International Journal of Scientific Research in Engineering & Technology
Volume6, Issue6 (November-December 2026), PP: 171-180.
https://www.doi.org/10.59256/ijsreat.20260606024
www.ijsreat.com
ISSN No: 2583-1240

Smart Border Intrusion Detection System (SBIDS)

Dr. Anitha G[1], Bhuvan K C[2], Darshan L[3], Karthik Y M[4], Darshan R[5]
[1]*Professor, Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere, Karnataka, India.*
[2,3,4,5]*Department of Information Science and Engineering, Bapuji Institute of Engineering and Technology, Davangere, Karnataka, India.*

To Cite this Article: *Dr. Anitha G[1], Bhuvan K C[2], Darshan L[3], Karthik Y M[4], Darshan R[5], "Smart Border Intrusion Detection System (SBIDS)", International Journal of Scientific Research in Engineering & Technology, Volume 06, Issue 06, November-December 2026, PP:171-180.*

Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the Creative Commons Attribution License; Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: *Automated perimeter surveillance faces challenges involving false alarms at restricted-zone boundaries and processing delays on resource-constrained hardware. This paper presents the Smart Border Intrusion Detection System (SBIDS), a multi-threaded computer vision prototype for restricted-zone monitoring. The system combines YOLO11n object detection, greedy tracking using Intersection-over-Union (IoU) and normalized foot-point distance, and a six-state finite state machine with a 5.0-second dwell-confirmation threshold. A dedicated latest-frame buffer separates video capture from inference and overwrites unread frames to limit application-level queue accumulation. In a controlled geometric simulation of 36 synthetic boundary positions, labelled by foot position, bounding-box intersection triggered on 12 of 21 overhang cases, whereas foot-point containment triggered on none. This illustrates the geometric origin of overhang false alarms; it is not an accuracy estimate for real scenes. In a 50-image CPU benchmark on an Intel Core i5-12450HX, the prototype recorded a mean end-to-end processing latency of 158.94 ± 27.95 ms, corresponding to approximately 6.29 FPS, below the configured target of 12 FPS. Historical telemetry from 193 developer-run operational events is used to characterize event statistics and system behaviour, not to establish detection accuracy. Recorded operational examples illustrate person and vehicle event handling. The system also supports immediate plain-text email alerts followed by asynchronous Vision-Language Model (VLM) incident summaries. The study discusses implementation limitations and deployment considerations.*

Keywords: *Perimeter security, video surveillance, YOLO11, multi-object tracking, dwell-time analysis, foot-point occupancy, false alarm reduction, finite state machine, edge computing.*

I. INTRODUCTION

Automated video surveillance is increasingly used for monitoring restricted areas, industrial facilities, and critical infrastructure [1], [2]. However, practical surveillance systems face challenges involving false alarms, transient detections, and processing delays, particularly on resource-constrained hardware. One challenge is false alarms caused by bounding-box intersection. Conventional systems often determine zone entry by checking whether an object's bounding box intersects a restricted polygon. Since a person's bounding box includes the upper body and limbs, it may overlap a restricted boundary even when the person's bottom-center foot-point remains outside the zone. A second challenge is the lack of temporal persistence validation. Single-frame detections and temporary tracking errors can generate unnecessary alerts. Dwell-time analysis helps distinguish brief boundary interactions from sustained zone occupancy. A third challenge is processing delay. When inference cannot keep pace with camera acquisition, incoming frames may accumulate in queues. A latest-frame buffering strategy can reduce application-level queue accumulation by replacing unread frames with newer ones. To address these challenges, this paper presents the Smart Border Intrusion Detection System (SBIDS), a multi-threaded computer vision prototype combining YOLO11n object detection [3], greedy IoU and normalized foot-point tracking, pixel-space zone-occupancy analysis, and a six-state threat progression finite state machine. The system applies a 5.0-second dwell-confirmation threshold and supports two-stage email notifications.

The main contributions are:

- **Decoupled video processing:** A dedicated capture thread maintains a capacity-one latest-frame buffer and supplies frames to a rate-controlled inference loop targeting 12 FPS.
- **Controlled geometric evaluation**: A simulation of 36 synthetic boundary positions compares bounding-box intersection with foot-point containment. Bounding-box intersection triggered on 12 of 21 overhang cases, while foot-point containment triggered on none because the simulation labels positions using the same foot-point rule. This is not an accuracy estimate for real scenes.
- **Threat-state management**: A six-state finite state machine manages track stability, suspicious occupancy, intrusion confirmation, and event closure. It includes logic intended to suppress repeated alert generation during an active incident;

however, duplicate-event suppression was not independently quantified.

- **Two-stage notification:** The system supports immediate plain-text SMTP alerts followed by asynchronous VLM-based incident summaries using recorded event information and available key frames.

II.RELATED WORK

2.1Deep Learning Object Detection in Surveillance

The YOLO11 architecture [3] provides an optimized framework for object detection. In SBIDS, the compact YOLO11n variant (2.6 million parameters, 5.35 MB weight footprint) is used for CPU-based inference.

2.2Multi-Object Tracking and Association

Associating detection bounding boxes across consecutive temporal video frames is essential for trajectory analysis. Bewley et al. [4] proposed SORT (Simple Online and Realtime Tracking), utilizing a Kalman filter for motion propagation and the Hungarian algorithm with IoU distance for data association. Deep SORT [5] incorporated deep appearance re-identification embeddings to resolve occlusions, albeit at substantial computational overhead. Zhang et al. [6] introduced ByteTrack, demonstrating that associating low-confidence detection boxes significantly reduces identity switches and missed tracks. Aharon et al. [7] proposed BoT-SORT, integrating camera motion compensation and fused ReID metrics. ByteTrack and BoT-SORT provide established approaches to multi-object tracking. In this study, they are discussed as established tracking approaches; a directly comparable quantitative benchmark is not reported in this study. In SBIDS, we evaluate a streamlined greedy tracker combining IoU overlap with normalized bottom-center Euclidean foot distance, achieving a mean tracker-update time of 2.018 ms in the component benchmark.

