizes the resulting behavior in typical scenarios with and without a collision.

Two consequences follow. First, collisions that do not end in a standstill, in particular side impacts after which the vehicles keep moving, minor scrapes, and hit-and-run incidents, are missed by design. At the selected operating point, 22% of the development accident videos were missed. Second, any abrupt stop without a collision, such as emergency braking or a stop caused by an obstacle, produces a false alarm, because the score observes only the motion of individual vehicles and cannot see whether contact has occurred; five of the 16 normal test videos exceeded the threshold. Distinguishing these cases requires cues beyond the speed of a single vehicle, such as the simultaneous stop of two nearby vehicles, the overlap of their boxes at the moment of the stop, a stationary vehicle remaining in a moving lane, or verification of the clip by an end-to-end classifier (Section 6.4). When a clip begins after the collision, the score depends on secondary stops of following vehicles; this was sufficient in 12 of the 13 such test videos but cannot be guaranteed in general. Finally, because the score relies on the continuity of individual tracks, it inherits the remaining weaknesses of the tracker (Section 6.5, item (iii)).

Table 10. Expected behavior of the sudden-stop score in typical traffic scenarios. TP: correct alarm; FN: missed accident; FP: false alarm.

Scenario	Collision	Behavior of the Score	Outcome
Collision after which at least one vehicle comes to rest (e.g., rear-end or frontal)	yes	High speed before the stop, then standstill	TP
Vehicle crashes into a stationary queue	yes	The striking vehicle stops abruptly	TP
Clip starts after the collision	yes	Depends on secondary stops of following vehicles	TP or FN
Side impact or sideswipe after which the vehicles keep moving	yes	No standstill	FN
Minor scrape or low-speed contact in slow traffic	yes	Speed before the contact below the threshold	FN
Hit-and-run; vehicles drive away after the impact	yes	No standstill	FN
Collision with a pedestrian or cyclist without an abrupt vehicle stop	yes	Road user not tracked; vehicle decelerates gradually	FN
Stopped vehicle occluded or outside the field of view	yes	Track ends before the stationarity test	FN
Emergency braking to a standstill without a collision	no	Indistinguishable from a stop after a collision	FP
Stop caused by an obstacle (debris, animal, broken-down vehicle, crossing pedestrian)	no	Abrupt stop of a vehicle not involved in a collision	FP
Tail of a queue forming at a signal or in congestion	no	Usually gradual (low score); abrupt stops possible	FP (occasional)
Track that ends at the image border or jumps to another vehicle	no	Apparent stop caused by tracking	FP

6.3. Comparison with End-to-End Models

The end-to-end models learn appearance cues that a trajectory rule cannot use, such as deformation, debris, overturned vehicles, and pedestrians on the road. This explains why MViTv2-S reached the highest specificity (0.812). The proposed modular configuration reached a similar ranking performance (AUC 0.875 versus 0.844) without any training on labeled accident videos, whereas both end-to-end models were fine-tuned on the labeled development videos. The differences in AUC, recall, and F1 score lie well within the confidence intervals of a 32-video test set and should not be read as an advantage of either approach. The two approaches differ mainly in what they require and what they provide. The modular configuration needs no accident videos for training, and its decision threshold is a single value in physically interpretable units. It identifies the vehicle and the frame that triggered each alarm, runs at 67 fps including detection, and allows an operator to check the evidence directly. The end-to-end models require labeled accident videos and GPU training, provide one decision per clip or video, and cannot indicate which vehicle is involved. The two approaches are therefore complementary: the sudden-stop score can serve as an inexpensive, always-on trigger with high recall, and an end-to-end classifier can then verify only the short clips around triggered events to reduce false alarms.

6.4. False Alarms and Practical Deployment

The latency measured on the three test videos with a visible onset (alarm available 3.6–5.4 s after the first contact) is dominated by the time that vehicles need to come to rest, not by computation. This delay is inherent to the design: the score detects a consequence of

the collision, namely the vehicles coming to rest, rather than the impact itself. Cues such as box overlap or abrupt changes of heading would be needed to react at the moment of contact.

