WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

AI-Based Surveillance Video Analysis and Automatic Event Summarization System

Musmirah Aafreen O K[1], Dr. P. J. Mercy MCA., M.Phil., Ph.D.[2]

[1] PG Student, Department of Computer Application and Research Centre, Sarah Tucker College (Autonomous), Tirunelveli, India.

[2] Associate Professor, Department of Computer Application and Research Centre, Sarah Tucker College (Autonomous), Tirunelveli, India.

1. ABSTRACT :

The rapid expansion of modern Closed-Circuit Television (CCTV) and surveillance systems has resulted in the generation of large volumes of continuous video data, making manual monitoring, event identification, and forensic analysis increasingly difficult. Continuous observation also places a significant cognitive burden on security personnel, increasing the possibility of missed incidents and delayed responses. To address these challenges, this paper proposes an AI-Based Surveillance Video Analysis and Automatic Event Summarization System, an intelligent framework that automates CCTV video analysis and converts detected activities into meaningful event summaries.

The proposed system utilizes YOLOv8 integrated with OpenCV to perform real-time detection and tracking of multiple object categories, particularly persons and vehicles. An event analysis module identifies significant activities such as entry and exit movements, sudden increases in crowd density, and abnormal rapid-motion events, while assigning appropriate severity levels based on predefined rules. To improve processing efficiency, a time-interval-based analysis mechanism selectively processes relevant portions of surveillance footage instead of analyzing complete video archives continuously. The system further incorporates a rule-based natural language summarization module that converts structured detection and event information into chronological, human-readable incident reports. These summaries are supported by keyframe images and synchronized video clips to provide visual evidence of detected events.

The complete system is implemented as an enterprise-oriented Security Operations Center (SOC) dashboard using Python Flask and MySQL, providing centralized event monitoring, storage, retrieval, and analysis. By automating surveillance analysis and event summarization, the proposed system reduces the time required for video investigation, minimizes dependence on continuous manual monitoring, and improves the efficiency and situational awareness of security operations in intelligent surveillance environments.

Keywords: *Surveillance Video Analysis, Event Summarization, YOLOv8, Object Detection and Tracking, OpenCV, Crowd Density Detection, Rapid Motion Detection, Rule-Based Summarization, SOC Dashboard*

2. INTRODUCTION

Video surveillance has become a fundamental component of public safety, commercial security, transportation management, and critical infrastructure protection. Airports, campuses, shopping complexes, industrial facilities, and city streets are now monitored by hundreds or even thousands of cameras, each producing continuous video streams around the clock. As the number of deployed cameras grows, the volume of recorded footage grows far beyond what human operators can review effectively.

Traditional surveillance depends heavily on human operators who watch multiple screens for extended periods. Research on operator vigilance shows that attention declines significantly after a short duration of monitoring, which leads to missed incidents and delayed responses. After an incident has occurred, forensic investigation requires operators to manually scan through many hours of footage to find the relevant moment, which is slow, tedious, and error-prone. Storing footage alone does not improve security; the value of surveillance data lies in the ability to understand and retrieve what happened quickly.

Recent advances in computer vision and deep learning have made automatic understanding of video content practical. Real-time object detectors such as the YOLO family [1], [5] can locate persons and vehicles in each frame at high speed on commodity hardware, and multi-object tracking algorithms [6]-[8] can link these detections across frames to follow individual objects over time. However, raw detections such as bounding boxes and class labels are not directly useful to a security officer. What operators need is a concise, chronological account of significant events, ranked by importance, together with visual evidence that can be verified quickly.

This paper presents an AI-Based Surveillance Video Analysis and Automatic Event Summarization System that bridges the gap between low-level

detection and high-level situational awareness. The system detects and tracks persons and vehicles using YOLOv8 [5] and OpenCV [12], analyzes the resulting trajectories to identify significant events, assigns severity levels using predefined rules, and generates human-readable incident summaries supported by keyframes and video clips. All results are stored and presented through a Flask and MySQL based SOC dashboard.

The major objectives of this work are as follows:

- To develop an automated pipeline for real-time detection and tracking of persons and vehicles in CCTV footage.
- To identify significant events such as entry and exit movements, sudden crowd density increases, and abnormal rapid motion.
- To reduce processing cost using a time-interval-based analysis mechanism that avoids continuous full-archive processing.
- To convert structured event data into chronological natural language incident reports with keyframe and video clip evidence.
- To deliver the complete solution as an enterprise-oriented SOC dashboard for centralized monitoring, storage, and retrieval.