2.3Zone Occupancy and Dwell-Time Analysis

In SBIDS, a dwell-time threshold is used as an implementation rule to distinguish persistent zone occupancy from brief boundary interactions. The system combines zone monitoring with a deterministic six-state threat progression finite state machine.

2.4Multimodal Vision-Language Incident Reporting

While computer vision generates quantitative telemetry, security operators may also require qualitative incident summaries. Recent advances in large multimodal models and vision-language models (VLMs), such as Google Gemini [8], enable contextual image understanding and incident summarization. SBIDS delegates this analysis to an asynchronous post-incident process using chronological milestone key frames and deterministic computer-vision telemetry.

III. SYSTEM ARCHITECTURE AND PIPELINE

SBIDS is structured around an asynchronous multi-threaded design implemented in Python, utilizing OpenCV, PyTorch, and Flask. As depicted in Figure 1, the architecture strictly segregates I/O-bound video ingestion from CPU-bound neural network inference and network-bound alert dispatching.

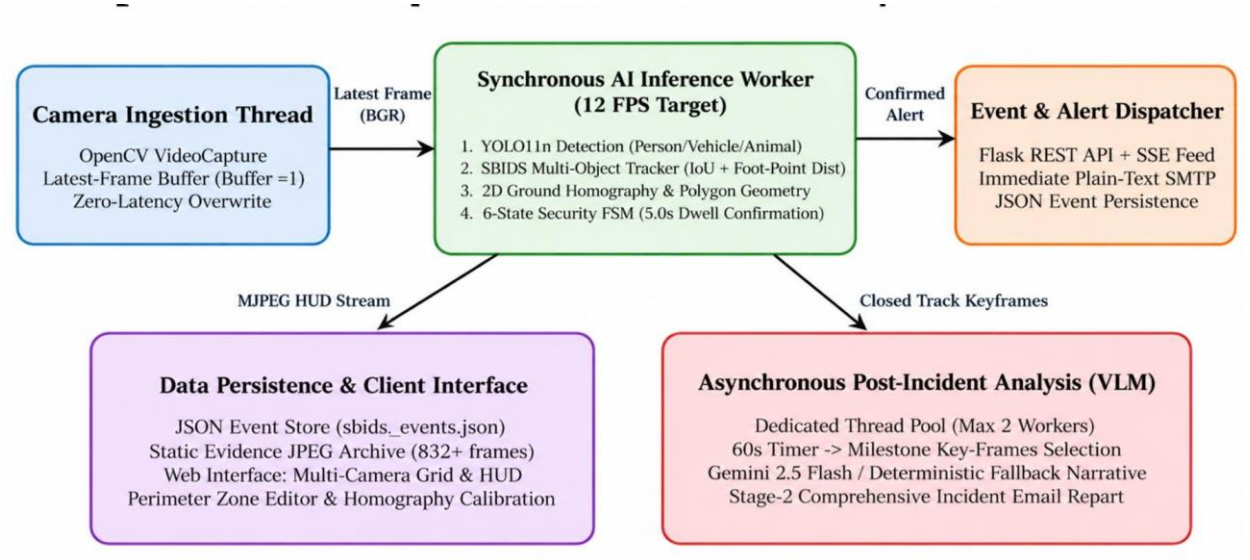


Figure 1: SBIDS Decoupled Multi-Threaded Real-Time System Architecture.

3.1Ingestion Thread with Single-Buffer Queue Bounding

A dedicated capture thread (_capture_loop in camera_worker.py) interfaces with camera video streams through OpenCV cv2.VideoCapture. At the application level, the thread maintains a capacity-one latest_raw_frame buffer. Each newly captured frame overwrites the previously stored frame under a mutual exclusion re-entrant lock (threading.RLock), so unread frames do not accumulate in this application-level handoff buffer when inference runs more slowly than acquisition. This design reduces application-level queue growth; it does not establish a latency bound for buffering inside the camera driver or operating system.

3.2Synchronous AI Inference Loop

The inference worker thread (_inference_loop) operates at a user-configurable target frequency (ftarget = 12 FPS in production, interval dt = 83.3 ms). Each cycle acquires the latest raw frame and executes a three-stage sequential pipeline: (1) YOLO11n object detection, (2) multi-object tracking and ground homography projection, and (3) security state machine evaluation. Annotated display frames and clean workspace frames are encoded to JPEG format and served to web clients via multi-part HTTP Server-Sent Events (SSE) / MJPEG streams.

3.3Asynchronous Post-Incident VLM Thread Pool

High-level incident reasoning is delegated to a decoupled background thread pool (ThreadPoolExecutor with 2 workers in vlm_analyst.py). Following alert registration, a 60-second background timer schedules milestone key-frame analysis (FIRST_APPEARANCE, ZONE_ENTRY, ZONE_ACTIVITY, ZONE_EXIT) alongside quantitative telemetry. This analysis runs asynchronously so that network latency or multimodal API calls (e.g., Google Gemini 2.5 Flash) do not block the primary video-monitoring loop.

