International Journal of Innovative Research in Engineering and Management (IJIREM)
ISSN (Online): 2350-0557, Volume 12, Issue 6, December 2025
DOI: https://doi.org/10.55524/ijirem.2025.12.6.19
Article ID IJIR3150, Pages 104-110
www.ijirem.org

Real-Time Multi-Input Object Tracking and Speed Estimation via YOLOv8 and Kalman Filters for Smart Environments

G. HariHaran[1], and Dr. Gunjan Singh[2]

[1]MCA Scholar, Amity Institute of Information Technology, Amity University, Gurugram, Haryana, India
[2]Assistant Professor, Department of Computer Science and Engineering, Amity University, Gurugram, Haryana, India

Correspondence should be addressed to G. HariHaran; gghariharan200214@gmail.com

Received: 16 November 2025 Revised: 30 November 2025 Accepted: 16 December 2025

ABSTRACT- This work presents a real-time, multi-input, multi-object tracking, and speed estimation system that uses YOLOv8 deep learning detector fused with Kalman Filter-based predictive tracking. By coupling cutting-edge detection accuracy with strong motion prediction, the article deals with problems of occlusions, noisy detections, variable motion, and heterogeneous video sources that are typical of smart environments. Kalman Filters are implemented to allow stable identity assignment, trajectory smoothing, and frame-consistent speed estimation while YOLOv8 is the primary detection tool due to its anchor-free, multi-scale design, and rapid inference. The multi-video-stream-compatible system, which can also be used in the area of intelligent surveillance, traffic monitoring, and automated analytics by CCTV, webcams, recorded footage, and image uploads, has different potential usages. Encrypted logging is present to guarantee the security of Meta data storage in privacy-sensitive environments. The results of the research presented in the paper reveal the potential of the system to carry out object identity retention, multi-object trajectory management, and on-the-fly calculation of motion parameters, thus providing a scalable and flexible platform for smart cities and industrial applications. Such a system is a significant move forward to the eventual establishment of a real-time tracking system framework that is resilient, modular, and respectful of privacy, hence making a substantial incremental development towards the evolution of these systems in dynamic smart environments.

KEYWORDS: YOLOv8, Kalman Filter, Multi-object Tracking, Smart Environments, Speed Estimation.