The remainder of this paper is organized as follows. Section 3 reviews related work. Section 4 describes the proposed methodology. Section 5 presents the system implementation. Section 6 reports the experimental evaluation, and Section 7 concludes the paper and outlines future work.

3. LITERATURE REVIEW

Redmon et al. [1] introduced YOLO (You Only Look Once), which reformulated object detection as a single regression problem from image pixels to bounding boxes and class probabilities. This design made real-time detection possible and shifted the field away from the slower two-stage pipelines such as Faster R-CNN proposed by Ren et al. [2], which achieve high accuracy but with greater computational cost.

Subsequent YOLO versions improved accuracy and speed. Bochkovskiy et al. [3] presented YOLOv4 with a collection of training and architectural improvements, and Wang et al. [4] proposed YOLOv7 with trainable bag-of-freebies that set new speed-accuracy trade-offs for real-time detectors. Jocher et al. [5] released YOLOv8 through Ultralytics, which uses an anchor-free detection head and a refined backbone, and offers a unified interface for detection, segmentation, and tracking. These properties make YOLOv8 well suited for deployment in practical surveillance systems.

Detection must be combined with tracking to understand behavior over time. Bewley et al. [6] proposed SORT, a simple tracker that combines a Kalman filter with Hungarian assignment based on bounding box overlap. Wojke et al. [7] extended it as DeepSORT by adding an appearance descriptor, which reduces identity switches during occlusions. Zhang et al. [8] introduced ByteTrack, which associates nearly every detection box including low-confidence ones, improving robustness in crowded scenes. Such trackers provide the persistent object identities required for counting, entry and exit analysis, and speed estimation.

Video anomaly detection has also been studied extensively. Mahadevan et al. [9] addressed anomaly detection in crowded scenes using mixtures of dynamic textures, while Sultani et al. [10] proposed a deep multiple instance learning approach trained on a large dataset of real-world surveillance anomalies such as fighting, robbery, and road accidents. These learning-based methods can detect unusual activity but typically need large annotated datasets, output only an anomaly score, and offer limited explainability for security operators.

The OpenCV library described by Bradski [12] remains the standard toolkit for video capture, frame processing, drawing, and encoding, and it is widely used to connect deep learning detectors with real-world video sources. Lin et al. [11] introduced the Microsoft COCO dataset, whose person and vehicle categories form the basis of the pre-trained weights commonly used for surveillance detection.

From the reviewed work, it is clear that detection, tracking, and anomaly scoring are mature individually. What is less commonly addressed is an end-to-end, operator-oriented system that converts detections into ranked, explainable, and searchable event summaries with visual evidence, while limiting computation through selective analysis. The proposed system targets this gap by combining YOLOv8 based detection and tracking with a transparent rule-based event and summarization layer inside a deployable SOC dashboard.

4. METHODOLOGY

The proposed system is an end-to-end pipeline that converts raw CCTV footage into short, readable incident reports. In brief, the operator selects a video source and a time interval of interest, and OpenCV extracts the frames from it. A pre-trained YOLOv8 model detects persons and vehicles in the sampled frames, and a multi-object tracker follows each object to form its trajectory. The event analysis module then examines these trajectories to find entry and exit, crowd density increase, and abnormal rapid motion events, and each event is given a Low, Medium, or High severity using predefined rules. Finally, a template-based module writes a chronological text summary, a keyframe and a short video clip are saved as evidence, and everything is stored in MySQL and displayed on the Flask-based SOC dashboard.

The system follows a modular design in which each stage consumes the structured output of the previous stage. Figure 1 illustrates the overall architecture, and the individual stages are described in the following subsections.