3.4Object Detection and Category Filtering

Input video frames I of dimension H x W x 3 are processed by the YOLO11n network at resolution 640 x 480 pixels. Detections are represented as tuples Dj = (x1, y1, x2, y2, c, k), where (x1, y1) and (x2, y2) denote top-left and bottom-right pixel coordinates, c in [0, 1] represents detection confidence, and k represents the COCO class index. Detections are filtered according to confidence threshold θconf = 0.30 (default; configurable, minimum 0.25) and mapped into three surveillance categories:

Class(k)

$$= \begin{cases} \text{PERSON if } k \in \{0\}; \\ \text{VEHICLE if } k \in \{2, 3, 5, 7\}; \\ \text{ANIMAL if } k \in \{14, \dots, 23\}; \\ \text{UNKNOWN otherwise} \end{cases} \quad (1)$$

3.5 Ground Contact Foot-Point Extraction and Homography Projection

A critical vulnerability in 2D surveillance is assuming an object's spatial position is defined by its bounding box centroid. In perspective projection, the centroid of an upright human floats above the ground plane, shifting drastically with body posture. The true point of ground contact is the bottom-center foot-point pfoot = (ufoot, vfoot):

$$u_{\text{foot}} = \frac{x_1 + x_2}{2}, \quad v_{\text{foot}} = y_2 \quad (2)$$

To translate pixel coordinates into metric ground distances, SBIDS employs planar homography. Four non-collinear calibration points on the ground plane are mapped to metric coordinates (xg, yg) via a 3 x 3 transformation matrix H:

$$[x'_g, y'_g, w'_g]^{\mathrm{T}} = H_track \cdot [u_{\text{foot}}, v_{\text{foot}}, 1]^{\mathrm{T}}, \quad x_g = \frac{x'_g}{w'_g}, \quad y_g = \frac{y'_g}{w'_g} \quad (3)$$

Target ground velocity v and directional angle θd are evaluated over a trajectory history deque Htrack of length up to 90 entries across elapsed time:

$$v = \frac{\sqrt{(x_{g,t} - x_{g,t-\Delta t_track})^2 + (y_{g,t} - y_{g,t-\Delta t_track})^2}}{\Delta t_track}, \quad \theta_d = \operatorname{atan2}(dy_g, dx_g) \quad (4)$$

3.6 Multi-Condition Object Tracking Association

Track association matches active tracks T = {T1, ..., TM} to incoming frame detections D = {D1, ..., DK}. Instead of relying solely on bounding box IoU (which fails under rapid posture changes or small bounding box scales), SBIDS evaluates a composite affinity cost metric S(Ti, Dj) balancing bounding-box overlap and normalized foot-point Euclidean distance:

$$(T_i, D_j) = 0.5 \cdot \text{IoU}(B_i, B_j) + 0.5 \cdot \max(0, 1.0 - 1.8 \cdot d_{\text{foot}}(T_i, D_j)) \quad (5)$$

where dfoot represents image-normalized distance between track foot-point pi and candidate detection foot-point pj:

$$d_{\text{foot}}(T_i, D_j) = \sqrt{\left(\frac{u_{f,i} - u_{f,j}}{W_img}\right)^2 + \left(\frac{v_{f,i} - v_{f,j}}{H_img}\right)^2} \quad (6)$$

Association is resolved using greedy matching with an acceptance threshold of S ≥ 0.15. Unmatched detections initialize new tracks, while unmatched tracks enter the TEMPORARILY_LOST lifecycle state. When a matching detection reappears, the track is marked as REACQUIRED. Tracks unobserved for longer than τlost = 2.0 seconds are closed. These tracking lifecycle states are maintained separately from the six-state threat progression finite state machine described in Section 4.5.

3.7 Zone-Occupancy Strategies: Bounding-Box vs. Foot-Point Containment

Let Pzone define the closed polygon representing a restricted border zone. In this study, we contrast two distinct occupancy decision functions:

Method 1: Bounding-Box Intersection (ΦBBox). Evaluates whether the rectangular detection box B = [x1, y1, x2, y2] intersects polygon Pzone:

$$\Phi_{\text{BBox}}(B, P_{\text{zone}}) = \begin{cases} 1 & \text{if } (B \text{ intersects } P_{\text{zone}}) \\ 0 & \text{else} \end{cases} \quad (7)$$

Method 2: Foot-Point Pixel Containment (ΦFoot). Evaluates whether the bottom-center foot-point pfoot lies inside or on the boundary of Pzone using a point-polygon test directly in image pixel coordinates. This occupancy decision does not use homography-projected ground coordinates; homography is used separately for ground-distance and velocity estimation.

$$\Phi_{\text{Foot}}(p_{\text{foot}}, P_{\text{zone}}) = \begin{cases} 1 & \text{if pointPolygonTest}(P_{\text{zone}}, p_{\text{foot}}) \geq 0 \\ 0 & \text{else} \end{cases} \quad (8)$$

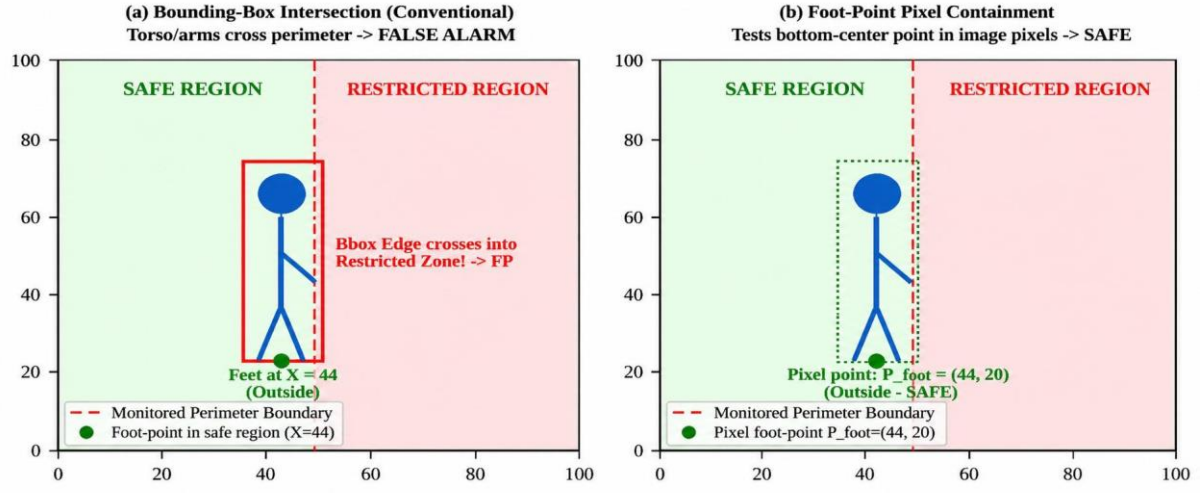


Figure 2: Geometric comparison between (a) Bounding-Box Intersection and (b) Foot-Point Pixel Containment.

