TELKOMNIKA Telecommunication, Computing, Electronics and Control
Vol. 23, No. 6, December 2025, pp. 1729∼1742
ISSN: 1693-6930, DOI: 10.12928/TELKOMNIKA.v23i6.27500

Object detection and tracking with decoupled DeepSORT based on $\alpha\beta$ filter

Lakhdar Djelloul Mazouz, Abdessamad Kaddour Trea, Tarek Amiour, Abdelaziz Ouamri
Image and Signal Laboratory (LSI), University of Sciences and Technology of Oran (USTO-MB), Oran, Algeria

Article Info

Article history:

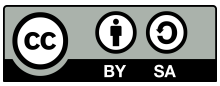
Received Aug 25, 2025
Revised Oct 5, 2025
Accepted Oct 19, 2025

Keywords:

Deep learning
DeepSORT
High order tracking accuracy
Object detection
Object tracking
Video surveillance

ABSTRACT

With the rapid growth of the population, the demand for autonomous video surveillance systems has substantially increased. Recently, artificial intelligence has played a key role in the development of these systems. In this paper, we present an enhanced autonomous system for object detection and tracking in video streams, tailored for transportation and video surveillance applications. The system comprises two main stages: detection stage; this stage employs you only look once (YOLO)v8m, trained on the KITTI dataset, and is configured to detect only pedestrians and cars. The model achieves an average precision of 97.3% and 87.1% for cars and pedestrians classes respectively, resulting a final mean average precision (mAP) of 92.2%. Tracking stage; the tracking component utilizes the DeepSORT algorithm, which originally incorporates a Kalman filter for motion prediction and performs data association using cosine and Mahalanobis distances to maintain consistent object identifiers across frames. To improve tracking performance, we introduce two key modifications to the original DeepSORT: architecture modification and Kalman filter replacement. The tracking tests are carried out on KITTI and MOTChallenge Benchmarks. The final order tracking accuracy (HOTA) scores achieve 77.645 and 54.019 for Cars and Pedestrians classes respectively in the KITTI-Benchmark and 45.436 for the Pedestrians class in the MOTChallenge-Benchmark.



Corresponding Author:

Lakhdar Djelloul Mazouz
Signals and Image Laboratory (LSI), University of Sciences and Technology of Oran (USTO-MB)
Bir El Djir 31000, Oran, Algeria
Email: lakhdar.djelloul@univ-usto.dz

1. INTRODUCTION

Real-time object detection is a critical component in applications such as autonomous vehicles, robotics, and video surveillance. Among the leading algorithms, you only look once (YOLO) stands out for its optimal balance between speed and accuracy, enabling efficient object recognition in static images. Since its introduction, several variants of YOLO have been developed, each improving upon its predecessor to enhance performance and address previous limitations. To public safety and minimize risk, we have developed an intelligent, autonomous video surveillance system capable of real-time detection and tracking of pedestrians and vehicles. The system operates in two stages: object detection using the YOLO algorithm, followed by tracking with an optimized version of DeepSORT.

During recent years, extensive research has been conducted in the field of object detection, leading to the development of various techniques. These techniques can be broadly categorized into two groups: tradi-

tional techniques, based on color [1]-[3], texture [2], morphology [4], edge detection [3], and classical machine learning [5], [6], and advanced techniques using deep learning and artificial intelligence [7]–[12].

In this work, we focus on the detection and tracking of two classes of objects: pedestrians and cars. We train our YOLO model on the KITTI dataset, which achieves a high mAP score. We first apply object detection, using YOLOv8, to identify pedestrians and vehicles. Then, for tracking, we employ DeepSORT, an enhanced version of the simple online real-time tracking (SORT) algorithm. Rather than using a single DeepSORT instance for all object classes, we implement a decoupled approach assigning a dedicated DeepSORT tracker to each class. This reduces inter-class confusion and minimizes ID switches. To optimize computational efficiency, we replace the traditional Kalman filter with a simpler and fast $\alpha\beta$ filter.

This paper is organized as follows. Section 2 presents a comprehensive theoretical basis. Firstly we present the dataset used followed by an introduction of the YOLOv8 detection algorithm and describe how it was trained to enhance its performance. Next, we present the DeepSORT tracker along with two modifications we applied to improve its tracking capabilities. Section 3 is devoted to the detailed description of the steps of the proposed method. Section 4 presents the different metrics employed to assess the performance of the detection algorithm and the various DeepSORT based trackers. Section 5 is primarily dedicated to reporting the experimental results used to evaluate the performance of the detection algorithm and three versions of the DeepSORT tracker (the original and the two modified versions). Finally, section 6 concludes the paper.