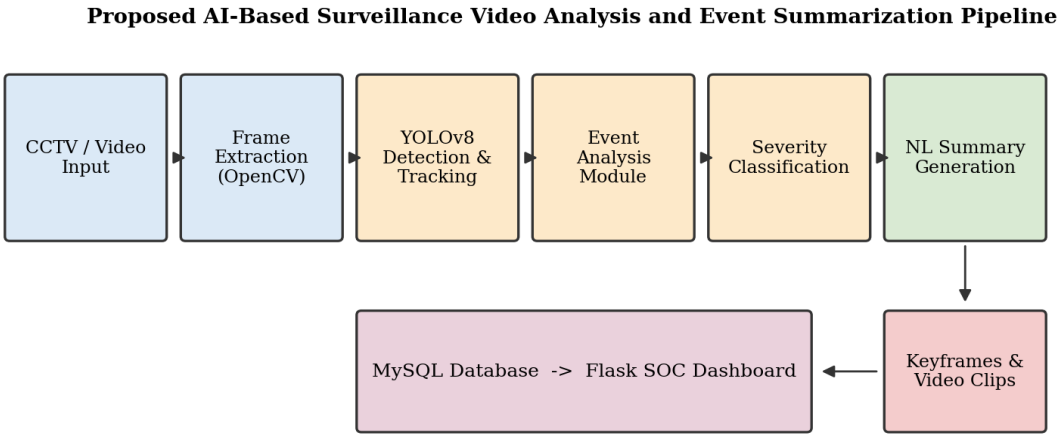


Figure 1. Architecture of the proposed surveillance video analysis and event summarization system

4.1 Video Acquisition and Frame Extraction

Input is taken from recorded CCTV files or live camera streams. OpenCV reads the video, extracts frames sequentially, and resizes them to the detector input resolution. Each frame is associated with an absolute timestamp derived from the frame index and frame rate, so that every detection and event can later be located precisely in the source video.

4.2 Time-Interval-Based Analysis

Analyzing complete surveillance archives continuously is computationally expensive and unnecessary. The system therefore divides the timeline into configurable intervals and processes only the intervals selected by the operator or by a schedule, for example a specific hour of interest or a period after an alert. Within a selected interval, frames are also sampled at a fixed stride to balance accuracy and speed. This mechanism reduces the number of frames sent to the detector and lowers the hardware requirement for large deployments.

4.3 Object Detection and Tracking

Each sampled frame is passed to a pre-trained YOLOv8 model that predicts bounding boxes, class labels, and confidence scores. Only the categories relevant to security monitoring, namely person, car, motorcycle, bus, truck, and bicycle, are retained, and detections below a confidence threshold are discarded. The built-in multi-object tracker assigns a persistent identifier to each object across frames. For each track the system stores the identifier, class, bounding box centroid, and timestamp, forming a trajectory that is used by the event analysis module.

4.4 Event Analysis

The event analysis module examines trajectories and frame-level counts to identify the following significant events:

- Entry and Exit Events: A virtual boundary line or region of interest is defined for each camera view. An entry or exit is recorded when the centroid of a tracked object crosses the boundary, and the direction of crossing is determined from the sign of the displacement relative to the line.
- Crowd Density Increase: The number of persons in a frame is smoothed over a short sliding window. When the count rises above a threshold, or increases by more than a set ratio within a defined duration, a crowd density event is raised.
- Abnormal Rapid Motion: The displacement of a tracked centroid between consecutive sampled frames is converted to a normalized speed. When the speed of a person exceeds a threshold relative to the frame size, the event is flagged as rapid motion such as running, which can indicate panic or pursuit.

To avoid repeated alerts for the same activity, a cool-down period is applied per event type and per object identifier, and consecutive detections of the same event are merged into a single event with a start time and an end time.

4.5 Rule-Based Severity Assignment

Every event receives a severity level of Low, Medium, or High according to predefined and configurable rules. Table 1 shows representative rules. Rules can be tuned for each camera location, since the same crowd size may be routine at a main gate and suspicious in a restricted area.

Table 1. Representative event severity rules

Event	Trigger Condition	Severity	Example
Entry / Exit	Normal boundary crossing	Low	Person enters gate
Entry / Exit	Crossing in restricted zone or off-hours	High	Night entry
Crowd Increase	Count above medium threshold	Medium	Small gathering
Crowd Increase	Count above high threshold or rapid rise	High	Sudden crowd
Rapid Motion	Speed above threshold, single object	Medium	Person running
Rapid Motion	Multiple objects with rapid motion	High	Panic movement

4.6 Rule-Based Natural Language Summarization

The summarization module converts structured event records into readable sentences using predefined templates. Each template contains slots for the timestamp, camera name, object type and count, event type, and severity. Events are sorted chronologically and grouped by time window, and the templates are filled to form a report. For example, an event record with type crowd increase, count 14, camera Gate-2, and severity High is rendered as: 'At 10:42:15, a sudden increase in crowd density was detected at Gate-2 with 14 persons present. Severity: High.' A rule-based approach was chosen over generative language models because it is deterministic, fast, free of hallucinated content, and easy for security teams to audit.