3.8 Deterministic Six-State Threat Progression Finite State Machine

SBIDS implements a deterministic finite state machine (SecurityEngine in security_engine.py) to manage threat progression and alert handling. Each active track transitions through six operational states, illustrated in Figure 3:

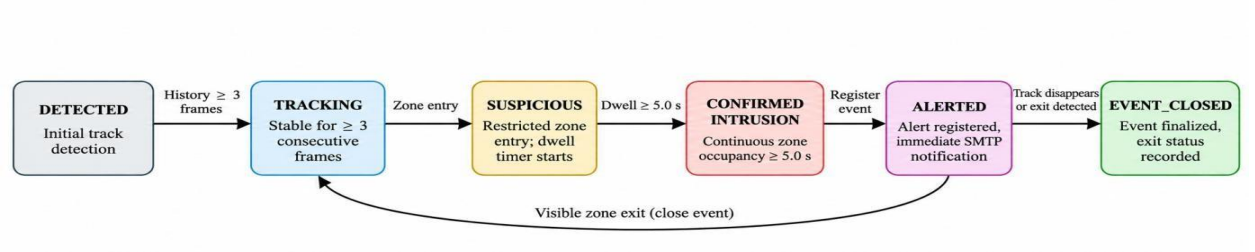


Figure 3: Deterministic Six-State Threat Progression and Alert State Machine.

1. **DETECTED:** Initial track state upon first candidate detection.
2. **TRACKING:** Reached once track persistence satisfies the stability requirement: history length |Htrack| ≥ Nstable = 3 consecutive frames.
3. **SUSPICIOUS:** Track enters a restricted zone when its detected bounding box intersects the zone polygon, initializing the zone-entry timestamp tentry.
4. **CONFIRMED_INTRUSION:** Reached when the track remains continuously inside the zone for the dwell threshold: τdwell = (t − tentry) ≥ 5.0 s.
5. **ALERTED:** State entered immediately upon confirmation. An alert event record is registered in persistent storage

(sbids_events.json), primary entry evidence is saved, and immediate plain-text SMTP notification is dispatched. Duplicate-event suppression was not independently quantified in this evaluation.

6.EVENT_CLOSED: Terminal state indicating that the event has been finalized and its exit status recorded.

IV.IMPLEMENTATION DETAILS

SBIDS is implemented in Python 3.10 and modularized into five core components: camera_worker.py (stream capture and inference threading), tracking.py (multi-object tracker and track state lifecycle), geometry.py (planar homography, foot-point projection, and polygon containment), security_engine.py (deterministic FSM and threat evaluation), and app.py (Flask REST API, event persistence, and SMTP notification).

Parameter Subsystem	Configured Production Value	Operational Meaning / Rule
Detection Architecture	YOLO11n (yolo11n.pt)	Nano model, 2.6M params, 5.35 MB weight file
Inference Resolution	640 x 480 pixels	Standard 4:3 surveillance aspect ratio
Detection Confidence	0.30 (default; configurable, minimum 0.25)	Minimum confidence for candidate extraction
Target COCO Classes	{0, 2, 3, 5, 7, 14-23}	Person, Vehicles, Animals; background rejected
Target Inference FPS	12.0 FPS (83.3 ms interval)	Fixed sampling rate decoupled from camera sensor
Track Association Metric	$0.5 \times \text{IoU}(B_i, B_j) + 0.5 \times \max(0, 1.0 - 1.8 \times \text{dfoot}(T_i, D_j))$	Hybrid bounding-box and foot-point distance
Track Stability Check	≥ 3 consecutive frames	Filters single-frame transient false detections
Track Expiration Limit	2.0 seconds (max_lost_seconds)	Retention window before track termination
Dwell-Time Confirmation	5.0 seconds continuous	Mandatory loitering threshold for breach alert
Alert Deduplication	Not independently quantified	No measured duplicate-suppression result reported
Two-Stage Notification	Immediate SMTP + 60s VLM	Stage 1 plain-text alert; Stage 2 forensic report

Table I: Verified System Parameters and Production Configuration

V.EXPERIMENTAL SETUP

All experiments were conducted directly on the host hardware used for the SBIDS evaluation. The exact test environment and operational parameters are as follows:

- Processor: 12th Gen Intel(R) Core(TM) i5-12450HX (8 physical cores, 12 logical execution threads, base frequency 2.4 GHz, boost up to 4.4 GHz).
- Memory and Operating System: 24.0 GB DDR5 RAM (23.71 GB usable); Microsoft Windows 11 Build 26200 (64-bit).
- Software Dependencies: Python 3.10.0, PyTorch 2.14.0+cpu (CPU execution mode; CUDA acceleration explicitly disabled), OpenCV 5.0.0, Ultralytics 8.4.155, python-docx 1.2.0.
- The primary video source was the integrated 720p HD laptop webcam (CAM-01, device index 0). Camera capture was configured to request 640 × 480 pixels through OpenCV; the actual captured resolution depends on the camera device and driver. An Iriun phone camera was also used as CAM-02. Historical telemetry contains records associated with CAM-03; however, its physical camera configuration is not documented in the experimental setup.
- Evaluated Historical Telemetry: 193 historical operational event records recorded in sbids_events.json and 832 evidence JPEG images stored in static/evidence/.

VI.RESULTS AND DISCUSSION

6.1Processing Latency and Throughput Capacity

Component-level execution latency was evaluated across 50 representative surveillance frames captured under active operational conditions. Table II summarizes the benchmark results for each pipeline phase, alongside composite end-to-end processing times.