2. THE COMPREHENSIVE THEORETICAL BASIS

2.1. Dataset

YOLOv8 was initially trained on the common objects in context (COCO) dataset, which contains approximately 330000 annotated images, spanning 80 object categories. However, our application focuses on developing an autonomous video surveillance system dedicated to monitoring pedestrians and vehicles, that is to say two object classes: person and car. This narrower scope necessitates the use of a more targeted dataset. Consequently, we opted for the KITTI dataset, which includes only 8 object categories, among them the two classes relevant to our study: pedestrian and car [13], [14]. For object detection training, we utilized the KITTI object detection dataset, comprising 7481 training images and 7518 test images, with a total of 80256 labeled instances. All images are in color and stored in PNG format. For object tracking evaluation, we employed the KITTI tracking dataset [15] . It consists of 21 training sequences and 29 test sequences, totaling 8008 color images in PNG format. Among these, 31 sequences comprise images with a resolution of 1242×375 pixels, while the remaining sequences contain images with similar dimensions (i.e., $12xx \times 37x$). To ensure efficient inference and alignment with benchmark standards, our system is configured to detect and track only two object classes: car and person. Notably, the KITTI benchmark restricts its evaluation to these specific categories using the TrackEval-Master python source codes [16]. To generalize our application, another benchmark evaluation is used (MOTChallenge [17]).

2.2. Object detection

As previously mentioned, this study employs the YOLO algorithm for object detection. YOLO is a single-stage detector that partitions the input image into a grid of equally sized cells, typically of dimension $(N \times N)$. Each cell is responsible for predicting the presence and location of objects within its boundaries. In this work, we utilize the YOLOv8m model, released by Ultralytics in January 2023. The YOLOv8 family comprises five variants designed for tasks such as object detection, segmentation, and classification: YOLOv8n (Nano), YOLOv8s (Small), YOLOv8m (Medium), YOLOv8l (Large), and YOLOv8x (Extra large) [18], [19]. YOLOv8n is the most lightweight and fastest, making it suitable for real-time applications with limited computational resources. In contrast, YOLOv8x offers the highest accuracy, albeit with increased computational cost and inference time.

2.3. Object tracking

Online object tracking systems commonly rely on the SORT algorithm, which consists of four principal components: detection, estimation, data association, identity management (creation and termination). Despite its efficiency, SORT faces significant limitations in maintaining consistent object identifiers (IDs), particularly when objects reappear following occlusion. In such cases, the algorithm often assigns a new ID, treating the reappearing object as entirely new. To address this limitation, the DeepSORT algorithm introduces

a more robust tracking mechanism by incorporating a deep appearance-based association metric. This enhancement enables the tracker to consider visual features of the object, thereby improving identity preservation across occlusions. The architecture of the DeepSORT algorithm is illustrated in Figure 1.

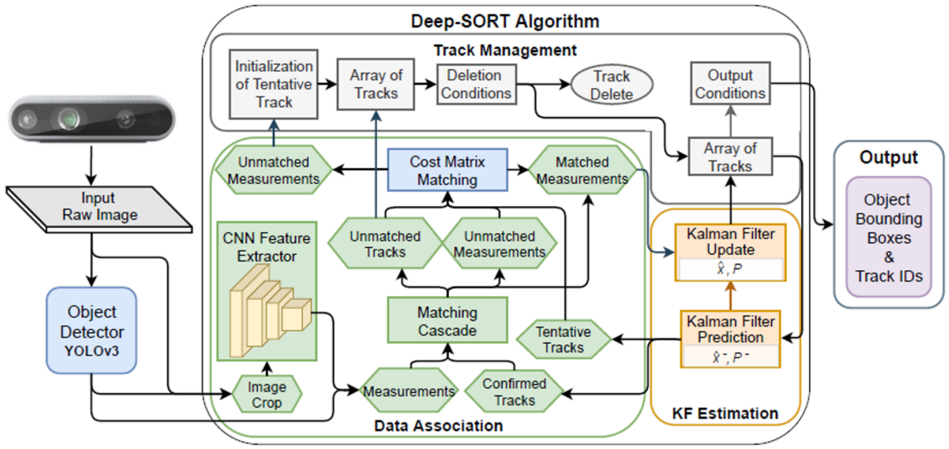


Figure 1. DeepSORT architecture [20]