Even after the revision, the false-alarm rate at the default operating point (107 and 124 alarm events per hour of normal video on the development and test sets, respectively) is far too high for unattended alerting. Two factors put this figure into perspective. The normal videos of TAD are short clips of busy road scenes, which probably contain more stops per hour than typical continuous surveillance streams, and the test estimate is based on only 5.8 min of normal video. Nevertheless, the result shows that the rule is suitable only as a pre-screening mechanism for human operators or as the first stage of a two-stage system, not for fully unattended alerting. This conclusion does not change with parameter tuning: even the strictest operating point in Table 7 still produces 16 alarm events per hour on the development set while missing about half of the development accidents (recall 0.52), and the failure scope described in Section 6.2 cannot be removed by any choice of threshold. False alarms could be reduced further by calibrating the threshold for each camera, restricting the analysis to lane regions, requiring corroborating cues such as the stop of several vehicles or a stationary vehicle in a moving lane, and verifying alarms with an end-to-end model.

All runtimes in this study were measured on an NVIDIA A100 data-center GPU (Table 9), which is far more powerful than the low-power embedded hardware installed in cameras or roadside cabinets. On such hardware, the throughput will be considerably lower, and the slowdown will be dominated by the detector, which accounts for 12.0 of the 15.0 ms per frame (80%) in the proposed configuration. ByteTrack (1.5 ms) and the sudden-stop score (0.03 ms) operate only on box coordinates and are inexpensive even on an embedded CPU. Whether the pipeline sustains 25 fps on a specific embedded platform was not measured in this study. The detector cost can be reduced by compiling the network with a dedicated inference engine at reduced numerical precision (FP16 or INT8), by lowering the input resolution, or by using a smaller detector, each of which may reduce detection quality and must therefore be re-validated. Processing a stream at a lower frame rate would also reduce the cost, but the effect of lower frame rates on ByteTrack tracks and on the sudden-stop score was not evaluated here. Alternatively, the video streams can be analyzed centrally: at 67 frames per second, the unbatched implementation can process approximately two 25-fps streams on one A100 GPU, and processing several streams in one batch is expected to increase this number.

6.5. Limitations

Several limitations should be considered. (i) The test set of TAD contains only 32 videos (16 per class), so all test metrics have wide confidence intervals; in particular, the difference between the proposed configuration and MViTv2-S is statistically inconclusive, and the test false-alarm rate is based on only 5.8 min of normal video. In addition, the development set is biased by image resolution. Validation on other datasets is therefore needed to confirm that the results generalize, for example on the road-accident and normal videos of UCF-Crime [24], which would allow specificity to be estimated on independent surveillance footage, and on the CCTV accident videos of CADP [29], which would test recall and, where temporal annotations are available, latency on a larger and more diverse set of accidents. (ii) TAD provides only one label per video; accident onsets for the latency analysis were marked manually by a single annotator in only three test videos, so the latency results have limited statistical power. (iii) The framework depends strongly on the quality of multi-object tracking. Because the score is computed on individual tracks, track fragmentation, identity switches, and long-term occlusion directly degrade detection:

a fragmented track can hide the stop of a vehicle, and an identity switch or a track that jumps between vehicles can create a false stop. The ablation quantifies this dependence: replacing ByteTrack with MOSSE reduced the test AUC of the sudden-stop score from 0.875 to 0.582 (Table 6), and performance can be expected to decrease further in scenes where even ByteTrack fragments tracks, such as dense traffic, strong occlusion, or low illumination. (iv) The benchmark experiments use the Ultralytics port of YOLOv3, whose detection head differs from the original TensorFlow implementation used in the preliminary experiments; the preliminary and benchmark results are therefore not directly comparable. (v) The parameters of the sudden-stop score and all thresholds were selected on the development set, but the design of the score was guided by an exploratory analysis of the same set, so its test performance may be optimistic. (vi) The end-to-end models were trained with standard settings on a small training set and were not tuned extensively; stronger end-to-end models are likely to perform better. (vii) All motion quantities are measured in the image; without camera calibration, the system does not estimate metric speed or acceleration, and the effects of perspective and vehicle orientation on the score were not quantified. (viii) Runtimes were measured on a data-center GPU; the throughput on embedded hardware will be lower and was not measured (Section 6.4).

7. Conclusions

This study evaluated an interpretable, modular pipeline for detecting candidate traffic accidents in surveillance video. In its original form, the pipeline detected vehicles every five frames with YOLOv3, tracked them with MOSSE, and raised an alarm when a pixel-based motion-change score exceeded 11 pixels. It detected 60% of the accidents in a corpus that contained only accident videos. This figure, however, proved misleading: on the TAD benchmark, which also contains normal traffic, the pipeline raised alarms in 14 of the 16 normal videos, and its score did not separate accident videos from normal videos (test AUC 0.48). Controlled experiments identified two causes of this failure. First, the MOSSE tracks drifted away from the vehicles. Second, the score measured small frame-to-frame changes of the bounding box in pixels rather than real changes in vehicle motion. Running the detector more often made the pipeline slower but not more accurate, and a newer detector did not improve the results.

A proposed configuration that keeps the modular structure, links detections with ByteTrack, and scores each vehicle by the speed at which it comes to a sudden stop, relative to its apparent size, reached a test AUC of 0.88 and an F1 score of 0.83 with a threshold fixed on the development set. Its ranking performance was statistically indistinguishable from that of fine-tuned R3D-18 and MViTv2-S classifiers, although it required no training on labeled accident videos. It also identified the vehicle and frame responsible for each alarm and processed 67 frames per second on a GPU. The score does not depend on image resolution, which proved important because the development data of the benchmark were biased by resolution.

The false-alarm rate of the proposed configuration remains too high for unattended operation, even after threshold tuning, and the method cannot detect accidents in which no vehicle comes to rest. It is therefore intended for pre-screening by human operators, not for fully automatic alerting. Because the test set is small, future work will evaluate the method on other datasets, in particular on the road-accident and normal videos of UCF-Crime and on CADP; combine the sudden-stop trigger with an end-to-end verification model in a two-stage system; measure its throughput on embedded hardware; add complementary cues such as interactions between vehicles and stationary vehicles in moving lanes; and use camera calibration to estimate metric speed and deceleration. Overall, the results show that a simple, physically motivated rule on top of robust tracking can serve as a transparent

and efficient first stage for accident screening in traffic surveillance, provided that it is evaluated on normal traffic as well as on accidents.

Author Contributions: Conceptualization, Y.D. and A.S.; methodology, Y.D.; software, Y.D.; validation, Y.D. and A.S.; formal analysis, Y.D.; investigation, Y.D.; resources, A.S.; data curation, Y.D.; writing—original draft preparation, Y.D.; writing—review and editing, Y.D. and A.S.; visualization, Y.D.; supervision, A.S.; project administration, A.S. All authors have read and agreed to the published version of the manuscript.

Funding: No external funding was received for this research.

Data Availability Statement: The TAD benchmark used in this study is publicly available at https://github.com/UnicomAI/UnicomBenchmark/tree/main/TADBench, accessed on 1 September 2026.

Acknowledgments: During the preparation of this manuscript, the authors used Antrophic Claude Opus 5.5 for language editing and for assistance with code development and data analysis. The authors have reviewed and edited all outputs and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. World Health Organization. Road Traffic Injuries. 2026. Available online: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries (accessed on 14 September 2026).
2. Fang, J.; Qiao, J.; Xue, J.; Li, Z. Vision-Based Traffic Accident Detection and Anticipation: A Survey. *IEEE Trans. Circuits Syst. Video Technol.* **2024**, *34*, 1983–1999. https://doi.org/10.1109/TCSVT.2023.3307655.
3. Robles-Serrano, S.; Sanchez-Torres, G.; Branch-Bedoya, J. Automatic Detection of Traffic Accidents from Video Using Deep Learning Techniques. *Computers* **2021**, *10*, 148. https://doi.org/10.3390/computers10110148.
4. Htun, S.S.; Park, J.S.; Lee, K.W.; Han, J.H. TempoLearn Network: Leveraging Spatio-Temporal Learning for Traffic Accident Detection. *IEEE Access* **2023**, *11*, 142292–142303. https://doi.org/10.1109/ACCESS.2023.3343410.
5. Adewopo, V.A.; Elsayed, N. Smart City Transportation: Deep Learning Ensemble Approach for Traffic Accident Detection. *IEEE Access* **2024**, *12*, 59134–59147. https://doi.org/10.1109/ACCESS.2024.3387972.
6. Yao, Y.; Wang, X.; Xu, M.; Pu, Z.; Wang, Y.; Atkins, E.; Crandall, D.J. DoTA: Unsupervised Detection of Traffic Anomaly in Driving Videos. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 444–459. https://doi.org/10.1109/TPAMI.2022.3150763.
7. Ghahremannezhad, H.; Shi, H.; Liu, C. Real-Time Accident Detection in Traffic Surveillance Using Deep Learning. In Proceedings of the 2022 IEEE International Conference on Imaging Systems and Techniques (IST), Kaohsiung, Taiwan, 21–23 June 2022; pp. 1–6. https://doi.org/10.1109/IST55454.2022.9827736.
8. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.
9. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550. https://doi.org/10.1109/CVPR.2010.5539960.
10. Zhang, Y.; Sun, P.; Jiang, Y.; Yu, D.; Weng, F.; Yuan, Z.; Luo, P.; Liu, W.; Wang, X. ByteTrack: Multi-object Tracking by Associating Every Detection Box. In *Proceedings of the Computer Vision—ECCV 2022*; Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T., Eds.; Springer: Cham, Switzerland, 2022; pp. 1–21. https://doi.org/10.1007/978-3-031-20047-2_1.
11. Revaud, J.; Humenberger, M. Robust Automatic Monocular Vehicle Speed Estimation for Traffic Surveillance. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021; pp. 4531–4541. https://doi.org/10.1109/ICCV48922.2021.00451.
12. Xu, Y.; Hu, H.; Huang, C.; Nan, Y.; Liu, Y.; Wang, K.; Liu, Z.; Lian, S. TAD: A Large-Scale Benchmark for Traffic Accidents Detection From Video Surveillance. *IEEE Access* **2025**, *13*, 2018–2033. https://doi.org/10.1109/ACCESS.2024.3522384.
13. Ibrahim, O.; ElGendy, H.; ElShafee, A.M. Speed Detection Camera System using Image Processing Techniques on Video Streams. *Int. J. Comput. Electr. Eng.* **2011**, *3*, 771–778. https://doi.org/10.7763/IJCEE.2011.V3.418.
14. Xia, S.; Xiong, J.; Liu, Y.; Li, G. Vision-based traffic accident detection using matrix approximation. In Proceedings of the 2015 10th Asian Control Conference (ASCC), Kota Kinabalu, Malaysia, 31 May–3 June 2015; pp. 1–5. https://doi.org/10.1109/ASCC.2015.7244586.

15. Wang, C.; Dai, Y.; Zhou, W.; Geng, Y. A Vision-Based Video Crash Detection Framework for Mixed Traffic Flow Environment Considering Low-Visibility Condition. *J. Adv. Transp.* **2020**, *2020*, 1–11. https://doi.org/10.1155/2020/9194028.
16. Yang, D.; Wu, Y.; Sun, F.; Chen, J.; Zhai, D.; Fu, C. Freeway accident detection and classification based on the multi-vehicle trajectory data and deep learning model. *Transp. Res. Part C Emerg. Technol.* **2021**, *130*, 103303. https://doi.org/10.1016/j.trc.2021.103303.
17. Yu, R.; Wang, Y.; Zou, Z.; Wang, L. Convolutional neural networks with refined loss functions for the real-time crash risk analysis. *Transp. Res. Part C Emerg. Technol.* **2020**, *119*, 102740. https://doi.org/10.1016/j.trc.2020.102740.
18. Zheng, L.; Sayed, T. A novel approach for real time crash prediction at signalized intersections. *Transp. Res. Part C Emerg. Technol.* **2020**, *117*, 102683. https://doi.org/10.1016/j.trc.2020.102683.
19. Lu, Z.; Zhou, W.; Zhang, S.; Wang, C. A New Video-Based Crash Detection Method: Balancing Speed and Accuracy Using a Feature Fusion Deep Learning Framework. *J. Adv. Transp.* **2020**, *2020*, 1–12. https://doi.org/10.1155/2020/8848874.
20. Tran, D.; Wang, H.; Torresani, L.; Ray, J.; LeCun, Y.; Paluri, M. A Closer Look at Spatiotemporal Convolutions for Action Recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 6450–6459. https://doi.org/10.1109/CVPR.2018.00675.
21. Feichtenhofer, C.; Fan, H.; Malik, J.; He, K. SlowFast Networks for Video Recognition. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October 2019–2 November 2019; pp. 6202–6211. https://doi.org/10.1109/ICCV.2019.00630.
22. Li, Y.; Wu, C.Y.; Fan, H.; Mangalam, K.; Xiong, B.; Malik, J.; Feichtenhofer, C. MViTv2: Improved Multiscale Vision Transformers for Classification and Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 4804–4814. https://doi.org/10.1109/CVPR52688.2022.00476.
23. Kay, W.; Carreira, J.; Simonyan, K.; Zhang, B.; Hillier, C.; Vijayanarasimhan, S.; Viola, F.; Green, T.; Back, T.; Natsev, P.; et al. The Kinetics Human Action Video Dataset. *arXiv* **2017**, arXiv:1705.06950.
24. Sultani, W.; Chen, C.; Shah, M. Real-World Anomaly Detection in Surveillance Videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–23 June 2018; pp. 6479–6488. https://doi.org/10.1109/CVPR.2018.00678.
25. Jocher, G.; Qiu, J. Ultralytics YOLO11. Software, Version 11.0.0. 2024. Available online: https://github.com/ultralytics/ultralytics (accessed on 1 September 2026).
26. Bewley, A.; Ge, Z.; Ott, L.; Ramos, F.; Upcroft, B. Simple Online and Realtime Tracking. In Proceedings of the IEEE International Conference on Image Processing (ICIP), Phoenix, AZ, USA, 25–28 September 2016; pp. 3464–3468. https://doi.org/10.1109/ICIP.2016.7533003.
27. Kul, S.; Eken, S.; Sayar, A. Distributed and collaborative real-time vehicle detection and classification over the video streams. *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1729881417720782.
28. Şentaş, A.; Kul, S.; Sayar, A. Real-time traffic rules infringing determination over the video stream: Wrong way and clearway violation detection. In *Proceedings of the 2019 International Artificial Intelligence and Data Processing Symposium (IDAP)*; IEEE: Piscataway, NJ, USA, 2019; pp. 1–4.
29. Shah, A.; Lamare, J.B.; Nguyen-Anh, T.; Hauptmann, A. CADP: A Novel Dataset for CCTV Traffic Camera Based Accident Analysis. In Proceedings of the 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand, 27-30 November 2018; pp. 1–9. https://doi.org/10.1109/AVSS.2018.8639160.
30. General Directorate of Security, Traffic Department. KGYS Camera Images. 2021. Available online: https://www.trafik.gov.tr/ (accessed on 15 December 2021).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.