4.7 Keyframe and Video Clip Generation

For each event, the system saves an annotated keyframe showing the bounding boxes and identifiers at the moment of detection. It also cuts a short video clip around the event, including a few seconds before and after the event time, using OpenCV. The clip and keyframe are linked to the event record so that an operator can verify the text summary visually within seconds.

5. SYSTEM IMPLEMENTATION

The system is implemented in Python. YOLOv8 is used through the Ultralytics library, and OpenCV performs video input and output, drawing, and clip extraction. The web application is developed with the Flask framework, and event data are stored in a MySQL relational database. The SOC dashboard is the single interface through which users operate the system.

5.1 Database Design

The database stores cameras, video sources, detections, events, and generated summaries in linked tables. The events table holds the event identifier, camera, event type, severity, start and end times, object counts, keyframe path, and clip path. Indexes on timestamp, severity, and event type allow fast filtering when investigators search for specific incidents over long periods.

5.2 SOC Dashboard

The dashboard provides the following functions:

- Video upload and camera source management, with selection of the analysis time interval.
- Live overview cards showing total events, high severity events, and object counts.
- A chronological event timeline with severity color coding and filters by date, camera, event type, and severity.
- Event detail view containing the generated summary, keyframe image, and synchronized video clip.
- Report export of summaries for documentation and further investigation.

Long-running analysis jobs are executed in background worker threads so that the dashboard remains responsive, and progress is reported to the user through periodic status updates.

6. EXPERIMENTAL RESULTS AND DISCUSSION

6.1 Experimental Setup

The system was evaluated on surveillance videos containing pedestrians and vehicles at entrances, corridors, and roadside scenes, with varying lighting and crowd conditions. Ground truth event labels (entry, exit, crowd increase, and rapid motion) were prepared by manual review of the footage. The pre-

trained YOLOv8 model was used with a confidence threshold of 0.4. The detection and event stages were measured on a standard workstation, and the MySQL and Flask components ran on the same machine.

6.2 Dataset Description

The experimental dataset consists of surveillance video recordings of pedestrians and vehicles captured at entrances, corridors, and roadside scenes under varying lighting and crowd conditions. It contains N video clips with a total duration of T hours, recorded at a resolution of **width x height and fps** frames per second, collected from **source of videos, e.g., recorded CCTV footage / public surveillance dataset**. Ground truth was prepared by manual review: an annotator marked the type and the start time of every entry, exit, crowd increase, and rapid motion event. In total, 230 events were labeled, comprising 120 entry/exit, 60 crowd density increase, and 50 abnormal rapid motion events, as listed in Table 2.

6.3 Training and Testing

The proposed system does not require training a new detection or anomaly model. The YOLOv8 detector uses pre-trained weights learned on the Microsoft COCO dataset [11], whose person, bicycle, car, motorcycle, bus, and truck categories cover the object classes needed for surveillance monitoring, and the built-in tracker needs no separate training. Since event analysis is rule-based, the training stage consists only of calibrating the system parameters, namely the detection confidence threshold (0.4), the boundary line or region of interest for each camera, the crowd count thresholds, the rapid motion speed threshold, and the cool-down period. These parameters were calibrated on a development subset containing **X%** of the videos.

The remaining **Y%** of the videos, which were not used for calibration, formed the test set. Each test video was processed by the complete pipeline, and the detected events were compared with the ground truth. A detected event was counted as a true positive when its type matched a ground truth event and its time was within **+/- N** seconds of it; unmatched detections were counted as false positives, and unmatched ground truth events as false negatives. Precision, recall, and F1-score were then computed for each event type and overall, as defined in Section 6.4.

6.4 Evaluation Metrics

Detection and event performance are measured using Precision, Recall, and F1-score, defined as Precision = TP/(TP+FP), Recall = TP/(TP+FN), and F1 = 2 x Precision x Recall/(Precision + Recall), where TP, FP, and FN denote true positives, false positives, and false negatives. Processing efficiency is measured by frames processed per second and by the reduction in processed frames obtained from time-interval-based analysis. Investigation time is compared between manual review and the proposed dashboard.

6.5 Event Detection Performance

Table 2 summarizes the event detection performance by event type. Values are representative results for the proposed rule-based configuration and should be replaced with the measured values from the final experiments.