The DeepSORT algorithm maintains consistent object identifiers by evaluating a combined distance metric. An object is assigned the same ID if the following condition is satisfied [21]-[23]:

$$\lambda \times d_1(i,j) + (1-\lambda) \times d_2(i,j) \leq Threshold \tag{1}$$

In this formulation, $d_1(i,j)$ represents the cosine distance, while $d_2(i,j)$ denotes the Mahalanobis distance. The variables i and j correspond to the coordinates of the object's center, and $\lambda \in [0,1]$ is a weighting factor that balances the contribution of appearance-based and motion-based metrics. This criterion enables the algorithm to associate reappearing objects with their original identities, thereby improving tracking robustness in scenarios involving occlusion.

3. METHOD

In order to enhance the performance of DeepSORT, two critical modifications were incorporated into its original framework.

3.1. Architecture modification

To reduce identity confusion between distinct object classes, we propose to modify the architecture of the DeepSORT tracking algorithm. Specifically, the modification involves deploying separate DeepSORT instances for each class: one dedicated to pedestrians and the other to vehicles (e.g., cars). These trackers operate concurrently and independently, allowing for class-specific identity management and reducing cross-class mis-association. The final tracking output is obtained by merging the results from both trackers, thereby preserving class integrity throughout the tracking process. The proposed dual-tracker architecture is illustrated in Figure 2. It will be referred in the following as the decoupled DeepSORT algorithm.

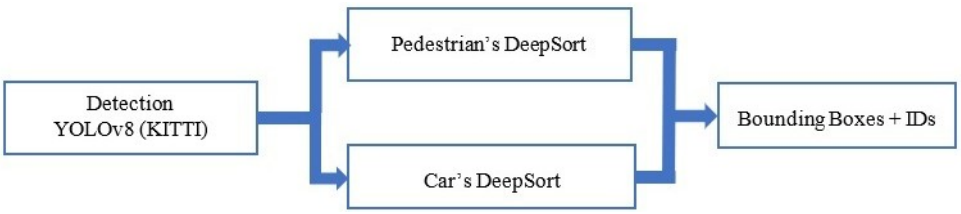


Figure 2. Decoupled DeepSORT architecture

3.2. Kalman filter replacement

The second modification that we made to the DeepSORT algorithm involves replacing the Kalman filter with a simplified and computationally efficient alternative: the $\alpha\beta$ filter. This filter is a fixed coefficients, which may be viewed as a second-order steady-state Kalman. It was originally designed for target tracking in the radar field. Compared to the Kalman filter, the $\alpha\beta$ filter offers several advantages in the context of real-time object tracking [24]:

- Simplified prediction and update mechanisms: unlike the Kalman filter, which dynamically computes gain values based on the innovation covariance matrix in each frame, the $\alpha\beta$ filter utilizes fixed gain parameters, α for position and β for velocity, resulting in more straightforward computations, as shown in (2) and (3).
- Reduced computation complexity: the $\alpha\beta$ filter updates the state vector and associated parameters without requiring matrix inversion, thereby significantly lowering the computational burden compared to the Kalman filter, as shown in (4) and (5).
- Comparable performance in simple tracking scenarios: despite its simplicity, the $\alpha\beta$ filter achieves tracking accuracy similar to that of the Kalman filter in scenarios with limited noise and linear motion.

The state vector at time step (frame) k, used in the filter, encompasses the object's bounding box parameters $[X_k, Y_k, W_k, H_k]$ and their velocities [$\dot{X}_k$,$\dot{Y}_k$,$\dot{W}_k$,$\dot{H}_k$], with $[X_k, Y_k]$ denoting the object center coordinates, and $[W_k, H_k]$ the height and width of the bounding box.

The operational steps of the $\alpha\beta$ filter are outlined below. To simplify, we give the equations only for the x component. Similar equations apply for the remaining component. Let Xe_k, Xp_k and Xm_k denote, respectively, the estimate, the predicted and the measurement (provided by the detection stage) of X_k.

- Initialization: The X_k component of the state vector is initialized with the X coordinate of the center of first ($k = 0$) bounding box, provided by the detection stage. The $\dot{X}_k$ of the state vector is initialized with zero.
- Prediction

$$Xp_k = Xe_k + T \times \dot{X}e_k \tag{2}$$

$$\dot{X}p_k = \dot{X}e_k \tag{3}$$

where T represents the frame period.