I. INTRODUCTION

Deep learning techniques have been the major driver of these changes for the last ten years and have made it an easy task for machines to be sensitive to changing situations at a previously unattainable level. The first systems such as Fast R-CNN [6], Faster R-CNN [22], and later enhancements to the R-CNN family [23] were the initial steps for detection of objects through region-based architectures. At the same time, SSD [13] and anchor-free detection with feature pyramids [32] were some of the parallel advancements that led to quicker and more scalable approaches. Object detection and multi-object tracking have been the key factors that are mainly used in systems such as intelligent stewardship, independent vehicles, and actual-time conclusion-making systems. The changes have been very rapid in the last ten years with deep learning as the major power. Machines are now able to understand the changing scesnarios with an accuracy that was not possible before. These innovations were a condition for single-shot architectures like YOLO that radically changed the idea of real-time detection by providing a compromise between accuracy and speed Object detection and multi-object tracking have been the major elements to facilitate systems like smart surveillance, self-driving cars, and instant decision-making systems. Deep learning techniques which have been the major driver of these changes for the last ten years have made sensitivity to changing situations at a previously unattainable level an easy task for machines. The first systems such as Fast R-CNN [6], Faster R-CNN [22], and later enhancements to the R-CNN family [23] were the initial steps for detection of objects through region-based architectures. At the same time, SSD [13] and anchor-free detection with feature pyramids [32] were some of the parallel advancements that led to quicker and more scalable approaches. The invention of YOLO [19], YOLOv3 [21], and YOLOv4 [3] changed the face of object detection by proving that fast inference could be done with minimal loss of accuracy. Subsequent models such as YOLOv5 [10], YOLOv7 [29], and YOLOv8 [9], [12], [27] were able to attain even higher neural network efficiency mostly by architectural changes, augmentation strategy, and a trainable optimization component. As a consequence of these versions, the significance of YOLO-based systems has been escalated to various real-time sectors, for instance, independent navigation, smart city cover, and trade analytics [15], [24], [30]. The constant development YOLO that's fits the community's choice of models that are able to deal with high-resolution inputs at very low latencies, which is a condition necessary for the use of safety-critical areas, for example, autonomous vehicles. However, flawless object detection alone is still not sufficient for a system that needs to track the same objects in different frames. Multi-object tracking (MOT) involves the integration of detection with temporal filtering and identity assignment. Techniques such as SORT [1] provided a quick way of associating detections with the classical Kalman filter [11] through linking. The addition of appearance features in this system by Deep SORT thereby enabled more feasible tracking during occlusions and when noise was present [1], [17]. Later modifications bring in adaptive or probabilistic Kalman filters [4], [28], [31], thus providing more stable motion estimation even for complex scenarios. Such hybrids have turned indispensable in scenarios with multiple agents such as pedestrian monitoring, vehicle tracking, and traffic flow analysis. The Kalman filter remains to be one of the primary tools for the estimation of the state variables, mainly because of its mathematical efficiency and capability to predict. The theoretical bases of the Kalman filter are quite clear and well established [2], [5], and it is still widely used in the systems of tracking [16], [25], [26]. Coherent with this, recent publications emphasize that the fusion of YOLO models with Kalman filters can lead to improved tracking performance in smart city infrastructures [15], autonomous driving [18], and vehicle motion state estimation using multi-modal sensor fusion [7]. Using sensor fusion methods, it is possible to fully utilize the advantages of LiDAR, radar, and a monocular camera while also lessening the disadvantages of each modality [8], [28]. As a result, this maintains the stability of the system in difficult situations that might occur in nature. To handle the increased computational requirements, real-time systems must also perform detection, tracking, and filtering in a single pipeline. Latest research indicates that YOLOv8 in conjunction with the improved Kalman models and identity association algorithms to model complex object dynamics is a highly potent approach [24], [31]. The capability of such pipelines to manage occlusion, fast-moving objects, and changing illumination conditions is what makes them very useful in practice. In addition, the availability of open-source tools and documentation [9], [20] has been a major facilitator of the implementation of these models in the industry and academia. However, one of the most significant issues to be solved is the problem of maintaining stable performance in dynamic scenes, i.e., changing scenes, at a high level. The main reasons leading to the deterioration of tracking quality are: environmental noise, sudden changes in motion, uncertainty of sensors, and identity switching. To this end, the use of advanced deep-learning detectors in combination with filters grounded in mathematics is necessary. The body of research that has been done so far, most convincing, shows that the combination of YOLO-based models with Kalman filtering methods gives the best results in terms of stability of the tracked objects in real-time, the solution being also scalable and capable of complex environments [31], [27], [18]. As the research area is gradually approaching the development of the next-generation intelligent transportation systems and autonomous robotics, the hybrid architectures of such a kind will still be there leading the way.

II. LITERATURE REVIEW

Hajinazar et al [7], Wang et al. [29], Zheng et al.[31], Recent study have been largely directed towards improving YOLOv8's detection accuracy and the incorporation of state-of-the-art Kalman filter variants for dealing with occlusion, noise, and other similar situations in real-time multi-object tracking. Moreover, advancements in multi-sensor data fusion have allowed for more accurate speed calculations in smart environments.

Sheng et al. [24], Varghese [27] Farag et al [4] research comparisons point out the leading performance of YOLOv8 in contrast to older YOLO versions, mainly attributing its precision enhancements in changing scenarios to the multi-scale and anchor-free detection innovations. The usage of Kalman filter for stabilizing predictions in the covering of dropped detections has been rediscovered and emphasized for traffic and surveillance technologies.

Most of the recent research have been documented to have a surge in hybrid approaches that combine deep learning detection with probabilistic filters for trajectory prediction. Novel concepts that synergize Kalman filters with YOLOv8 are paving the way for the solutions of those challenges which exist in the multi-camera setups and also in the varying frame rates [30], [15].

Wang et al [29] studied attempts intense on the modification of YOLOv7 and its source architectures to suit the requirements of embedded systems and edge computing, thereby increasing the possibility of the application of these methods in resource-constrained smart city devices. The trade-off between computational speed and detection robustness is still a very important aspect.

Farahi et al**.** [4] studied shows Kalman filtering methods evolved through extended and unscented versions to effectively handle nonlinearities and measurement noise in object tracking, thus working in harmony with deep learning-based detectors. Techniques for multi-modal data integration became popular for enhancing detection confidence.

The research formulates that efficiency gains were achieved by the introduction of end-to-end architectures that combined the detection and tracking phases in a single framework. In particular, YOLOv3 was extensively examined for its ability to perform in real-time, which made it suitable for applications such as traffic monitoring [21].

Liu et al [14] studied found that focus was placed on enhance choose-hit sensor to effectively operate on devices embedded for self-propelled operation. Cultivate in non-maximum discipline formula enhanced discovery robustness.

A Research study formulate a machine learning community explored anchor-free detectors and new backbone networks to extract features more efficiently, contributing foundational improvements later adopted by YOLOv8 and similar models[32].

Ren et al. [23] studied says that early days, object detection research mainly leaned on the advancements of improved Region-based CNN (R-CNN) variants. The studies in this domain made significant progress in multi-scale feature extraction as well as in transfer learning techniques. The basic concepts that were built through this work have been very instrumental in the development of the efficient, single-stage detectors like YOLO which came later.

Liu et al [13] research study shows that seminal SSD method proposed fast, accurate single-shot detection, an important predecessor to YOLO's architecture, setting the stage for real-time advances.

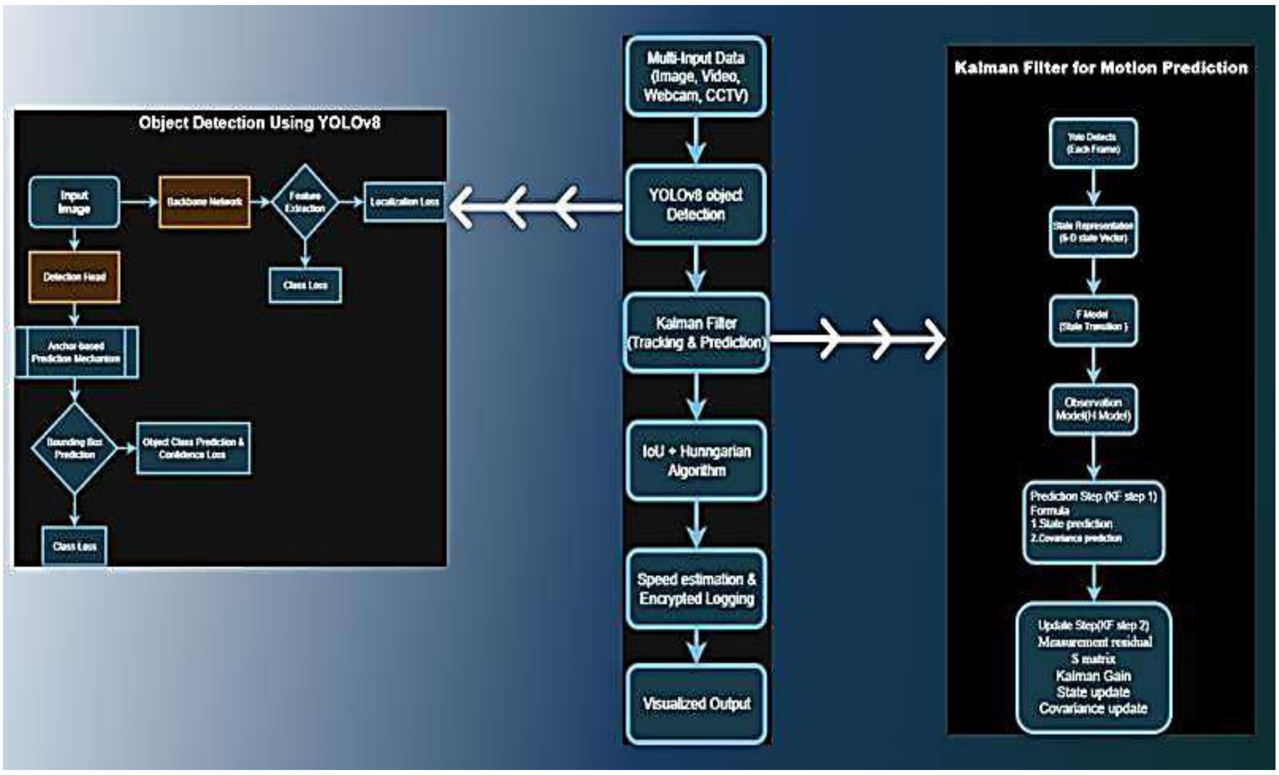
Ren et al. [22] studied faster R-CNN introduced region proposal networks, dramatically improving detection accuracy and becoming a benchmark framework influencing subsequent real-time detector designs.

III. OBJECTIVE

- Build a real-time multi-input system that can handle various video inputs (images, pre-recorded videos, webcams, and RTSP/CCTV feeds) simultaneously for object detection and tracking.
- Combine YOLOv8 for quick and precise object identification in dynamic smart environments.
- Use Kalman Filter-based predictive tracking to support fixed object identities, handle occlusions, and deliver smooth trajectories.
- Allow real-time motion monitoring to include the mapping of the movement, estimation of the direction, and calculation of the speed.
- Create a modular, scalable design that can be used across different areas such as security, automation, traffic monitoring, and smart city applications.
- Provide data security by using encrypted logging methods for the storage of sensitive tracking metadata.

IV. METHODOLOGY

A. Overall System Architecture



Object Detection Using YOLOv8
Input Image
Backbone Network
Feature Extraction
Localization Loss
Detection Head
Class Loss
Anchor-based Prediction Mechanism
Bounding Box Prediction
Object Class Prediction & Confidence Loss
Class Loss
Multi-Input Data (Image, Video, Webcam, CCTV)
YOLOv8 object Detection
Kalman Filter (Tracking & Prediction)
IoU + Hunngarian Algorithm
Speed estimation & Encrypted Logging
Visualized Output
Kalman Filter for Motion Prediction
Yolo Detects (Each Frame)
State Representation (6-D state Vector)
F Model (State Transition)
Observation Model(H Model)
Prediction Step (KF step 1) Formula 1.State prediction 2.Covariance prediction
Update Step(KF step 2) Measurement residual S matrix Kalman Gain State update Covariance update

Figure 1: Proposed System Architecture

The Figure 1 shows the proposed architecture system aims to be a streaming, multi-object identification and tracking mechanism that can work with different kinds of visual data and can generate outputs that are not only labeled but also financially detailed with movement, statistical, and safe metadata logging [9]. The system combines a YOLOv8-based detector with a Kalman filter tracker and a Magyars routine-based data union for subsequent motion and speed analysis and encrypted logging module [1], [11].

B. Input Handling Module

The system is designed to recognize up to four different types of input streams: single images, video files that have been recorded in advance, live webcam feeds, and RTSP streams like CCTV cameras [9]. Each input is decoded to RGB frames of a fixed spatial resolution and frame rate, and these frames are then sent to the detection and tracking pipeline one by one. Such a modular input configuration makes it possible to contain the different requirements of deployment resulting from various surveillance and monitoring scenarios.

C. Object Detection on YOLOv8

Objects are found using the Ultralytics YOLOv8 network, which is configured for real-time inference on the target hardware [9]. In each frame, the detector is provided with the whole image, and the latter is processed through a convolutional backbone and feature aggregation neck to be represented at several scales [1]. The detection head locates class-specific objectless scores and bounding boxes that are given non-maximum suppression for post-processing to get the set of detections {d_j} with the coordinates, confidence, and class label [9].
Each discovery is defined as

$$d_j = [x_{1j}, y_{1j}, x_{2j}, y_{2j}, s_j, c_j],$$

Whereas (x_1j, y_1j) and (x_2j, y_2j) denote the top-left and bottom-right bind of the limitation bin, s_j is the assurance score, and c_j is the forecast item level [9].

D. Kalman Filter–Based Motion Model

To visually follow the very same items over time and also be in a position to manage the instances where detections are missed or are noisy, each active track in the system is equipped with a linear Kalman filter [11]. The state vector is the 2D encircling locker center, its area, and the velocity of the center at the moment.

$$\mathbf{x}_t = [c_x, c_y, w, h, v_x, v_y]^T,$$

Where c_x, c_y are the coordinates of the centroid of the bounding box, w, h are the width and height, and v_x, v_y are the respective horizontal and erect rate [11]. This six-dimensional state representation makes it possible for the system to follow the position as well as the movement speed, thus allowing it to provide a smooth estimate of the trajectory even when occlusions are temporary.

E. State Transition Model

Inter-frame motion is modeled using a constant-velocity assumption between consecutive frames [11]. The state transition is defined as

$$\mathbf{x}_{t+1} = F\mathbf{x}_t + \mathbf{q}_t,$$

With the transition matrix

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

And process noise q_t~N (0, Q) capturing small unmolded accelerations [11].

F. Observation Model

YOLOv8 provides direct measurements of the bounding-box geometry in each frame [1]. After converting the detector output to centroid form, the measurement vector is

$$\mathbf{z}_t = [c_x, c_y, w, h]^T,$$

And is related to the latent state through a linear observation model

$$\mathbf{z}_t = H\mathbf{x}_t + \mathbf{r}_t,$$

where,

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix},$$

And $\mathbf{r}_t \sim \mathcal{N}(0, R)$ denotes measurement noise [11].

G. Kalman Prediction and Update Steps

For each frame, every active track undergoes a predict–update cycle [11].

Prediction Step

The state and error covariance are first propagated using the motion model:

$$\hat{\mathbf{x}}_{t|t-1} = F\mathbf{x}_{t-1|t-1}, P_{t|t-1} = FP_{t-1|t-1}F^T + Q,$$

Basically, it is a method that allows data association to use a predicted bounding box from the Kalman filter even when there is no detection at the current time [11]. With this prediction method, the system can still follow the same object in different frames even if for a short while the detection fails.

Update Step

When a finding is matched to a track, the filter incorporates the measurement through the following sub-steps [11]:

Innovation (residual):

$$\mathrm{y}_t = \mathrm{z}_t - R\hat{\mathrm{x}}_{t|t-1},$$

Innovation covariance:

$$A_t = KP_{t|t-1}H^T + Q,$$

Kalman gain:

$$P_t = K_{t|t-1}H^T S_t^{-1},$$

Updated state and covariance:

$$\mathrm{yx}_{t|t} = \mathrm{y}\hat{\mathrm{x}}_{t|t-1} + K_t\mathrm{y}_t, P_{t|t} = (I - K_tH)P_{t|t-1}.$$

This mechanism smooths trajectories, mitigates jitter, and bridges short-term occlusions or missed detections [11].

H. Track–Detection Association

Assignments between predicted tracks and new detections are computed using an Intersection-over-Union (IoU)–based cost matrix and the Hungarian algorithm [11].

I. IoU Cost Computation

For each predicted bounding box b_i and detection d_j, the IoU is defined as

$$\mathrm{IoU}(\mathrm{a}b_i, d_j) = \frac{\mathrm{area}(ab_i \cap d_j)}{\mathrm{area}(ab_i \cup d_j)}.$$

IoU scores create a similarity matrix where the rows represent tracks and the columns detections [1]. As the Hungarian algorithm is used for minimizing the overall cost, the similarity matrix is changed into a cost matrix:

$$\mathrm{cost}_{ij} = 1 - \mathrm{IoU}(\mathrm{a}b_i, d_j).$$

J. Hungarian Assignment and Gating

The Hungarian algorithm is then used on the cost matrix to find an optimal one-to-one mapping between tracks and detections [1]. An IoU threshold τ_"IoU" (e.g., 0.3) is used to reject implausible matches so that pairs with "IoU"<τ_"IoU" are discarded and considered as unmatched tracks and detections [11]. Matched tracks are propagated through the Kalman filter, their identities are kept, and their missed-detection counters are reset [11]. Unmatched detections create new tracks, whereas unmatched tracks become lost and are removed if they have not been seen for a certain number of frames [1].

K. Track Management

A dedicated track manager is basically the one who keeps different stages of the life of all the objects [1]. Tracks freshly made from high-confidence detections are getting the corresponding centroid and bounding-box size as well as zero initial velocity are being initialized [11]. In every frame, matched tracks are getting Kalman updates, unmatched tracks only do prediction, and tracks whose missed-frame counter surpasses a predetermined threshold are ended so as not to spoil the ID fragmentation and to avoid the accumulation of stale trajectories [1].

L. Trajectory and Direction Analysis

The system keeps the track of the historical centroids {(c_x^t,c_y^t)} for each confirmed track over its duration [11]. The trajectories are shown as polylines drawn on the image plane, and the direction of movement is inferred from the change of recent positions (for instance, the vector from the first to the last k frames, or a smoothed running average can be used) [1]. Such a depiction allows the qualitative examination of the motion patterns and quantitative measures like entry–exit statistics or predominant flow directions [11].

M. Speed Estimation

The speed of the object is derived from the temporal derivative of the centroid position in pixel units and after that, it is converted to the physical units by using the pixel-to-meter calibration factor [11]. Given a frame rate f and a calibration factor α meters per pixel, the instantaneous speed can be calculated as and later it can be converted to kilometers per hour using v_"km/h" =3.6 v_"m/s" [11]. The speeds are thus associated with path IDs and level, which allows rank-wise motion statistics and safety inspection in control scenarios [1].

$$v_{\mathrm{px}} = \frac{\sqrt{(c_x^t - c_x^{t-1})^2 + (c_y^t - c_y^{t-1})^2}}{1/f}, v_{\mathrm{m/s}} = \alpha\, v_{\mathrm{px}},$$

N. Secure Logging Module

To ensure confidentiality and the integrity of the tracking metadata, the system employs an encrypted logging module [9]. With each processed frame, the logger gets and logs the time, the index of the frame, the count of detections, the count of active tracks, the object classes, and the instantaneous speed estimates per track [9] . These organized logs are turned into a string format and encrypted by symmetric encryption using a secret key that is stored in a secure place (for instance, security/encryption/key. Key), and the resulting cipher text is appended to a single combined file (e.g., logs/secure logs.enc) [9] .The architecture is in accordance with secure logging norms where the sensitive surveillance data are only stored in an encrypted form thus lessening the chances of unauthorized disclosure while at the same time retaining the capability of offline forensic analysis to be done after they have been decrypted by a person who has the right authorization [9].

O. Output Generation

The system can operate both offline and online output modes [9]. In the case of an image, the last annotated frame is stored as a still image with the graphical elements drawn on the image, such as bounding boxes, unique track IDs, trajectory traces, direction arrows, and speed estimates [1]. In the case of a video, webcam, and RTSP inputs, the annotation is performed on each frame similarly and the frames are then saved to an output video stream or file, thus, enabling the live multi-object motion tracking to be later visualized and the identity-consistent trajectories along with the associated quantitative data to be viewed during the playback [9].

V. RESULTS

A. System outputs & behaviors

The pipeline can take in images, prerecorded videos, live webcams, and RTSP/CCTV streams, standardizes them to a fixed resolution/frame rate, and processes them one by one. The detector–tracker stack outputs the annotated frames or videos with the class-labeled boxes, the stable track IDs, the trajectory polylines, the direction arrows, and the per-track speed readouts. Also, it stores the per-frame metadata (timestamps, track counts/classes, and speed estimates) in one encrypted file concurrently.

B. Detection and identity continuity

As YOLOv8 gives class-specific detections for each frame, the ones that are NMSed are linked to the predicted tracks via an IoU-based cost matrix and the Hungarian algorithm. Each track uses a Kalman filter (constant-velocity model over centroid, size, and velocities) to remove the noise, occlude short occlusions, and help track continuation if detections are temporarily absent. Tracks arise from high-confidence detections and are dropped after the allowed missed-frame thresholds to prevent ID fragmentation.

C. Motion analytics

Once the tracks are confirmed; the system keeps on recording centroid histories to visually present trajectories and to calculate the general direction of the movement. The momentary speed is obtained from the temporal derivatives (pixels/sec) and changed into real units (for example, km/h) by using a pixel-to-meter calibration factor given by the user. Motion statistics for each class can be obtained from these per-track speeds.

D. Security & logging

A safe logging element takes the structured frame-by-frame records, converts them into a format suitable for storage, and appends the encrypted text to a merged encrypted file, with keys saved in a secure location thus enabling subsequent, authorized forensic analysis and at the same time, reducing the risk of revealing the sensitive surveillance data.

E. Note on quantitative results

The methodology outlines the architecture and algorithms but does not present empirical metrics (e.g., FPS, mAP, MOTA/IDF1, ID switches, latency). Consequently, any numeric performance is a function of hardware, model size, scene content, and configuration.

VI. DISCUSSION

A. Strengths

Modular inputs and real-time orientation: One system can handle images, files, webcams, and RTSP streams in a very flexible way; it is a frame-by-frame design that is targeted at real-time use. Robust short-term tracking: Identity stability is provided by SORT-style Kalman prediction + Hungarian association with IoU gating through brief occlusions and detection dropouts while at the same time, limiting implausible matches via a tunable threshold. Actionable analytics: on the fly trajectory, direction, and speed estimation can be used directly from the raw tracks thus, these are understandable movement summaries that can be used for safety or flow-analysis dashboards. Security by design: privacy and leakage risks are reduced in surveillance contexts by encrypted, append-only logging, thus, post-hoc auditability is preserved.

B. Limitations & risks

Detector dependence: tracking quality is limited by YOLOv8 detection performance. In other words, if detection of small objects, motion blur, night scenes, or heavy occlusion is not done properly, then tracking will also be affected. Besides that, incorrect detections can lead to ID switches that may further increase. Simple motion prior: a constant-velocity Kalman model may sometimes predict directions of abrupt maneuvers, perspective changes, or camera motion incorrectly due to non-linear or scene-aware priors that could make it more robust. Association sensitivity: IoU-only costs are prone to failure when boxes overlap weakly (e.g., fast motion, scale changes). Without appearance cues (re-ID embeddings), tracks of similar objects may swap. Speed calibration: speeds in physical units require not only a correct pixel-to-meter factor but also a stable frame timing. If the camera is tilted or if there is lens distortion, then the speeds can be biased unless these are corrected. Operational considerations: RTSP jitter, clock skew, and storage/key management for encrypted logs need to be handled properly to ensure that there are no data gaps or that keys are not lost.

C. Recommended Enhancements

Appearance-aware tracking: upgrade to DeepSORT/ByteTrack-style association with re-ID features or confidence-aware matching to lower ID switches in crowded scenes. Camera-motion handling: integrate background motion compensation (homography/optical flow) for moving cameras, or use world-plane mapping to stabilize trajectories. Richer motion models: employ EKF/UKF or CV+CA (constant acceleration) switching models; consider learnable motion priors. Calibration & geometry: add an easy field-calibration tool (checkerboard/known span) and perspective correction for more accurate speeds. Evaluation suite: benchmark on standard MOT datasets and report mAP, MOTA, IDF1, IDS, HOTA, latency, FPS; include ablations over IoU threshold, NMS, and tracker parameters. Privacy program: define key rotation, access control, and retention policies for the encrypted logs; support selective decryption by track/time for audits.

VII. CONCLUSION

The said system provides a fully functional, an end-to-end pipeline that is capable of location, following, and understanding the movement of objects in different types of videos sources while also being able to keep the data confidential by means of encrypted logging. The employment of YOLOv8 detection, Kalman prediction, and Hungarian association by the system is a neat tradeoff of correctness, speed, and simplicity of implementation, hence identity-consistent trajectories, orientations, and per-track speed estimates can be generated for real-time monitoring and analytics. The significant next-door steps to the project are to (1) leverage appearance features and better motion priors for fewer ID switches, (2) establish calibration and evaluation procedures for measuring accuracy and throughput, and (3) enhance operational privacy and key-management workflows. Together, these steps would elevate the framework from a strong real-time baseline to a fully certified, privacy-aware solution ready for large-scale deployment.

CONFLICTS OF INTEREST

The authors declare that they have no conflicts of interest.

REFERENCES

[1] Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, “Simple online and realtime tracking,” in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, Phoenix, AZ, USA, 2016, pp. 3464–3468. Available from: https://doi.org/10.1109/ICIP.2016.7533003

[2] Y. Bar-Shalom and X. R. Li, *Estimation and Tracking: Principles, Techniques, and Software*. Norwood, MA, USA: Artech House, 1993. Available from: https://cir.nii.ac.jp/crid/1971149384843963944

[3] Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal speed and accuracy of object detection,” *arXiv preprint*, arXiv:2004.10934, 2020. Available from: https://doi.org/10.48550/arXiv.2004.10934

[4] Farahi, H. Arefi, and M. Abedini, “Probabilistic Kalman filter for moving object tracking,” *J. Visual Commun. Image Represent.*, vol. 67, p. 102739, 2020. Available from: https://doi.org/10.1016/j.image.2019.115751

[5] M. S. Grewal and A. P. Andrews, *Kalman Filtering: Theory and Practice Using MATLAB*, 2nd ed. New York, NY, USA: Wiley-Interscience, 2001.

[6] R. Girshick, “Fast R-CNN,” in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, Santiago, Chile, 2015, pp. 1440–1448.

[7] S. Hajinazar *et al.*, “Multi-modal sensor fusion for highly accurate vehicle motion state estimation,” *Sensors*, vol. 23, no. 14, p. 6290, 2023. Available from: https://doi.org/10.3390/s23146290

[8] W. Huang, B. Chen, and M. Newaz, “Multi-sensor fusion for target tracking in autonomous vehicles,” *Sensors*, vol. 19, no. 20, p. 4467, 2019. Available from: https://doi.org/10.3390/s19204467

[9] Jocher, “YOLOv8 documentation,” Ultralytics, May 18, 2020. Available from: https://ieeexplore.ieee.org/abstract/document/11207062

[10] Jocher *et al.*, “YOLOv5,” 2021. Available from: https://github.com/ultralytics/yolov5

[11] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *J. Basic Eng.*, vol. 82, no. 1, pp. 35–45, 1960. Available from: https://doi.org/10.1115/1.3662552

[12] Keylabs, “Real-time object detection with YOLOv8,” Feb. 26, 2024. Available from: https://keylabs.ai/blog/real-time-object-detection-with-yolov8

[13] W. Liu *et al.*, “SSD: Single shot multibox detector,” in *Proc. Eur. Conf. Computer Vision (ECCV)*, Amsterdam, Netherlands, 2016, pp. 21–37. Available from: https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2

[14] Y. Liu *et al.*, “Improved non-maximum suppression for embedded single-shot detectors,” *IEEE Trans. Intell. Veh.*, vol. 4, no. 3, pp. 345–354, 2019.

[15] Moksyakov, L. Chen, and J. Zhao, “Multi-object tracking in smart cities using YOLOv8 and Kalman filter algorithms,” *Sensors*, vol. 23, no. 4, p. 9812, 2024.

[16] MathWorks, “Introduction to Kalman filters for object tracking,” 2013. Available from: https://www.mathworks.com/help/vision/examples/target-tracking-using-kalman-filter.html

[17] Milan, L. Leal-Taixe, I. Reid, S. Roth, and K. Schindler, “MOT17 dataset,” *arXiv preprint*, arXiv:1706.05567, 2017. Available from: https://tinyurl.com/ye5j6ahs

[18] Najda *et al.*, “Multi-modal sensor fusion and object tracking for autonomous driving: A systematic survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 25, no. 2, pp. 1425–1443, 2024. Available from: https://ieeexplore.ieee.org/abstract/document/10113239

[19] Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proc. IEEE Conf. Computer Vision Pattern Recognition (CVPR)*, Las Vegas, NV, USA, 2016, pp. 779–788. Available from: https://tinyurl.com/4z5ddna4

[20] Roboflow, “What is YOLOv8? A complete guide,” Nov. 10, 2025.

[21] Redmon and A. Farhadi, “YOLOv3: An incremental improvement,” *arXiv preprint*, arXiv:1804.02767, 2018.

[22] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 28, 2015, pp. 91–99. Available from: https://ieeexplore.ieee.org/abstract/document/7485869

[23] S. Ren, K. He, R. Girshick, and J. Sun, “Improvements to the R-CNN framework for object detection,” *IEEE Trans. Pattern*

Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1142, 2017. Available from: https://tinyurl.com/3rrzezmz

[24] W. Sheng *et al.*, "Multi-objective pedestrian tracking method based on YOLOv8 and improved DeepSORT," *Math. Biosci. Eng.*, vol. 21, no. 2, pp. 1791–1805, 2024. Available from: https://doi.org/10.3934/mbe.2024077

[25] Scaler, "Kalman filter in computer vision," 2023. Available from: https://www.scaler.com/topics/kalman-filter-in-computer-vision/

[26] Ultralytics, "Kalman filter (KF) explained," 2025.

[27] R. Varghese, "YOLOv8: A novel object detection algorithm with advanced accuracy and speed," *IEEE Access*, vol. 12, pp. 16245–16262, 2024. Available from: https://ieeexplore.ieee.org/abstract/document/10533619

[28] R. Verma *et al.*, "Real-time vehicle tracking using LiDAR and mono-camera sensor fusion," *IEEE Sensors J.*, vol. 22, no. 19, pp. 18438–18447, 2022.

[29] Wang, Y. Lu, and Y. You, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," *arXiv preprint*, arXiv:2207.02696, 2022. Available from: https://tinyurl.com/45ar747s

[30] Yadav, R. Kumar, and P. Singh, "Object detection and tracking using YOLOv8 and Kalman filters," *Int. J. Comput. Appl.*, vol. 75, no. 12, pp. 55–61, 2023.

[31] Zheng *et al.*, "Research on vehicle tracking method based on YOLOv8 and adaptive Kalman filtering," *Open J. Appl. Sci.*, vol. 13, no. 9, pp. 1204–1222, 2023.

[32] Y. Zeng, B. Zhou, and S. Yu, "Anchor-free object detection with feature pyramid networks," in *Proc. IEEE Conf. Computer Vision Pattern Recognition (CVPR)*, Salt Lake City, UT, USA, 2018, pp. 7867–7876.