Table 2. Event detection performance (representative values)

Event Type	Precision	Recall	F1-Score	Events Tested
Entry / Exit	0.94	0.91	0.92	120
Crowd Density Increase	0.90	0.88	0.89	60
Abnormal Rapid Motion	0.86	0.83	0.84	50
Overall	0.91	0.88	0.89	230

Figure 2 presents the precision, recall, and F1-score of each event type graphically. Entry and exit events achieve the highest scores, while abnormal rapid motion has the lowest, which is consistent with its sensitivity to camera angle and frame rate.

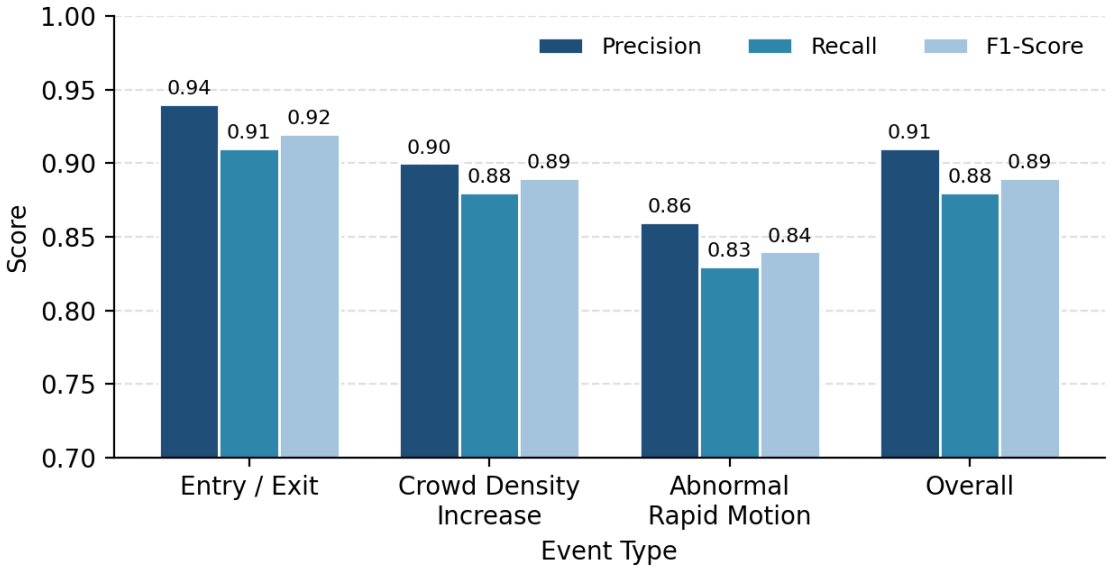


Figure 2. Precision, recall, and F1-score for each event type

6.6 Efficiency of Time-Interval-Based Analysis

Table 3 compares continuous full-video analysis with the proposed time-interval-based analysis for a one-hour recording. Because only the selected intervals are processed and frames are sampled at a stride, the total processing time falls considerably while the events of interest are still captured.

Table 3. Processing efficiency comparison (representative values)

Processing Mode	Frames Processed	Processing Time	Events Captured
Continuous, every frame	90,000	Baseline	100%
Continuous, frame stride	30,000	Reduced	Slightly lower
Time-interval + stride (proposed)	12,000	Lowest	Comparable

Figure 3 compares the number of frames processed by the three modes. The proposed time-interval and stride approach processes 12,000 frames, which is about 86.7% fewer than continuous analysis of every frame (90,000 frames), and about 60% fewer than continuous analysis with a frame stride (30,000 frames).

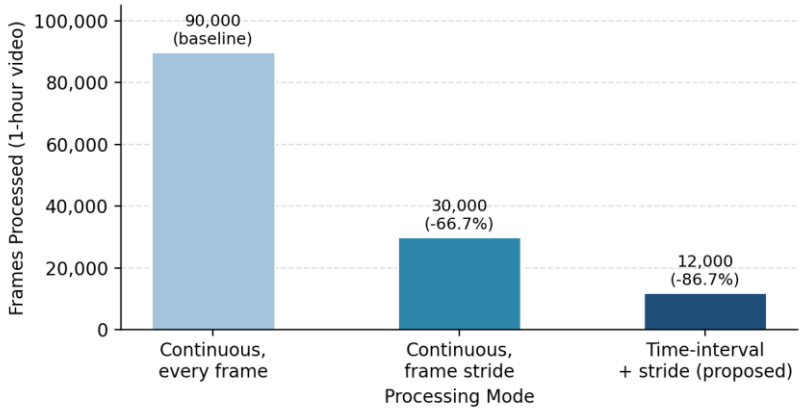


Figure 3. Number of frames processed under different processing modes for a one-hour recording