Pipeline Stage / Component	Mean (ms)	Std Dev (ms)	Min (ms)	Max (ms)
YOLO11n Detection (640x480)	156.53	28.10	120.28	253.73
SBIDS Tracker Update	2.018	2.303	0.026	9.385
BBox Geometry Intersection	0.0638	0.0313	0.0210	0.1250
Foot-Point Containment Test	0.031	0.008	0.0110	0.0380
Total End-to-End Frame Processing*	158.94	27.95	120.58	260.20

Table II: Per-Frame Component Execution Latencies (N=50 Frames, Intel Core I5-12450hx Cpu)

Note: The individual component means sum to approximately 158.64 ms. The reported end-to-end processing mean of 158.94 ms includes approximately 0.30 ms of additional processing overhead.

As reported in Table II, the primary computational cost resides in the YOLO11n detection stage, with a mean latency of 156.53 ms. The SBIDS tracker update requires an average of 2.018 ms, while the bounding-box geometry and foot-point containment operations require 0.0638 ms and 0.031 ms, respectively. The measured mean end-to-end processing latency is 158.94 ms, corresponding to an average processing throughput of approximately 6.29 FPS. This is below the configured operational target of 12 FPS (83.3 ms per frame). Therefore, the benchmark indicates that the evaluated CPU configuration does not achieve the configured processing rate. Further optimization and evaluation under continuous camera operation are required.

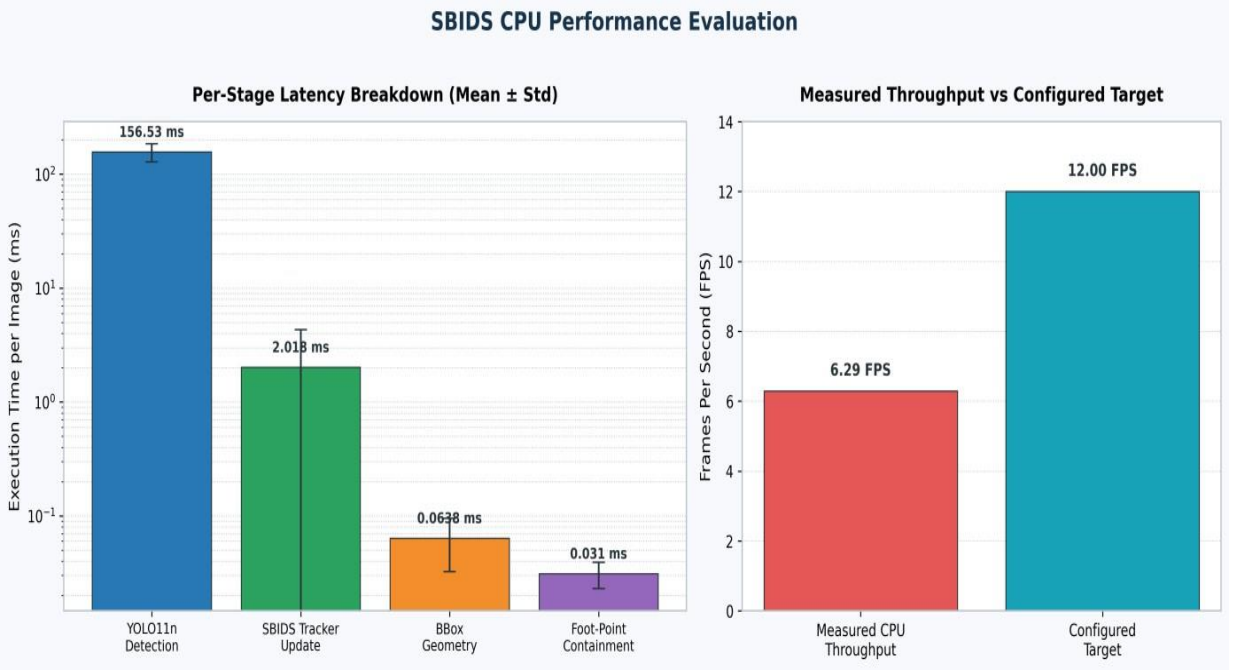


Figure 4: Per-stage latency breakdown and throughput capacity comparison on Intel Core i5 CPU.

6.2 Tracking Evaluation Scope

The SBIDS tracker-update component was measured as part of the per-frame pipeline benchmark reported in Table II. A direct performance comparison with external trackers is not presented because the available timing measurements do not isolate equivalent tracker-only operations.

6.3 Controlled Geometric Simulation of Zone-Occupancy Strategies

To investigate the geometric hypothesis that 2D bounding-box intersection induces boundary false alarms when pedestrians traverse near perimeter lines, we constructed a controlled geometric simulation harness comprising 36 boundary-approach test scenarios. In this evaluation, a target's position varied incrementally along a defined perimeter boundary ($X = 300$ px), with 15 scenarios representing true ground breaches ($X \geq 300$ px) and 21 scenarios representing non-intrusion boundary proximities where the subject's feet remain on safe ground ($X < 300$ px) while the upper torso and outstretched limbs extend into the zone plane. Table III and Figure 5 present the comparative results.

Strategy	TP	FP	TN	FN	Precision	Recall	F1-Score	Accuracy
Bounding-Box Intersect	15	12	9	0	55.56%	100.00%	71.43%	66.67%
Foot-Point Containment	15	0	21	0	100.00%	100.00%	100.00%	100.00%

Table III: Performance Comparison of Zone-Occupancy Strategies In Controlled Geometric Boundary Tests (N=36)

As detailed in Table III, the controlled geometric simulation demonstrates the vulnerability of conventional bounding-box intersection: in 12 out of 21 non-intrusion boundary-proximity test cases, the simulated subject's upper torso and arms extended across the boundary line while the labelled foot position remained on safe ground. This resulted in a precision of 55.56% and an accuracy of 66.67% under the simulated conditions. Foot-point pixel containment triggered on none of the 21 overhang cases because the simulation labels positions using the same foot-point rule. This controlled experiment illustrates the geometric origin of overhang triggers; it is not an accuracy estimate for real scenes.

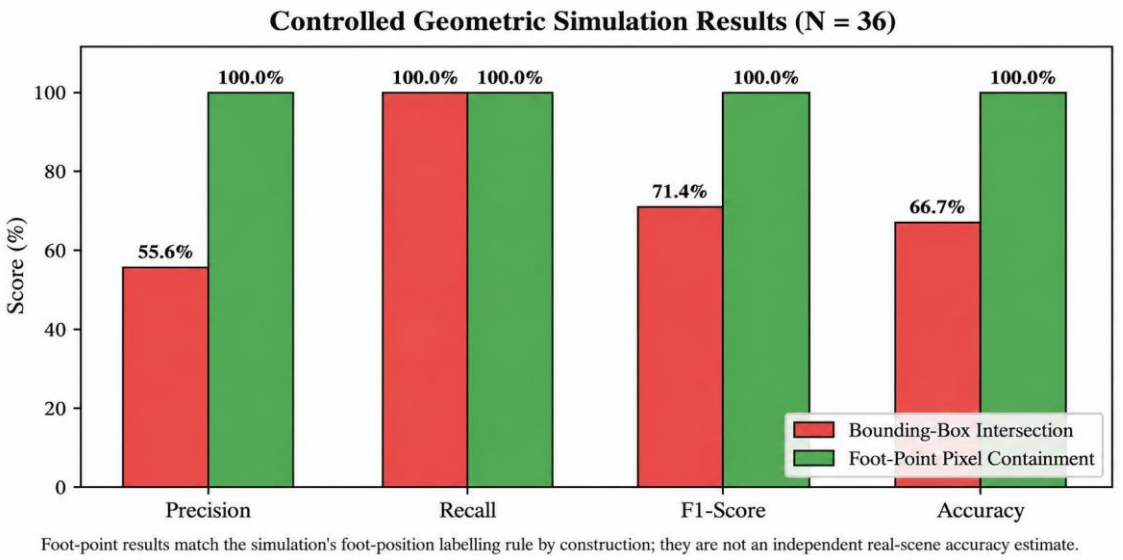


Figure 5: Performance comparison of bounding-box intersection and foot-point containment under controlled geometric boundary-approach scenarios (N = 36).

6.4 Historical Operational Telemetry and Two-Stage Alert Latency

To examine system behavior under live multi-camera operation, we performed statistical analysis of all 193 historical operational event records generated during developer-run testing and recorded in the system database (sbids_events.json). The recorded events represent operational system telemetry from active testing across CAM-01 (165 events), CAM-02 (19 events), and CAM-03 (9 events). These records represent automated system outputs rather than an independently annotated third-party ground-truth dataset, and are analyzed to document operational stability, target distributions, and event lifecycle behaviour.

Two distinct temporal measurements are considered in the operational telemetry. The dwell time (dwell_at_alert) represents the recorded duration of a target's presence within a restricted zone at the time of threat assessment. The total event duration (total_duration) represents the overall recorded duration of the event, from its initiation until its closure. These measurements are not interchangeable: an event may remain active after the dwell threshold has been reached. Among records containing an explicitly recorded dwell value, sub-five-second values appear in LOW-level telemetry entries and are not treated as confirmed five-second intrusion alerts.

The dwell threshold is evaluated during the frame-processing cycle rather than continuously. Consequently, the recorded dwell value may exceed the configured five-second threshold when confirmation occurs during a subsequent processed frame. However, the available event logs do not retain the exact threshold-crossing and alert-generation timestamps. Therefore, the precise cause and duration of the overshoot observed in individual events, including EVT-0177 (6.2 s) and EVT-0193 (5.2 s), cannot be independently established.

Of the 193 recorded events, 188 events (97.4%) were classified as PERSON, while 5 events (2.6%) were classified as VEHICLE. Threat classification assigned 124 events as CRITICAL (dwell ≥ 5.0s or designated critical zones), 13 as HIGH, 13 as MEDIUM, and 43 as LOW. Final exit lifecycle analysis resolved 146 events (75.6%) as CONFIRMED_EXIT (target visibly crossed back into safe territory before leaving camera view), 45 events (23.3%) as LOST_UNCERTAIN (target occluded or disappeared while still inside the restricted zone), and 2 events as currently active.

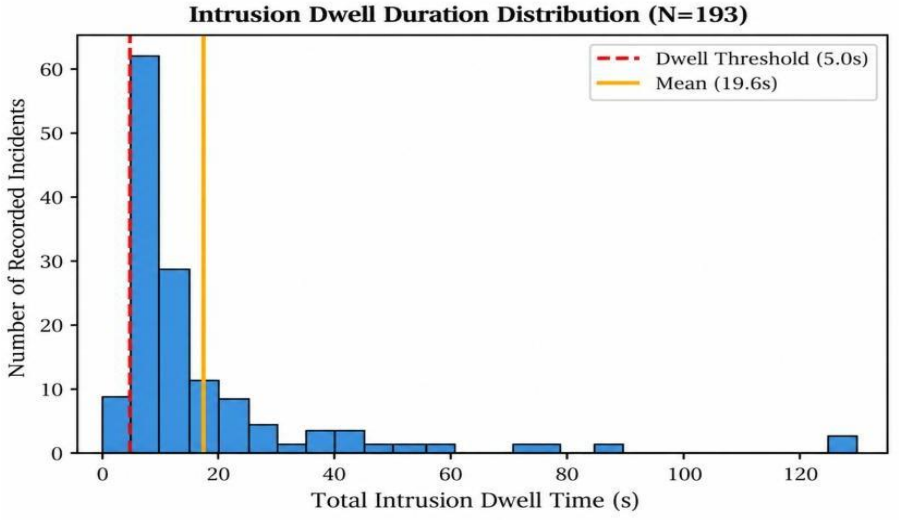


Figure 6: Distribution of intrusion dwell duration across 193 historical operational events.