- Update

$$Xe_k = Xp_k + \alpha_x \times (Xp_k - Xm_k) \tag{4}$$

$$\dot{X}e_k = \dot{X}p_k + \frac{\beta_x}{T} \times (Xp_k - Xm_k) \tag{5}$$

where α_x and β_x are the fixed coefficients of the filter, relative to the component X_k of the state vector. The selection of these coefficients depends on the system's dynamics: The higher these coefficients are the more responsive is the filter.

4. METRICS

4.1. Object detection metrics

To determine which object detector is fit for our application, we can employ different metrics such as recall, precision and intersection over union *IoU*, *F1 score*, average precision (*AP*) and mean average precision (*mAP*) [25]. To evaluate the suitability of various object detection models for a given application, several performance metrics are commonly employed. These include recall, precision, intersection over Union (IoU), F1 score, average precision (AP), and Mean average precision (mAP) [25].

4.1.1. Average precision

We begin by defining two very important metrics for detector evaluation, which are: precision and recall. These two metrics are given by:

$$Precision = \frac{TP}{TP + FP} \tag{6}$$

$$Recall = \frac{TP}{TP + FN} \tag{7}$$

where :

- TP = True positives
- FP = False positives
- FN = False negatives

For a single class, the AP is computed as the area under the precision-recall curve:

$$\int_0^1 P(r)\,\mathrm{d}r \tag{8}$$

where $P(r)$ is the precision as a function of recall.

In practice, the average precision for class may be approximated using:

$$AP_i = \sum_{j=0}^{k-1}[Recall(i,j) - Recall(i,j-1)] \times Precision(i,j) \tag{9}$$

where $Recall(i,j)$ and $Precision(i,j)$ are the recall and precision of class i, evaluated using the jth threshold and k is the number of thresholds. The AP is a metric that may be used to assess the performance of detection and localization algorithms, the higher it is the more efficient is an algorithm. It corresponds to the area under the precision-recall curve, and it may be estimated using the pairs (precision, recall) for several confidence thresholds

4.1.2. Mean average precision

The mAP is another important performance metric, mainly used for evaluating machine learning models. It is defined as the average of the average precisions of different detected classes, and is calculated through the (10):

$$mAP = \frac{1}{N} \times \sum_{i=1}^{N} AP_i \tag{10}$$

where N indicates the number of classes and AP_i is the average precision of class i.

4.1.3. Score F1 ($F1_{score}$)

The $F1_{Score}$ is a performance metric used in classification and detection tasks. It represents the harmonic mean of precision and recall, thus balancing both metrics into a single value.

$$F1_{score} = \frac{Precision \times Recall}{\frac{Precision+Recall}{2}} \tag{11}$$

This metric is particularly useful when the dataset is imbalanced, as it considers both false positives and false negatives.

4.2. Object tracking metrics [26]

The classification of events, activities, and relationships (CLEAR) workshop has defined a common, unified framework for evaluating multi-object tracking (MOT) algorithms, known as CLEAR−MOT metrics, which places the multiple object tracking accuracy (MOTA) metric as the primary metric for tracking evaluation, although it has been criticized for favoring detection over association. In recent years, the most commonly used benchmarks for evaluating multi-object tracking algorithms are MOTChallenge and KITTI, the main metrics used in these benchmarks are MOTA, IDF1 and high order tracking accuracy (HOTA).

4.2.1. DetA

Detection accuracy ($DetA$) measures how well the tracker detects objects, independent of identity preservation. The formula for its computation is:

$$DetA = \frac{TP}{TP + FP + FN} \tag{12}$$

4.2.2. AssA

Association accuracy ($AssA$) evaluates how well the tracker maintains object identities across frames. It is computed by:

$$AssA = \frac{TPA}{TPA + FPA + FNA} \tag{13}$$

where :
- TPA = True positives association
- FPA = False positives association
- FNA = False negatives association

4.2.3. Identification F1 score (IDF1)

$IDF1$ evaluates the accuracy of identity preservation in tracking. It is the harmonic mean of identity precision and identity recall:

$$IDF1 = \frac{2 \times IDTP}{2 \times IDTP + IDFP + IDFN} \tag{14}$$

where :
- $IDTP$ = Identity true positives
- $IDFP$ = Identity false positives
- $IDFN$ = Identity false negatives

4.2.4. LocA

Localization accuracy ($LocA$) measures the average spatial alignment of correctly detected objects using IoU. The formula for its computation is:

$$LocA = \frac{1}{|TP|} \times \sum_{c \in TP} Loc - IoU(c) \tag{15}$$

Where $IoU(c)$ represents the intersection over union for true positive candidate c.