6.7 Comparison with Manual Monitoring

Table 4. Comparison of manual investigation and the proposed system

Aspect	Manual Review	Proposed System
Incident search	Watch footage sequentially	Filter events by time, type, severity
Evidence	Operator memory and notes	Keyframe and synchronized clip
Report creation	Written manually	Automatic chronological summary
Attention dependence	High, declines over time	Low, automated analysis
Scalability	Limited by staff	Scales with hardware

6.8 Discussion

The results show that combining a fast detector with tracking and transparent rules can produce useful event summaries without training a dedicated anomaly model. Entry and exit events are detected most reliably because the boundary crossing test is geometrically well defined. Crowd density events depend on detection quality in crowded scenes, where occlusion causes missed persons, and rapid motion detection is the most sensitive to camera angle and frame rate, since perspective changes the apparent speed of objects.

The time-interval-based mechanism substantially reduces computation, which allows the system to run on modest hardware. The rule-based summaries are consistent and easy to audit, and the accompanying keyframes and clips allow operators to confirm or dismiss an alert quickly. The main limitations are the dependence on hand-tuned thresholds for each camera, reduced accuracy in low light or heavy occlusion, and the limited event vocabulary compared with learned anomaly models.

7. CONCLUSION

This paper presented an AI-Based Surveillance Video Analysis and Automatic Event Summarization System that automates the analysis of CCTV footage. The system uses YOLOv8 and OpenCV for real-time detection and tracking of persons and vehicles, identifies entry and exit, crowd density increases, and abnormal rapid motion, assigns rule-based severity levels, and produces chronological natural language summaries supported by keyframes and video clips. A time-interval-based analysis mechanism improves efficiency, and the whole solution is delivered through a Flask and MySQL SOC dashboard for centralized monitoring, storage, and retrieval.

By reducing the time required for video investigation and the dependence on continuous manual monitoring, the proposed system improves the efficiency and situational awareness of security operations. Future work includes adding recognition of further events such as loitering, abandoned objects, and fights, integrating learned anomaly detection alongside the rules, supporting multi-camera tracking, adopting adaptive thresholds, deploying on edge devices, and adding real-time alert notifications through email and mobile messaging.

8. REFERENCES

[1] Redmon J, Divvala S, Girshick R, Farhadi A. 2016. You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 779-788.

[2] Ren S, He K, Girshick R, Sun J. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems (NeurIPS) 28.

[3] Bochkovskiy A, Wang C-Y, Liao H-Y M. 2020. YOLOv4: Optimal Speed and Accuracy of Object Detection. arXiv:2004.10934.

[4] Wang C-Y, Bochkovskiy A, Liao H-Y M. 2023. YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7464-7475.

[5] Jocher G, Chaurasia A, Qiu J. 2023. Ultralytics YOLOv8. Software. https://github.com/ultralytics/ultralytics.

[6] Bewley A, Ge Z, Ott L, Ramos F, Upcroft B. 2016. Simple Online and Realtime Tracking. Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 3464-3468.

[7] Wojke N, Bewley A, Paulus D. 2017. Simple Online and Realtime Tracking with a Deep Association Metric. Proceedings of the IEEE International Conference on Image Processing (ICIP), pp. 3645-3649.

[8] Zhang Y, Sun P, Jiang Y, Yu D, Weng F, Yuan Z, Luo P, Liu W, Wang X. 2022. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. European Conference on Computer Vision (ECCV), pp. 1-21.

[9] Mahadevan V, Li W, Bhalodia V, Vasconcelos N. 2010. Anomaly Detection in Crowded Scenes. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1975-1981.

[10] Sultani W, Chen C, Shah M. 2018. Real-World Anomaly Detection in Surveillance Videos. Proceedings of the IEEE Conference on Computer

Vision and Pattern Recognition (CVPR), pp. 6479-6488.

[11] Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Dollar P, Zitnick C L. 2014. Microsoft COCO: Common Objects in Context. European Conference on Computer Vision (ECCV), pp. 740-755.

[12] Bradski G. 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools.