Beyond FPS: A Critical Review of Temporal Evidence for Real-Time Object Detection in Video Surveillance
http://www.doi.org/10.62341/istj-vol39-1-ri51

Received	**2026/08/26**	**تم استلام الورقة العلمية في**
Accepted	**2026/09/14**	**تم قبول الورقة العلمية في**
Published	**2026/09/15**	**تم نشر الورقة العلمية في**

Beyond FPS: A Critical Review of Temporal Evidence for Real-Time Object Detection in Video Surveillance

[1]Rafea Mohammed Almejrab*, [2]Mofeda Abdelrrazik Abdelwakil Madhi
College of Computer Technology - Benghazi, Libya
*Corresponding author: rafea.almejrab@cctben.edu.ly
Co-author email: fafymady807@gmail.com
1- https://orcid.org/0009-0008-0409-3191

Abstract
"Real-time" is one of the most frequently used performance claims in deep-learning video surveillance, yet the term is often supported by incompatible measures such as detector frames per second (FPS), isolated inference latency, edge-device throughput, or partially defined system timing. This targeted critical review examines how real-time performance is evidenced in ten empirical object-detection studies published between 2021 and 2026, selected purposively to maximize variation in detector architecture, computing platform, deployment setting, and temporal boundary. Google Scholar was used as the primary discovery source, supplemented by publisher verification and targeted citation chasing. A descriptive four-level temporal-evidence typology was applied: L0 (claim only), L1 (model-level timing), L2 (operational throughput or partial-pipeline timing), and L3 (explicit system-total or end-to-end timing). A separate structured reporting-reliability appraisal examined five dimensions: presence of an explicit temporal metric, hardware/context specification, timing-boundary clarity, timing-protocol reproducibility, and operational/live evaluation. Within this purposive sample, seven of ten studies (70.0%) reported FPS, five (50.0%) reported an explicit latency value, and three (30.0%) provided a clearly defined system-total or end-to-end temporal boundary. These proportions are descriptive of the selected sample

http://www.doi.org/10.62341/istj-vol39-1-ri51

and are not prevalence estimates for the field. Reported FPS ranged from 1.96 FPS on a Raspberry Pi 3B to 143.7 FPS on an RTX 3060 Laptop GPU, a greater than 73-fold span across heterogeneous hardware and workloads. Recent L3 studies further show that preprocessing, post-processing, decoding, buffering, networking, and rendering can dominate total response time even when detector inference is fast. The review uses the public scope of IEC 62676-6:2026 as a standards-informed lens for operational context, without claiming compliance assessment. The findings support a central conclusion: real-time performance is not an intrinsic property of an object detector; it is a property of the detector–hardware–workload–application configuration and of the temporal boundary being measured. A minimum reporting set and a future temporal–energy reporting extension are proposed to improve reproducibility and deployment relevance.
Keywords: real-time object detection; video surveillance; YOLO; latency; FPS; edge computing; deployment; reproducibility

ما وراء معدل الإطارات :مراجعة نقدية للأدلة الزمنية لكشف الأجسام في الزمن الحقيقي ضمن أنظمة المراقبة بالفيديو

رافع محمد المجراب*، مفيدة عبد الرازق عبد الوكيل ماضي
كلية تقنية الحاسوب – بنغازي، ليبيا
البريد الإلكتروني للمؤلف المسؤول:
rafea.almejrab@cctben.edu.ly

الملخص

يُعد وصف أنظمة كشف الأجسام في المراقبة بالفيديو بأنها تعمل» في الزمن الحقيقي « من أكثر ادعاءات الأداء شيوعاً، إلا أن هذا الوصف يُدعَم في كثير من الدراسات بمقاييس غير متجانسة، مثل معدل الإطارات في الثانية، أو زمن الاستدلال المنفصل، أو إنتاجية أجهزة الحافة، أو توقيت جزئي لمسار النظام .تقدم هذه الدراسة مراجعة نقدية موجهة لعشر دراسات تجريبية في كشف الأجسام نُشرت بين عامي 2021 و2026،

واختيرت بصورة قصدية لتحقيق تنوع في بنية الكاشف، ومنصة الحوسبة، وبيئة التشغيل، وحدود القياس الزمني .استُخدم Google Scholar كمصدر اكتشاف رئيسي مع التحقق من صفحات الناشرين وتتبع الاستشهادات بصورة موجهة .وطُبّق تصنيف وصفي رباعي المستويات للأدلة الزمنية L0 :للادعاء دون قياس زمني، و L1لتوقيت النموذج، و L2 للإنتاجية التشغيلية أو التوقيت الجزئي لمسار المعالجة، و L3للتوقيت الكلي للنظام أو من البداية إلى النهاية .كما أُجري تقييم وصفي منظم لموثوقية الإبلاغ عبر خمسة أبعاد : وجود مقياس زمني صريح، وتحديد العتاد والسياق، ووضوح حدود القياس الزمني، وقابلية إعادة إنتاج بروتوكول التوقيت، ووجود تقييم تشغيلي أو بث حي .ضمن هذه العينة القصدية، أبلغت سبع دراسات من أصل عشر (70.0%) عن معدل الإطارات، وأبلغت خمس (50.0%) عن قيمة صريحة للكمون، بينما قدمت ثلاث دراسات (30.0%) حدوداً زمنية واضحة للنظام ككل أو من البداية إلى النهاية .وتُعد هذه النسب وصفية للعينة المختارة وليست تقديرات لانتشار الممارسات في المجال .تراوح الأداء المبلغ عنه من 1.96 إطار/ثانية على Raspberry Pi 3B إلى 143.7 إطار/ثانية على RTX 3060 Laptop GPU، أي بفارق يتجاوز 73 ضعفاً عبر عتاد وأعباء عمل غير متجانسة .كما تُظهر دراسات المستوى L3 الحديثة أن فك الفيديو، والمعالجة المسبقة واللاحقة، والتخزين المؤقت، والشبكات، والعرض قد تهيمن على زمن الاستجابة الكلي حتى عندما يكون استدلال الكاشف سريعاً .استخدمت الدراسة النطاق العام لمعيار IEC 62676-6:2026كعدسة تفسيرية مستنيرة بالمعايير دون ادعاء تقييم المطابقة .وتدعم النتائج خلاصة مركزية مفادها أن الأداء في الزمن الحقيقي ليس خاصية كامنة في كاشف الأجسام، بل خاصية لتركيبة الكاشف والعتاد وعبء العمل والتطبيق وحدود القياس الزمني .وتقترح الدراسة مجموعة دنيا من عناصر الإبلاغ، مع امتداد مستقبلي يربط الأدلة الزمنية باستهلاك الطاقة لتحسين القابلية للتكرار وملاءمة النشر التشغيلي.

الكلمات المفتاحية: كشف الأجسام في الزمن الحقيقي؛ المراقبة بالفيديو؛ YOLO؛ الكمون؛ معدل الإطارات؛ الحوسبة الطرفية؛ النشر التشغيلي؛ قابلية التكرار.

http://www.doi.org/10.62341/istj-vol39-1-ri51

1. Introduction

Deep-learning object detectors have become central components of intelligent video surveillance, supporting applications such as intrusion detection, occupancy monitoring, person detection, vehicle monitoring, anomaly analysis, and automated alert generation. The YOLO family in particular has been widely adopted because one-stage detection can provide a favorable speed–accuracy trade-off for video analytics [1]. Broader reviews of the YOLO family and modern object-detection literature confirm that reported speed is closely tied to architecture, implementation choices, hardware, and evaluation protocol rather than to model name alone [2], [3]. Recent surveillance reviews likewise emphasize the diversity of operating environments, embedded platforms, and deployment constraints that shape practical performance [4], [5]–[7].

Despite this progress, the phrase "real-time" is frequently used without a consistent measurement boundary. A study may label a detector real-time because a GPU benchmark exceeds a chosen FPS value, because a forward pass takes only a few milliseconds, because an edge device processes a live stream, or because a complete application appears responsive to a user. These are not equivalent forms of evidence. In a deployed surveillance system, the observable response time may include camera acquisition, decoding, resizing, normalization, detector inference, non-maximum suppression, tracking or behavior analysis, network transmission, rendering, database operations, and alert generation. Measuring only one stage can substantially underestimate system-level delay.

The problem is amplified by hardware heterogeneity. The same or similar detector family may run at very different rates on a high-end GPU, an embedded NVIDIA platform, or a low-cost single-board computer. Input resolution, precision, batch size, number of simultaneous streams, and operational load can also alter throughput. Reviews focused on edge computing and smart-city surveillance identify compute capacity, memory, communication cost, and device placement as core constraints for time-critical video analytics [6], [7]. Consequently, a detector cannot be described

http://www.doi.org/10.62341/istj-vol39-1-ri51

meaningfully as real-time without specifying both the computing context and the application requirement.

This issue has become especially important with the publication of IEC 62676-6:2026, which addresses performance testing and grading of real-time intelligent video content analysis devices and systems used in video surveillance. Its public scope emphasizes functions, performance, interfaces, environmental adaptability, test methods, evaluation, and operation under stresses such as obscuration, extreme weather, vibration, and image degradation [8]. The present paper does not perform a compliance audit and does not rely on the paid full standard; instead, the public scope is used only as a standards-informed reminder that surveillance performance must be interpreted at system and operating-environment level.

The objective of this paper is therefore not to identify the "best" detector. Rather, it critically examines what selected surveillance studies actually measure when they claim real-time performance. The scientific novelty is not the general observation that FPS alone can be insufficient; that limitation is already recognized in the literature. The contribution is to operationalize the problem through a four-level L0–L3 temporal-evidence typology, combine that typology with a practical minimum reporting set, and interpret both through hardware, workload, deployment, and standards-informed context. Four research questions guide the review: (RQ1) How is real-time performance operationalized? (RQ2) What timing boundaries are measured—model inference, throughput, partial pipeline, or system total? (RQ3) How do hardware and application context affect interpretation? and (RQ4) What minimum information should be reported to make real-time claims reproducible and deployment-relevant?

2. Background and Related Work

2.1 Real-time object detection in surveillance

The wider literature provides important context for interpreting real-time claims. Terven et al. [2] trace the evolution of YOLO architectures and emphasize the recurring speed–accuracy trade-off across generations, while Edozie et al. [3] compare one-stage, two-stage, and Transformer-based detectors in terms of accuracy, computational speed, and deployment readiness. At the application level, Bello et al. [9] review real-time computer vision in smart infrastructure, and Olansile et al. [5] synthesize secure imaging and video-surveillance research with attention to real-world implementations. Edge-oriented reviews further show that surveillance performance depends on embedded hardware, data movement, network topology, and workload placement [6], [7]. Benchmark-focused surveillance literature also documents the high computational cost and hardware sensitivity of modern deep-learning video analytics [10]. These reviews motivate the present paper's narrower question: not whether fast detectors exist, but whether the evidence reported for a particular surveillance system is sufficient to justify the label real-time.

Nimma et al. [11] proposed an attention- and Transformer-enhanced YOLOv8 architecture for real-time video surveillance. The paper reports 96.78% precision, 96.89% recall, 89.67% mAP, and an inference time of 5.2 ms per frame. However, the authors state that the timing was obtained on "high-end hardware" and note that edge deployment may introduce additional latency. The value therefore represents strong model-level timing evidence but not a fully defined camera-to-output system latency.

Chen et al. [12] examined a distributed edge–cloud surveillance architecture in which three NVIDIA Jetson Xavier NX edge devices communicated with a cloud system using an RTX 2080 Ti GPU. YOLOv3 achieved 15.7 FPS on the edge platform, and the proposed communication mechanism reduced backbone-network bandwidth occupancy by approximately 94%. This is more deployment-oriented than isolated model benchmarking, but the study does not collapse acquisition, preprocessing, inference, communication, and output into one explicit end-to-end latency value.

Al-E'mari et al. [13] augmented YOLOv8 with anomaly-detection and behavior-analysis components and reported an average processing rate of 45 FPS in high-resolution surveillance analysis. The study includes live/field-oriented evaluation and therefore provides operational throughput evidence. Yet the reported FPS is not decomposed across preprocessing, detection, behavior analysis, post-processing, networking, and alert delivery.

http://www.doi.org/10.62341/istj-vol39-1-ri51

Azatbekuly et al. [14] developed a web-based intelligent surveillance system using IP cameras, RTSP, OpenCV, YOLOv8n, event triggers, databases, and Telegram notifications. Their implemented office environment used an NVIDIA GeForce GTX 1660 SUPER and a 504×706 RGB video stream. A model-comparison table reports 34 FPS and 2.2 ms latency for YOLOv8n. If both values represented the same serialized, batch-1 timing boundary, 2.2 ms would correspond to roughly 455 FPS, not 34 FPS. The paper does not explain the different timing scopes or protocols, making the deployment-level meaning of the two figures ambiguous.

Oguine et al. [15] presented a YOLOv3-based Smart Surveillance System using MS COCO and collected surveillance footage, reporting 99.71% detection accuracy and an mAP of 61.5. Although the work repeatedly frames the system as real-time and documents a GPU-capable development environment, the evaluated results focus on detection accuracy rather than an explicit measured FPS or latency for the proposed system. This makes it a useful example of a claim that is stronger than its temporal evidence.

Nguyen et al. [16] specifically targeted constrained edge hardware. Their YOLO-derived human detector ran on a Raspberry Pi 3B (ARM v7, 1.2 GHz, 1 GB RAM) and achieved 1.96 FPS for the selected 320×320 configuration. Importantly, the authors explicitly treated approximately 2 FPS as an acceptable design target for their surveillance task. This study demonstrates that a task-specific real-time requirement can be substantially lower than conventional video frame rates when the application does not require detection on every captured frame.

Zhao et al. [17] proposed CMCA-YOLO for parking-lot surveillance and evaluated it on a proprietary dataset of 4502 images. Using an RTX 3060 Laptop GPU, an Intel Core i9-12900H CPU, and 640×640 inputs, the model achieved 143.7 FPS with mAP@0.5 of 0.895. The paper provides detailed hardware and model configuration, but edge integration is identified as future work. The reported throughput therefore supports high model speed on the tested platform, not equivalent performance on resource-constrained surveillance hardware.

http://www.doi.org/10.62341/istj-vol39-1-ri51

A recent 2026 study by Saleem and Jerith [18] provides a comparatively stronger example of temporal reporting. Their SurveilSmartAI/ObjTrackNet system reports 62 FPS on an RTX 4090 cloud server with a 15.2 ms total per-frame latency, decomposed into 4.3 ms preprocessing, 8.6 ms inference, and 2.3 ms post-processing. On a Jetson Xavier NX, the corresponding total was 35.4 ms (9.7 + 19.1 + 6.6 ms), and the study further reports simultaneous-stream limits. This type of explicit boundary definition is closer to system-level real-time evidence than detector FPS alone.

Fierro Silva et al. [19] provide a second L3 example in real-time weapon detection for video surveillance. Their Jetson Nano deployment explicitly measures preprocessing, inference, post-processing, total end-to-end latency, and FPS under a fixed 640×640 live-video configuration. The study shows why architectural details after inference matter: YOLOv8s, YOLOv9s, and YOLOv11s exhibit total latencies of roughly 296–315 ms because post-processing alone consumes about 175–184 ms, whereas YOLOv10s and YOLOv26s reduce post-processing to about 2 ms and achieve total latencies around 124–131 ms. Thus, models with similar inference times can differ substantially in deployment-level responsiveness.

Castro-Castaño et al. [20] evaluate a low-latency autonomous surveillance architecture that integrates RTSP ingestion, WHEP/WebRTC distribution, YOLOv11 analytics, and browser visualization. A timestamp-based experiment measured approximately 267 ms end-to-end latency for the WebRTC baseline and about 1039 ms with AI analytics enabled, implying roughly 772 ms of additional pipeline overhead. The authors emphasize that this overhead includes frame extraction, CPU decoding, buffering, and metadata synchronization rather than detector inference alone. More than 300 hours of operational testing further provide evidence of sustained system behavior under realistic network conditions.

2.2 The temporal-boundary problem

The studies above show that the phrase real-time can refer to at least four distinct evidence types: a verbal claim with no timing metric; a detector-only metric such as FPS or inference

http://www.doi.org/10.62341/istj-vol39-1-ri51

milliseconds; operational throughput on a live or edge platform without a fully defined total boundary; or explicit system-total/end-to-end timing. Conflating these evidence types creates two risks. First, model benchmarks may be interpreted as deployment performance. Second, cross-study comparisons may rank algorithms that were tested under different hardware, resolutions, workloads, and timing boundaries. The present review therefore separates what was timed from where and under what load it was timed.

The importance of the timing boundary is also visible in foundational detector literature. Faster R-CNN reported approximately 5 FPS while explicitly stating that the figure included all processing steps [21], whereas SSD was designed to remove the separate proposal stage and provide a unified single-network detector [22]. DETR later simplified another part of the conventional detection pipeline by formulating detection as direct set prediction and removing hand-designed components such as non-maximum suppression and anchor generation [23]. These architectural differences illustrate why a raw FPS or millisecond value is meaningful only when the measured pipeline boundary is stated explicitly.

2.3 Operational definition of real-time and timing terms

For this review, "real-time" is treated as an application-relative deadline rather than a universal frame-rate threshold. Let D_app denote the maximum acceptable response time for the surveillance task. For a declared end-to-end boundary, the total response can be represented as $T\_E2E = T\_acq + T\_decode + T\_pre + T\_inf + T\_post + T\_net + T\_output$, where the terms denote acquisition, decoding/frame grabbing, preprocessing, detector inference, post-processing/analytics, network transfer, and rendering/alert output. A latency-based real-time claim is supported only when the measured T_E2E is less than or equal to D_app under the stated operating condition and reported statistic (for example, median or P95). For continuous-stream applications, sustained processing throughput should additionally satisfy $R\_proc \geq R\_req$, where R_req is the application-required frame-processing rate.

http://www.doi.org/10.62341/istj-vol39-1-ri51

The temporal terms used in this paper are therefore kept distinct. FPS is a throughput measure, FPS = N/Δt, describing how many frames are processed per unit time. Inference time is the duration of the neural-network forward pass for a frame or batch. Latency is the elapsed time between any explicitly defined start and end events and is meaningful only when that boundary is stated. End-to-end latency refers to the complete declared application path from an operational input event, such as frame acquisition, to the required output or decision, such as display, storage, or alert generation. These quantities may be related, but they are not interchangeable.

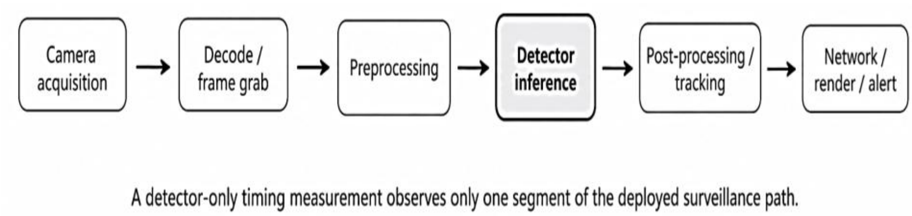

Figure 1. Temporal boundary of a deployed video-surveillance pipeline. Detector inference represents only one component of total system response time.

3. Materials and Methods

3.1 Review design and search strategy

This study was designed as a targeted critical review and methodological evidence audit, not as a systematic literature review, systematic review, or meta-analysis. The purpose was to interrogate the validity and interpretability of real-time claims across contrasting surveillance studies rather than to estimate prevalence across the entire literature. Google Scholar was used as the primary discovery source with the query: “real-time object detection” “video surveillance” YOLO, restricted to publications from 2020–2026. Because Google Scholar rankings and result counts are dynamic, no displayed hit count was used as a study-identification denominator or as PRISMA-style accounting. The search was intentionally targeted rather than exhaustive; accordingly, the design does not claim systematic-review coverage.

Titles and abstracts were screened iteratively, and full texts were selected purposively to maximize variation in computing platform

http://www.doi.org/10.62341/istj-vol39-1-ri51

(desktop GPU, embedded NVIDIA device, Raspberry Pi), deployment setting (model benchmark, IP-camera system, edge, edge–cloud), and temporal evidence. Publisher pages and DOI records were used to verify bibliographic details. Targeted citation chasing and a focused follow-up search were then used to deliberately identify stronger L3 examples, adding two 2026 studies with explicit end-to-end or system-total measurements [19], [20]. This step was undertaken to test the framework against well-reported temporal evidence rather than constructing a sample dominated by weak-reporting cases. Because purposive selection can introduce selection bias and does not yield a representative sampling frame, all percentages in this review are interpreted only as descriptive summaries of the selected ten-study corpus.

3.2 Eligibility criteria

Include: peer-reviewed empirical studies of deep-learning object detection used in security/video-surveillance contexts; publication date 2020–2026; English language; and an explicit real-time, speed, latency, throughput, edge, or deployment claim relevant to object detection.

Exclude from the empirical synthesis: review articles, non-surveillance domains such as purely autonomous-driving or remote-sensing studies, tracking-only work with no relevant detector evaluation, theory without implementation results, and retracted studies.

Background reviews were permitted for contextual framing but were not counted among the ten empirical studies.

3.3 Data extraction

For each empirical paper, the review extracted: surveillance context, detector/model, dataset or video source, hardware, input resolution when reported, FPS, latency, operational deployment evidence, edge/embedded execution, stated real-time threshold, and the temporal boundary represented by the reported metric. Particular attention was given to whether preprocessing, post-processing, tracking/analytics, network transmission, rendering, or alert generation were included in timing.

http://www.doi.org/10.62341/istj-vol39-1-ri51

3.4 Descriptive temporal-evidence typology

A four-level descriptive typology was used to classify the completeness of temporal evidence. It is a proposed diagnostic framework for this review, not a validated quality score, not a risk-of-bias instrument, and not a ranking of detector accuracy. The levels were assigned from the temporal boundary explicitly documented in each full text:

L0 — Claim-only evidence: the paper uses "real-time" or equivalent language but does not report an interpretable temporal measure for the proposed system.

L1 — Model-level timing evidence: the paper reports detector FPS and/or isolated inference latency, without a defined operational system boundary.

L2 — Operational throughput or partial-pipeline evidence: timing is measured on live, edge, embedded, multi-stage, or deployed processing, but a full camera-to-output/decision boundary is not defined.

L3 — Explicit system-total or end-to-end evidence: a total temporal boundary is explicitly defined or a set of relevant processing stages is reported and summed into a system-total latency.

A higher level indicates greater completeness of the temporal boundary, not a superior detector. For example, a slower but explicitly measured edge pipeline can provide stronger deployment evidence than a faster model-only GPU benchmark. The L0–L3 assignments were performed by one primary reviewer and rechecked against the full texts when the boundary was ambiguous; an independent second-rater calibration was not available. Accordingly, no inter-rater agreement statistic was computed. Future validation of the typology should use at least two independent raters and report agreement measures such as Cohen's kappa (two raters) or Fleiss' kappa (more than two raters).

3.5 Structured evidence appraisal

To address the reliability of the underlying evidence separately from the L0–L3 boundary classification, each empirical study was subjected to a structured reporting appraisal across five dimensions: (1) an explicit and interpretable temporal metric; (2) sufficiently specified hardware and input context; (3) a clearly defined timing

http://www.doi.org/10.62341/istj-vol39-1-ri51

boundary; (4) a timing protocol described well enough to understand what was measured and under what conditions; and (5) operational evidence such as live/field execution, sustained load, edge deployment, or networked processing. Each dimension was coded descriptively as Yes, Partial, or Not Reported from the published full text.
This appraisal was used to qualify interpretation of each study rather than to compute a composite “quality score” or exclude papers. No validated risk-of-bias instrument was identified that directly fits heterogeneous engineering timing studies of deployed object-detection systems; therefore, the appraisal should be understood as a transparent reporting-reliability check rather than a formally validated quality scale. As with the L0–L3 coding, it was conducted by one primary reviewer and should be independently replicated in future work.

3.6 Standards-informed operational lens

The public scope of IEC 62676-6:2026 was used as a non-normative interpretive lens. The standard addresses performance, interfaces, environmental adaptability, testing, evaluation, and operating stresses for real-time intelligent video analysis systems [8]. Because the full paid standard was not used, this paper does not claim IEC compliance, clause-level assessment, or certification. The standards-informed question was limited to whether published evaluations provide operationally meaningful context beyond a detector benchmark, including environmental stress, obscuration, image degradation, or realistic deployment load.

4. Results

4.1 Characteristics of the selected studies

Ten empirical studies were included in the critical synthesis. Together they cover model-only GPU benchmarking, office IP-camera surveillance, Raspberry-Pi edge processing, Jetson-based edge computing, edge–cloud collaboration, behavior/anomaly-enhanced surveillance, parking-lot monitoring, weapon detection on embedded hardware, and end-to-end autonomous surveillance. Table 1 summarizes the study contexts and temporal evidence.

http://www.doi.org/10.62341/istj-vol39-1-ri51

Table 1. Empirical studies included in the targeted critical review and their temporal-evidence classification.

Study	Context / model	Hardware / input	Key temporal evidence	Level
Nimma et al. [11]	Attention Transformer-YOLOv8; surveillance	"High-end hardware"; preprocessing sizes 416/608 stated	5.2 ms/frame inference; no explicit E2E	L1
Chen et al. [12]	Edge–cloud YOLOv3 surveillance	3× Jetson Xavier NX; cloud RTX 2080 Ti	15.7 FPS at edge; ~94% backbone-bandwidth saving	L2
Al-E'mari et al. [13]	YOLOv8 + anomaly / behavior analysis	Core i9; NVIDIA RTX 3000-class entry in table; 32 GB RAM	45 FPS average; live/field-oriented evaluation	L2
Azatbekuly et al. [14]	Office IP cameras; RTSP/OpenCV; YOLOv8n	GTX 1660 SUPER; 504×706 RGB stream	34 FPS and 2.2 ms listed for YOLOv8n; system boundary unclear	L1
Oguine et al. [15]	YOLOv3 smart surveillance; COCO + collected footage	Core i5; NVIDIA Quadro RTX 5000 environment	Real-time claim; no explicit measured FPS/latency for proposed system	L0
Nguyen et al. [16]	Human detection at the edge	Raspberry Pi 3B; 320×320 selected	1.96 FPS; ~2 FPS explicitly treated as sufficient task target	L2
Zhao et al. [17]	CMCA-YOLO parking-lot surveillance	RTX 3060 Laptop GPU; i9-12900H; 640×640	143.7 FPS; mAP@0.5=0.895; edge deployment future work	L1
Saleem & Jerith [18]	SurveilSmartAI / ObjTrackNet	RTX 4090; Jetson Xavier/Orin; Raspberry Pi 4 + NCS2	15.2 ms total on cloud; 35.4 ms total on Xavier; stage breakdown and stream scaling	L3

Beyond FPS: A Critical Review of Temporal Evidence for Real-Time Object Detection in Video Surveillance
http://www.doi.org/10.62341/istj-vol39-1-ri51

Study	Context / model	Hardware / input	Key temporal evidence	Level
Fierro Silva et al. [19]	Weapon detection; YOLOv8s/9s/10s/11s/26s; live surveillance	Jetson Nano; 640×640	Pre/infer/post + total latency; ~124–315 ms; 3.1–7.8 FPS depending on model/scenario	L3
Castro-Castaño et al. [20]	Autonomous defense surveillance; RTSP/WebRTC + YOLOv11	Tesla T4 training; Apple M2 operational testbed; 640×480 at 30 FPS	~267 ms baseline E2E; ~1039 ms with AI; ~772 ms added pipeline overhead; >300 h tests	L3

The structured reporting appraisal confirmed that evidence completeness was uneven across the selected studies. Model-centric papers commonly reported headline speed values but did not always define the measured boundary or provide a reproducible timing protocol, whereas the three L3 studies were the clearest about stage-level or system-total timing. Operational realism was a separate dimension: some live or edge studies still lacked a complete end-to-end boundary. Because this appraisal is descriptive and was applied by a single reviewer, these observations are used to qualify evidence strength rather than as formal risk-of-bias grades.

4.2 How "real-time" was evidenced

Within the purposive ten-study sample, seven studies (70.0%) reported FPS and five (50.0%) reported an explicit latency value. Three studies (30.0%) provided a clearly defined system-total or end-to-end temporal boundary: Saleem and Jerith [18], Fierro Silva et al. [19], and Castro-Castaño et al. [20]. Hardware was specified at least partially in nine studies (90.0%). These proportions describe only the selected maximum-variation sample and should not be interpreted as prevalence estimates for the wider surveillance literature.

The temporal-evidence typology produced one L0 study (10.0%), three L1 studies (30.0%), three L2 studies (30.0%), and three L3 studies (30.0%). Accordingly, seven of ten studies did not provide an explicit system-total/end-to-end temporal boundary.

Figure 2 visualizes this descriptive distribution; its percentages are sample descriptors, not population estimates.

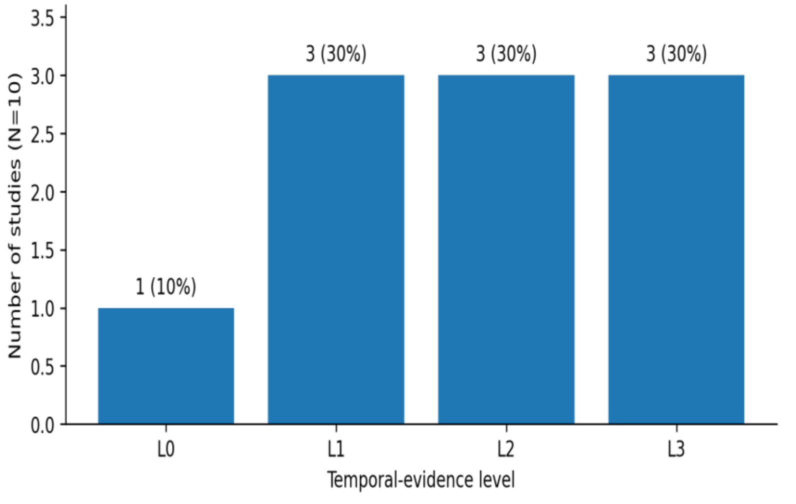

Figure 2. Distribution of the descriptive temporal-evidence levels. L3 reflects boundary completeness, not detector superiority.

4.3 Hardware and application dependence

Reported throughput varied dramatically across studies. Among papers that reported FPS, values ranged from 1.96 FPS on a Raspberry Pi 3B [16] to 143.7 FPS on an RTX 3060 Laptop GPU [17], a greater than 73-fold span. This range must not be interpreted as a detector ranking because the studies used different models, datasets, resolutions, timing scopes, and workloads. Instead, it illustrates the scale of hardware- and configuration-dependence within the purposive sample.

The Nguyen et al. study [16] is particularly informative because it explicitly links the temporal requirement to the application: approximately 2 FPS was treated as sufficient for its human-surveillance design goal on constrained hardware. In contrast, Zhao et al. [17] reports 143.7 FPS on a laptop GPU, while Saleem and Jerith [18] report 62 FPS on an RTX 4090 cloud server, 28 FPS on Jetson Xavier NX, 32 FPS on Jetson Orin Nano, and 12 FPS on Raspberry Pi 4 with an accelerator. These examples show that a universal 25- or 30-FPS rule would be both too strict for some event-

http://www.doi.org/10.62341/istj-vol39-1-ri51

detection tasks and too weak for other high-load, multi-stream applications.

4.4 Inference time is not system latency

Nimma et al. [11] report 5.2 ms per-frame inference, which would correspond mathematically to approximately 192 idealized serialized inferences per second if the timing represented a batch-1 detector loop with no other costs. The paper does not report this derived FPS and does not define the 5.2 ms as camera-to-output latency. By comparison, Saleem and Jerith [18] report a 15.2 ms system total composed of 4.3 ms preprocessing, 8.6 ms inference, and 2.3 ms post-processing. Fierro Silva et al. [19] make the same boundary explicit on Jetson Nano and show that post-processing can dominate total latency: roughly 175–184 ms of post-processing pushes several YOLO variants to approximately 296–315 ms total despite inference near 105–118 ms. These examples demonstrate that the temporal boundary, not the detector forward pass alone, determines deployment-level responsiveness.

The Azatbekuly et al. [14] study illustrates another reporting ambiguity. Its table lists 34 FPS and 2.2 ms latency for YOLOv8n. A 2.2 ms per-frame total would imply roughly 455 FPS under the same serialized timing boundary, which is not consistent with 34 FPS. The values may represent different benchmark scopes, devices, batching conditions, or source conventions, but these are not clarified. This example shows why every FPS or latency value should be accompanied by a timing protocol and boundary definition.

Castro-Castaño et al. [20] extend the boundary even further to the complete surveillance delivery path. Their timestamp-based measurement increased from approximately 267 ms for baseline WebRTC delivery to about 1039 ms when AI analytics were enabled. The resulting ~772 ms overhead was attributed to cumulative decoding, buffering, frame extraction, inference integration, synchronization, and delivery effects rather than to YOLO inference alone. This case is especially important because it shows how a model with millisecond-scale inference can coexist with a system response close to one second.

http://www.doi.org/10.62341/istj-vol39-1-ri51

4.5 Deployment realism and operational stress

Operational realism varied substantially. Chen et al. [12] deployed multiple edge devices connected to cameras and a cloud service. Azatbekuly et al. [14] implemented IP-camera ingestion, event triggers, databases, and notifications in an office surveillance system. Al-E'mari et al. [13] evaluated a higher-level surveillance pipeline including anomaly and behavior analysis. Nguyen et al. [16] evaluated the detector directly on constrained edge hardware. Fierro Silva et al. [19] processed live video streams on Jetson Nano, while Castro-Castaño et al. [20] evaluated an integrated networked surveillance path over more than 300 hours. Zhao et al. [17], by contrast, provides a strong model benchmark but explicitly treats edge-device integration as future work.

Environmental evidence was also uneven. Table 2 provides a standards-informed descriptive audit of operational stress evidence. Explicit lighting or low-light variation was evaluated in three of ten studies; occlusion or crowding in four; and weather/environmental variation in only one. No included study reported a controlled vibration or camera-shake stress test. Three studies provided meaningful sustained-load, multistream, or network-load evidence. These findings are not IEC compliance judgments; they simply show that operational stress dimensions highlighted by the public scope of IEC 62676-6:2026 [8] remain much less consistently evaluated than detector accuracy or headline speed.

Table 2. Standards-informed operational evidence reported in the ten-study purposive sample.

Study	Live / field	Lighting / low-light	Occlusion / crowding	Weather / degradation / vibration	Sustained / network load
Nimma et al. [11]	NR	Y	Y	NR	NR
Chen et al. [12]	Y	NR	NR	NR	Y
Al-E'mari et al. [13]	Y	NR	Y	NR	NR
Azatbekuly et al. [14]	Y	NR	NR	NR	NR
Oguine et al. [15]	P	NR	NR	P	NR

http://www.doi.org/10.62341/istj-vol39-1-ri51

Study	Live / field	Lighting / low-light	Occlusion / crowding	Weather / degradation / vibration	Sustained / network load
Nguyen et al. [16]	P	NR	NR	NR	NR
Zhao et al. [17]	P	Y	Y	Y / P*	NR
Saleem & Jerith [18]	P	NR	NR	NR	Y
Fierro Silva et al. [19]	Y	Y	Y	P	NR
Castro-Castaño et al. [20]	Y	P	Y	P	Y

Codes: Y = explicitly evaluated/reported; P = partial or uncontrolled operational evidence; NR = not reported. * Zhao et al. report varied weather and camera-angle/lighting conditions, but not controlled vibration/shake testing. No study in the sample reported a controlled vibration or camera-shake stress test.

5. Discussion

5.1 Main finding: real-time is relational, not intrinsic

The main finding of this review is that real-time capability is not an intrinsic property of an object detector. It is a relational property of a detector–hardware–workload–application configuration and of the temporal boundary being measured. A paper can legitimately report a fast detector without demonstrating a fast surveillance system; conversely, a relatively low-FPS detector can still satisfy an application if the event-detection requirement is defined accordingly. The problem arises when the same label—real-time—is applied to these different situations without qualification.

5.2 Why FPS alone is insufficient

FPS is useful but incomplete. It describes throughput, not necessarily response latency. A pipeline can maintain high average FPS while individual frames experience high queuing or network delay. FPS also hides tail behavior: average throughput may remain acceptable even when occasional frames exceed the response deadline that matters in a security application. Contemporary object-detection and surveillance benchmark reviews similarly show that speed figures are reported across heterogeneous

http://www.doi.org/10.62341/istj-vol39-1-ri51

architectures, devices, datasets, and computational budgets, limiting direct portability of headline FPS comparisons [3], [10]. For latency-sensitive deployments, reporting mean FPS should therefore be complemented by per-frame latency statistics, preferably including median and high-percentile values where feasible, plus sustained-load behavior under the intended number of camera streams.

Similarly, inference latency should not be interpreted as a synonym for end-to-end latency. The complete deployed path may include operations before and after the neural network. When tracking, behavior analysis, encryption, database writes, remote alerts, decoding, buffering, or browser delivery are added, detector inference can become only a fraction of the total delay. The explicit stage decompositions reported by Saleem and Jerith [18] and Fierro Silva et al. [19], together with the timestamp-based system measurement of Castro-Castaño et al. [20], demonstrate why model-only speed should not be used as a proxy for system responsiveness.

5.3 Architectural sources of latency in YOLO-family pipelines

Within the YOLO family, architectural evolution can shift latency between stages even when raw inference time changes little. Earlier dense-output designs rely on non-maximum suppression (NMS) and related candidate filtering after the network forward pass. The computational cost of these operations can grow with the number of candidate boxes, classes, and scene clutter. Consequently, crowded surveillance scenes may increase post-processing cost even if the neural-network inference time remains stable. Newer end-to-end or NMS-reduced designs attempt to move more decision-making into the network output itself, reducing downstream filtering overhead.
The Jetson Nano comparison of Fierro Silva et al. [19] illustrates this effect directly. YOLOv8s, YOLOv9s, and YOLOv11s had inference times broadly comparable to YOLOv10s and YOLOv26s, yet their post-processing costs were roughly two orders of magnitude larger, producing total latencies around 300 ms instead of about 125–130 ms. This demonstrates that “which YOLO version is faster?” cannot be answered solely from backbone complexity or forward-pass timing; preprocessing, output representation, NMS

http://www.doi.org/10.62341/istj-vol39-1-ri51

behavior, tracking, input resolution, and object density all influence the deployed latency budget.

5.4 Proposed minimum reporting set for real-time surveillance claims

Based on the gaps observed in the selected literature, the following minimum reporting set is proposed for studies that describe object detection as real-time. The set is intended as practical guidance rather than a validated scale or formal standard.

Table 3. Proposed minimum reporting set for reproducible real-time object-detection claims in video surveillance.

Item	Minimum information to report	Why it matters
1. Application requirement	Define the operational deadline or required detection rate and justify it.	Prevents an arbitrary universal FPS threshold.
2. Hardware	CPU, GPU/NPU, RAM/VRAM, edge device, accelerator, and power mode when relevant.	Real-time performance is hardware-dependent.
3. Input configuration	Resolution, batch size, precision/runtime, frame sampling, and number of streams.	These parameters strongly affect throughput and latency.
4. Timing boundary	State exactly what is timed: inference only, preprocessing+inference, detector+tracker, or camera-to-alert.	Allows valid interpretation of the reported number.
5. Timing protocol	Warm-up, repetitions, clock method, mean/median/P95, and whether I/O is included.	Improves reproducibility and reveals tail latency.
6. Pipeline decomposition	Report preprocessing, inference, post-processing/tracking, communication, rendering, and alert costs when relevant.	Identifies the actual bottleneck.
7. Sustained load	Evaluate expected simultaneous streams/cameras and prolonged operation.	Single-stream peak speed may not represent deployment load.

http://www.doi.org/10.62341/istj-vol39-1-ri51

Item	Minimum information to report	Why it matters
8. Operational stress	Report lighting, occlusion, crowding, weather, image degradation/shake, and network conditions as relevant.	Connects algorithmic speed to surveillance reliability.
9. Resource cost	Report memory, power/energy, and bandwidth for edge or distributed systems.	Feasibility depends on more than speed.
10. Reproducibility	Provide code/configuration/weights or sufficiently detailed versions and parameters.	Enables temporal-performance verification.

5.5 Implications for researchers and reviewers

For researchers, the practical implication is to define the application deadline before optimizing the detector. A security system that raises an intrusion alert within a specified response window should be evaluated against that D_app requirement, not against a generic frame-rate convention. A latency-based real-time claim should report whether the declared end-to-end boundary satisfies $T\_E2E \leq D\_app$ under the intended load and a stated latency statistic; continuous-stream systems should additionally report whether sustained throughput meets the required processing rate. For reviewers and editors, manuscripts should be asked to distinguish FPS, detector inference time, intermediate-stage latency, and end-to-end latency, and to avoid language implying field-ready real-time capability when only an offline GPU benchmark has been measured.

For practitioners, model-selection decisions should consider the complete deployment budget: camera count, resolution, device class, power, network availability, tracking or behavior modules, and the cost of maintaining acceptable latency under sustained load. Hardware-aware evaluation is therefore not a secondary engineering detail; it is part of the scientific meaning of a real-time claim.

5.6 Future extension: temporal–energy evidence

Future extensions of the proposed framework should evaluate energy consumption alongside temporal performance, particularly for edge and embedded surveillance platforms. Useful

http://www.doi.org/10.62341/istj-vol39-1-ri51

complementary measures include average power draw, joules per frame or per detection event, thermal/power mode, and performance per watt under sustained load. These measures should remain conceptually distinct from the temporal-evidence level: energy efficiency does not define whether a timing boundary is complete, but it determines whether the measured performance can be sustained within the deployment's power and thermal envelope.
A system that satisfies a temporal requirement but cannot sustain that performance within the available power and thermal budget may still be unsuitable for real-world edge deployment. A future extension could therefore report a paired temporal–energy profile rather than expanding the L0–L3 scale itself, preserving the clarity of the temporal typology while adding a deployment-efficiency dimension.

5.7 Limitations

This paper has several limitations. First, it is a targeted critical review based primarily on Google Scholar rather than a systematic multi-database review; the ten empirical studies form a purposive maximum-variation sample and should not be used to estimate prevalence across the entire literature. Percentages reported in the Results are therefore descriptive within-sample summaries only. Second, purposive sampling and Google Scholar ranking may introduce selection bias, and no PRISMA-style systematic identification flow is claimed. Third, although targeted follow-up searching increased the number of L3 cases from one to three, the sample remains intentionally small and heterogeneous. Fourth, the structured evidence appraisal and the L0–L3 typology are proposed diagnostic tools rather than validated quality instruments; coding was performed by one primary reviewer, so independent inter-rater reliability (for example, Cohen's kappa) was not estimated. Fifth, the studies differ in detector family, dataset, hardware, application, and timing protocol, precluding a valid meta-analysis of FPS or accuracy. Sixth, some timing values could not be independently reproduced because code, exact runtime settings, or timing protocols were incomplete. Finally, the standards-informed discussion uses only the publicly available scope of IEC 62676-

6:2026 [8]; it is not a clause-level standards assessment or a claim of compliance.

6. Conclusion

The term "real-time" in video-surveillance object detection currently covers a wide range of evidence, from claim-only language to model inference timing, edge-device throughput, and explicit system-total latency. Within the ten-study purposive sample, seven studies reported FPS, five reported an explicit latency value, and three provided a clearly defined system-total or end-to-end temporal boundary. These counts describe only the selected corpus and are not estimates for the wider field. Reported throughput still spanned more than 73-fold across heterogeneous hardware and workloads. The three L3 examples further show that post-processing, decoding, buffering, networking, and delivery can materially change total response time even when detector inference is fast. These findings demonstrate that FPS or inference time alone cannot establish deployment-level real-time performance.

The most defensible interpretation is that real-time performance depends on what was timed, where it was timed, under what operational load, and against which application deadline. Future surveillance studies should therefore define D_app, specify hardware and input conditions, state the timing boundary, report a reproducible timing protocol, and—where practical—measure the complete processing path under realistic load and environmental conditions. The principal methodological contribution of this paper is the combined L0–L3 evidence typology and minimum reporting set; both should be treated as proposed diagnostic guidance until independently validated across raters and larger study samples. For edge and embedded systems, temporal reporting should increasingly be complemented by power and energy measurements so that responsiveness and sustainability of deployment can be assessed together.

References

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in Proc. IEEE

http://www.doi.org/10.62341/istj-vol39-1-ri51

Conf. Computer Vision and Pattern Recognition (CVPR), 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.
[2] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS," Machine Learning and Knowledge Extraction, vol. 5, no. 4, pp. 1680–1716, 2023, doi: 10.3390/make5040083.
[3] E. Edozie, A. N. Shuaibu, U. K. John, and B. O. Sadiq, "Comprehensive review of recent developments in visual object detection based on deep learning," Artificial Intelligence Review, vol. 58, Art. no. 277, 2025, doi: 10.1007/s10462-025-11284-w.
[4] F. Alamri, "Comprehensive study on object detection for security and surveillance: A concise review," Multimedia Tools and Applications, vol. 84, no. 34, pp. 42321–42352, 2025, doi: 10.1007/s11042-025-20801-6.
[5] S. O. Olansile, O. A. Amodu, M. S. Adam, I. O. Elesemoyo, and R. A. Raja Mahmood, "Deep learning for secure imaging and video surveillance in smart cities: from bibliometric mapping to real-world implementations," Frontiers in Imaging, vol. 5, Art. no. 1854187, 2026, doi: 10.3389/fimag.2026.1854187.
[6] D. R. Patrikar and M. R. Parate, "Anomaly detection using edge computing in video surveillance system: review," International Journal of Multimedia Information Retrieval, vol. 11, no. 2, pp. 85–110, 2022, doi: 10.1007/s13735-022-00227-8.
[7] M. A. Ezzat, M. A. Abd El Ghany, S. Almotairi, and M. A.-M. Salem, "Horizontal Review on Video Surveillance for Smart Cities: Edge Devices, Applications, Datasets, and Future Trends," Sensors, vol. 21, no. 9, Art. no. 3222, 2021, doi: 10.3390/s21093222.
[8] International Electrotechnical Commission, IEC 62676-6:2026, "Video surveillance systems for use in security applications — Part 6: Performance testing and grading of real-time intelligent video content analysis devices and systems for use in video surveillance applications," 1st ed., Feb. 2026.
[9] A. Bello, A. S. Akintola, R. I. Abasiama, O. E. Esan, and O. A. Oluwaseun, "Real-time computer vision for safety and efficiency

in smart infrastructure," Discover Civil Engineering, vol. 3, Art. no. 62, 2026, doi: 10.1007/s44290-026-00457-3.
[10] K. U. Duja, I. A. Khan, and M. Alsuhaibani, "Video Surveillance Anomaly Detection: A Review on Deep Learning Benchmarks," IEEE Access, vol. 12, pp. 164811–164842, 2024, doi: 10.1109/ACCESS.2024.3491868.
[11] D. Nimma, O. Al-Omari, R. Pradhan, Z. Ulmas, R. V. V. Krishna, T. Y. A. B. El-Ebiary, and V. S. Rao, "Object detection in real-time video surveillance using attention based transformer-YOLOv8 model," Alexandria Engineering Journal, vol. 118, pp. 482–495, 2025, doi: 10.1016/j.aej.2025.01.032.
[12] Y.-Y. Chen, Y.-H. Lin, Y.-C. Hu, C.-H. Hsia, Y.-A. Lian, and S.-Y. Jhong, "Distributed Real-Time Object Detection Based on Edge-Cloud Collaboration for Smart Video Surveillance Applications," IEEE Access, vol. 10, pp. 93745–93759, 2022, doi: 10.1109/ACCESS.2022.3203053.
[13] S. R. Al-E'mari, Y. Sanjalawe, and H. Alqudah, "Integrating Enhanced Security Protocols with Moving Object Detection: A Yolo-Based Approach for Real-Time Surveillance," in 2024 2nd International Conference on Cyber Resilience (ICCR), 2024, doi: 10.1109/ICCR61006.2024.10532863.
[14] N. Azatbekuly, A. Mukhanbet, and S. D. Bekele, "Development of an Intelligent Video Surveillance System Based on YOLO Algorithm," in 2024 IEEE 4th International Conference on Smart Information Systems and Technologies (SIST), Astana, Kazakhstan, 2024, pp. 498–503, doi: 10.1109/SIST61555.2024.10629617.
[15] K. J. Oguine, O. C. Oguine, and H. I. Bisallah, "YOLO v3: Visual and Real-Time Object Detection Model for Smart Surveillance Systems (3s)," in 2022 5th Information Technology for Education and Development (ITED), 2022, pp. 1–8, doi: 10.1109/ITED56637.2022.10051233.
[16] H. H. Nguyen, V. T. Bui, T. N. Ta, H. M. Pham, N. C. Nguyen, and D. M. Nguyen, "YOLO Based Real-Time Human Detection for Smart Video Surveillance at the Edge," in 2020 IEEE Eighth International Conference on Communications and Electronics

http://www.doi.org/10.62341/istj-vol39-1-ri51

(ICCE), 2021, pp. 439–444, doi: 10.1109/ICCE48956.2021.9352144.

[17] N. Zhao, K. Wang, J. Yang, F. Luan, L. Yuan, and H. Zhang, "CMCA-YOLO: A Study on a Real-Time Object Detection Model for Parking Lot Surveillance Imagery," Electronics, vol. 13, no. 8, Art. no. 1557, 2024, doi: 10.3390/electronics13081557.

[18] L. A. Saleem and G. G. Jerith, "A deep learning driven framework with ObjTrackNet for real-time object and human detection in smart video surveillance," Discover Computing, vol. 29, Art. no. 238, 2026, doi: 10.1007/s10791-026-10143-8.

[19] C. J. Fierro Silva, C. Del-Valle-Soto, C. Bran, and J. Varela-Aldás, "Comparative analysis of previous YOLO detectors and YOLOv26s for real-time weapon detection in video surveillance," Frontiers in Computer Science, vol. 8, Art. no. 1789702, 2026, doi: 10.3389/fcomp.2026.1789702.

[20] J. J. Castro-Castaño, W. E. Chirán-Alpala, G. A. Giraldo-Martínez, J. D. Ortega-Pabón, E. C. Rodríguez-Amézquita, D. F. Gallego-Franco, and Y. A. Garcés-Gómez, "Low-Latency Autonomous Surveillance in Defense Environments: A Hybrid RTSP-WebRTC Architecture with YOLOv11," Computers, vol. 15, no. 1, Art. no. 62, 2026, doi: 10.3390/computers15010062.

[21] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.1109/TPAMI.2016.2577031.

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," in Computer Vision – ECCV 2016, pp. 21–37, 2016, doi: 10.1007/978-3-319-46448-0_2.

[23] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," in Computer Vision – ECCV 2020, pp. 213–229, 2020, doi: 10.1007/978-3-030-58452-8_13.