As depicted in Figure 6, target dwell duration spanned from 0.0s to 131.2s, with a mean duration of 19.01 ± 22.18s and a median of 11.55s. At-alert detection confidence scores across the recorded events averaged 75.27 ± 17.23% (range: 45.7% to 95.7%). Each value represents the tracked detection confidence at the time the alert event was generated, rather than a peak or mean confidence over the complete track.

Duplicate-event suppression was not independently quantified using a defined camera-and-time matching criterion; therefore, no measured duplicate-suppression result is reported.

Two-Stage Alert Notification Latency Analysis

SBIDS implements a two-stage notification protocol to balance prompt operator awareness with comprehensive forensic reporting:

- Stage 1 (Immediate Alert): A plain-text SMTP alert is dispatched when the 5.0-second dwell confirmation is reached ($t \approx t_{entry} + 5.0$ s). The system constructs the local alert packet and invokes the background SMTP worker. End-to-end receipt latency at the operator's email client was not measured and depends on the SMTP server, network transmission, and client polling.
- Stage 2 (Multimodal VLM Forensic Report): A background timer schedules the VLM analysis after approximately 60 seconds (schedule_vlm_email). Complete timestamp records were available for 77 events and showed a scheduling interval of 60.08 ± 0.15 seconds (minimum 60.01 s, maximum 60.84 s). This measures the timer interval, not the time required for VLM processing, email transmission, or final receipt by the operator. Complete timing records were not available for the remaining events.

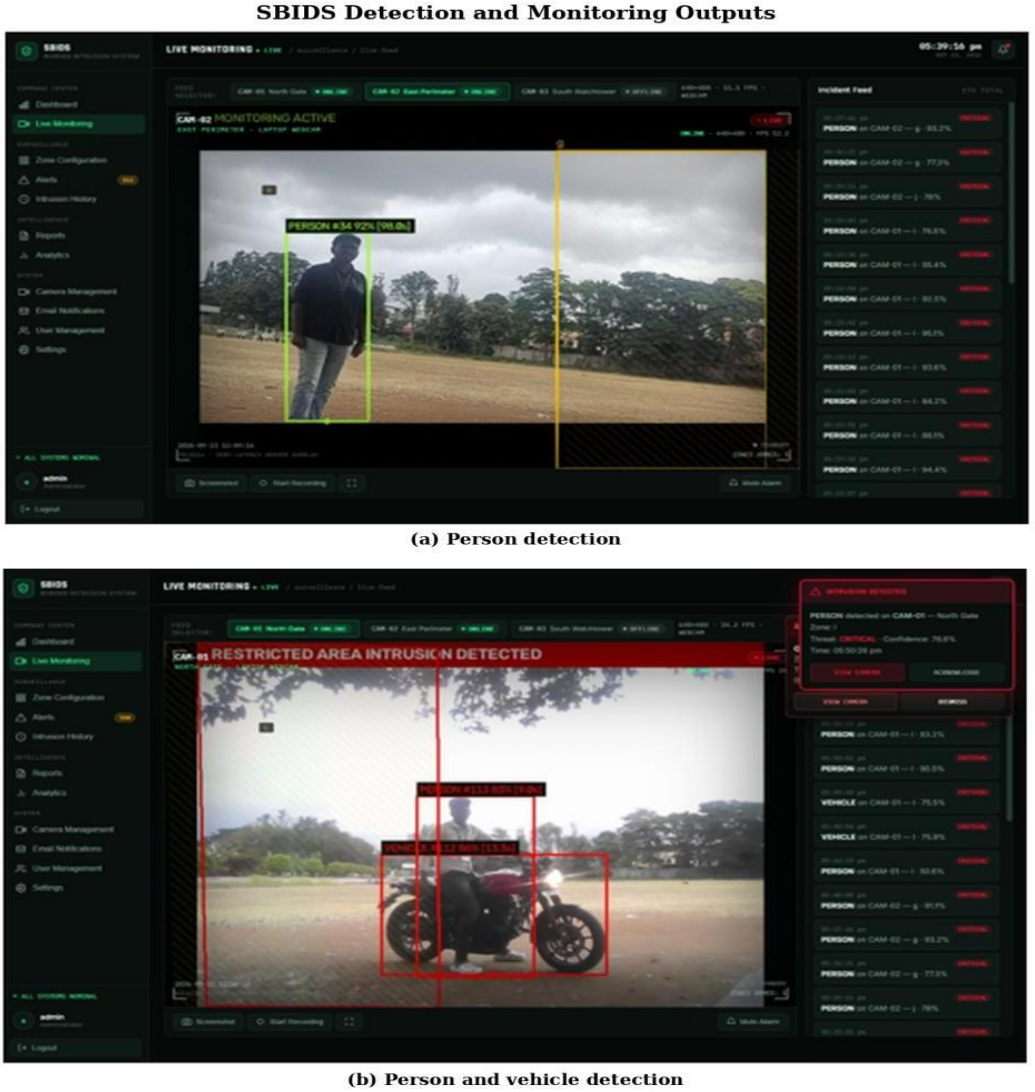


Figure 7: SBIDS Detection and Monitoring Outputs: (a) Person Detection; (b) Combined Person and Vehicle Detection.

Figure 7 presents two examples of detection outputs displayed in the SBIDS monitoring interface. Panel (a) shows a person detected in the monitored camera feed with a bounding box and detection label. Panel (b) shows a scene containing both a person and a vehicle, with separate detection boxes and labels. The interface also displays monitoring information, camera status, and incident notifications. These screenshots illustrate the system's person and vehicle detection and its visual monitoring interface.

VII.LIMITATIONS AND THREATS TO VALIDITY

In adherence to rigorous scientific integrity, several technical limitations and threats to external validity must be explicitly acknowledged:

1. The prototype was evaluated using an integrated 720p HD laptop webcam (CAM-01) and an Iriun phone camera (CAM-02). Historical telemetry also contains records associated with CAM-03, whose physical configuration is not documented. The evaluation therefore represents a limited prototype setup and does not establish performance in operational border environments or across a range of outdoor surveillance conditions.
2. **Monocular Planar Homography Assumptions:** Homography-based ground-distance and velocity estimation relies on the assumption of a locally planar ground surface. Terrain featuring steep slopes, ditches, or multiple elevation levels may introduce metric distortion and affect the accuracy of estimated distances and velocities. The zone-occupancy decision itself is performed in image pixel coordinates and does not use homography-projected ground coordinates. Stereovision or LiDAR-based sensing could be explored in future work to improve metric estimation in non-planar environments.
3. **Fixed Dwell-Time Threshold Trade-off:** A fixed dwell threshold of 5.0 seconds was selected to eliminate transient crossings and brief boundary touches. However, in high-velocity breach scenarios (e.g., a motorized vehicle crossing a2-meter boundary zone at 50 km/h in < 0.2 seconds), a 5-second dwell requirement would fail to confirm the breach before the target exits camera view. Future work must incorporate velocity-adaptive dwell thresholds.
4. **Tracking Under Heavy Crowd Occlusion:** While the greedy IoU and foot-distance tracker has a measured mean update time of 2.018 ms, it lacks deep appearance ReID feature embeddings. Under dense crowding where multiple targets cross paths and mutually occlude their foot-points for > 2.0 seconds, identity switches or track fragmentation may occur.
5. **Foot-Point Localization Limitations:** Bottom-center foot-point estimation may be unreliable when bounding boxes are truncated or partially occluded. It is also less suitable for seated subjects, such as motorcycle riders, and vehicles, where the bottom-center of the bounding box may not correspond to the actual ground-contact point. These cases can affect zone-occupancy decisions.
6. **Track-Loss Events:** Of the 193 historical operational event records, 45 (23.3%) ended with the LOST_UNCERTAIN status, indicating that the target disappeared or was occluded while still associated with a restricted zone. This reflects a limitation in maintaining continuous target tracking and confirming zone exits.
7. **Evaluation Data and Runtime Stability:** The 193 records were generated during developer-run testing and were not independently annotated against ground truth. They characterize operational event logging, not detection or tracking accuracy. Long-duration uptime, crash frequency, and recovery behaviour were not systematically measured.
8. **VLM Output Evaluation:** The generated incident summaries were not evaluated using a labelled dataset or a systematic factuality assessment. Their accuracy, omission rate, and tendency to produce unsupported descriptions therefore remain unquantified.

VIII. CONCLUSION AND FUTURE WORK

This paper presented SBIDS, a multi-threaded computer vision prototype for automated perimeter monitoring. The system combines frame capture, YOLO11n object detection, greedy object tracking, pixel-space zone monitoring, and a six-state finite state machine to manage intrusion events using a 5.0-second dwell-confirmation threshold. In a controlled geometric simulation involving 36 synthetic boundary positions, foot-point containment avoided the boundary-overhang triggers observed with bounding-box intersection. This result demonstrates geometric disambiguation under the simulated conditions rather than real-world detection accuracy. In a benchmark of 50 saved images on an Intel Core i5-12450HX CPU, the system recorded a mean end-to-end processing latency of 158.94 ± 27.95 ms, corresponding to approximately 6.29 FPS. This throughput was below the configured target of 12 FPS, indicating scope for further performance optimization. Historical telemetry from 193 developer-run operational events provided additional information about system behaviour in the tested environment but does not establish detection accuracy. Overall, the work demonstrates the implementation of a prototype perimeter-monitoring pipeline while identifying areas requiring further evaluation.

IX. STATEMENTS AND DECLARATIONS

9.1 Data and Code Availability

The complete SBIDS source code, model configurations, and experimental benchmarking scripts are maintained within the project repository.

9.2 Ethics and Privacy Statement

Surveillance frames presented in this paper were captured during operational system testing. To protect individual privacy, facial regions of all human subjects in published evidence figures have been blurred and anonymized in accordance with academic research standards.

9.3 Funding Statement

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

9.4 Competing Interests

The authors declare that they have no competing financial or personal interests that could influence the work reported in this paper.

9.5 Author Contributions

Bhuvan K. C.: Conceptualization, software architecture, camera worker pipeline, and manuscript drafting. Karthik Y. M.: Tracking algorithms, geometry engine, and experimental benchmarks. Darshan Lokesh: Security decision engine, email notification infrastructure, and validation.

REFERENCES

[1] A. Hampapur et al., “Smart video surveillance: exploring the concept of multiscale tracking,” IEEE Signal Processing Magazine, vol. 22, no. 2, pp. 38–51, 2005. doi: 10.1109/MSP.2005.1406476.

[2] X. Wang, “Intelligent multi-camera video surveillance: A review,” Pattern Recognition Letters, vol. 34, no. 1, pp. 3–19, 2013. doi: 10.1016/j.patrec.2012.07.005.

[3] G. Jocher and J. Qiu, “Ultralytics YOLO11: Architecture and Performance Documentation,” Ultralytics Inc., 2024. [Online]. Available: https://docs.ultralytics.com/models/yolo11/.

[4] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in Proc. IEEE Int. Conf. Image Processing (ICIP), 2016, pp. 3464–3468. doi: 10.1109/ICIP.2016.7533003.

[5] N. Wojke, A. Bewley, and D. Paulus, “Simple online and realtime tracking with a deep association metric,” in Proc. IEEE Int. Conf. Image Processing (ICIP), 2017, pp. 3645–3649. doi: 10.1109/ICIP.2017.8296962.

[6] Y. Zhang et al., “ByteTrack: Multi-object tracking by associating every detection box,” in Proc. European Conf. Computer Vision (ECCV), 2022, pp. 1–21. doi: 10.1007/978-3-031-20047-2_1.

[7] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, “BoT-SORT: Robust associations multi-pedestrian tracker,” arXiv preprint arXiv:2206.14651, 2022.

[8] Gemini Team, "Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities," Google, 2025. [Online]. Available: https://storage.googleapis.com/deepmind-media/gemini/gemini_v2_5_report.pdf.