4.2.5. Multiple object tracking accuracy

MOTA evaluates overall tracking performance by penalizing false positives, missed detections, and identity switches. It reflects how well the tracker maintains object presence and identity. Its formula is:

$$MOTA = 1 - \frac{\sum FP \times \sum FN \times \sum IDSW}{\sum GTDet} \tag{16}$$

where :
- k = Frame index
- $IDSW$ = Identity switches
- $GTDet$ = Ground-truth tracks detection

4.2.6. Multiple object tracking precision

$MOTP$ measures the average localization precision of matched object detections, based on the spatial overlap (IoU) between predicted and ground truth bounding boxes. It can be computed by:

$$MOTP = \frac{\sum_{k,i} IoU_{k,i}}{\sum_k c_k} \tag{17}$$

Where $IoU_{k,i}$ represents the bounding box overlap of object i, at time k and c_k the number of matches in frame k.

4.2.7. HOTA

HOTA is a more recent metric that jointly evaluates detection, association, and LocA. It represents the geometric mean of detection and association accuracies:

$$HOTA_\alpha = \sqrt{Dett_\alpha \times Ass_\alpha} = \sqrt{\frac{\sum_{c \in TP} Ass - IoU(c)}{|TPA_\alpha| + |FNA_\alpha| + |FPA_\alpha|}} \tag{18}$$

In this formula, the term α represents the different IoU thresholds used to used to compute the metric. A generalized version of $HOTA_\alpha$, denoted as HOTA is computed over a range of thresholds $\alpha \in$ [0,1]:

$$HOTA = \int_{0 \leq \alpha \leq 1} HOTA_\alpha = \sum_{\alpha=0.05}^{0.95} HOTA_\alpha \tag{19}$$

HOTA is a scalar metric that summarizes the overall tracking performance of a system by averaging the HOTA scores across a range of IoU thresholds. It captures the balance between detection accuracy, association accuracy, and localization precision, making it one of the most comprehensive metrics for evaluating multi-object tracking. Unlike traditional metrics that focus heavily on either detection MOTA or identity preservation (IDF1), HOTA integrates all three (aspects: detection, association, and localization) into a unified score that reflects performance across varying spatial tolerances.

5. EXPERIMENTAL RESULTS

5.1. Object detection

In training the KITTI dataset was split into different folders with a ratio of 80:10:10 for training, validation and test respectively. The KITTI dataset was used for the evaluation of the employed object detection method. This dataset was split into three folders, with a ratio of 80:10:10 for training, validation and test, respectively. The hardware configuration used for training is:

- Graphics processing unit (GPU): NVIDIA GeForce RTX 3060.
- Central processing unit (CPU): 10th Gen Intel Core(TM) i5-10400, 2.9 Ghz (12 CPU).
- Memory 64 GB.

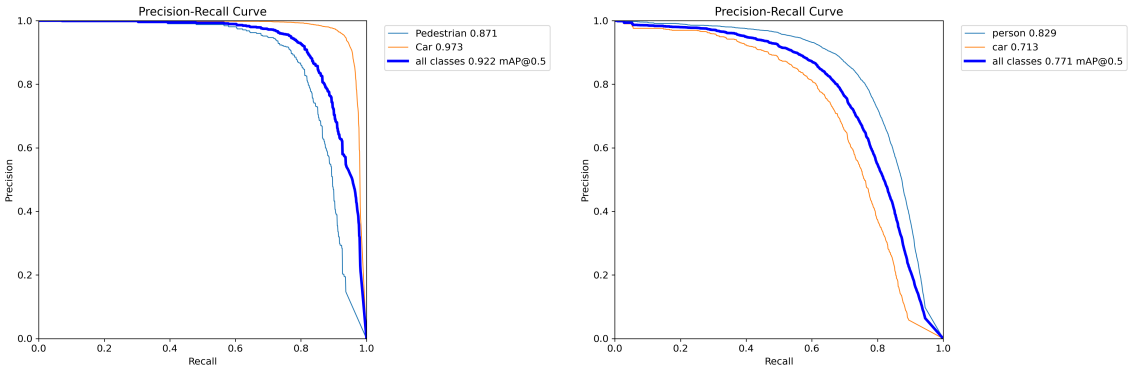
The software configuration used in training:

- Python version 3.11.9, and Visual studio code version 1.102.3.

Training hyperparameters:

- Epochs = 50, imgsz = 640, batch = 16, learning rate = 0.01 and 60fps.

The mean average precision (mAP) obtained from training is presented by Figure 3:



(a) (b)

Figure 3. Average precision: (a) KITTI dataset and (b) COCO dataset

Figure 3 shows the comparative mAP results, revealing a clear advantage in our $KITTI$ trained model Figure (3)a compared to the $COCO$ reference Figure 3(b), with a significant improvement in the score: 4.2% for the pedestrian class and 26% for the car class.

5.2. Object tracking results

Once a high-performance object detector has been obtained, it must be integrated with a tracking algorithm to develop a fully autonomous video surveillance system. For the tracking component, we selected DeepSORT, one of the most widely adopted algorithms in this domain, due to its proven effectiveness in tracking moving objects within video scenes.

As previously noted, two modifications were introduced to enhance DeepSORT's performance. To evaluate the impact of these improvements, we compare tracking metrics obtained using the modified versions against those from the original implementation. The tracking hyperparameters used in both configurations are detailed below:

- $max\_cosine\_distance$ = 0.3 : enforces stricter appearance matching.
- $nn\_budget = 100$: specifies the number of appearance features to cache.
- $max\_IoU\_distance = 0.7$: sets the bounding box overlap threshold.
- $max\_age = 60$: determines the number of frames a track is retained without updates.
- $n\_init = 3$: indicates the number of detections required to confirm a track.

The α value was chosen empirically from the interval [0,1]. The optimal β value was chosen according to the Benedict-Bordner rule [24] , where:

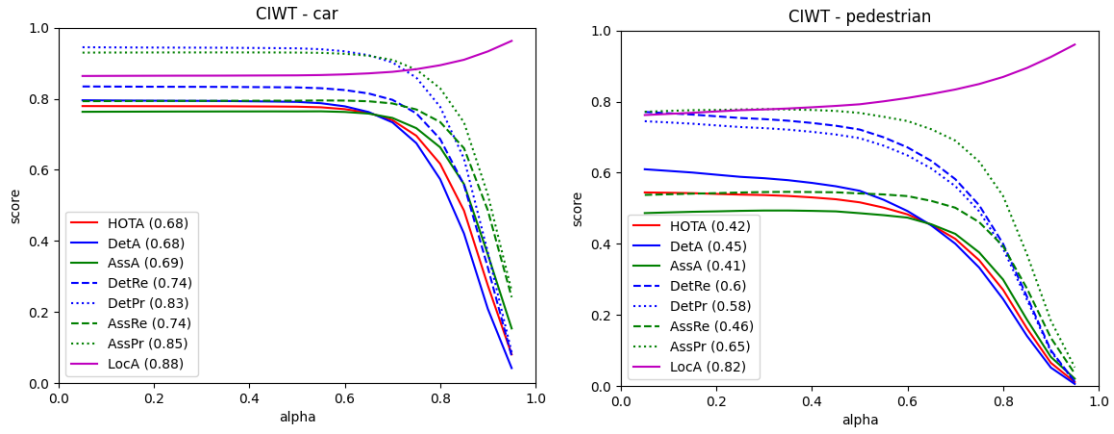
$$\beta = \frac{\alpha^2}{2 - \alpha} \tag{20}$$

After several tests, the couple (α, β) that gave the best results in terms of metrics was chosen to validate the experimental results of our application is (0.2, 0.022). The real-time performance specifications: FPS: 30, GPU: 15% and latency: 37 ms. Table 1 and Figure 4 present the performance metrics obtained from the original DeepSORT implementation.

According to Table 1 and Figure 4 , we can see that the scores obtained for the MOTA and HOTA (Figure 4(a)) metrics were 68.359 and 0.68 respectively, with an IDswitch of 434 for the car class, while for the pedestrian class, the MOTA and HOTA (Figure 4 (b)) scores were 42.474 and 0.42 respectively, with an IDswitch of 510. Table 2 and Figure 5 show the different metrics obtained from the implementation of the decoupled DeepSORT.

Table 1. Original DeepSORT metrics

Class	MOTA	MOTP	TP	FN	FP	IDSW	Dets	GT_Dets	IDs	GT_IDs
Car	68.359	68.007	20026	4044	1234	434	21260	24070	938	564
Pedestrian	42.474	44.682	8103	3006	3393	510	11496	11109	472	167

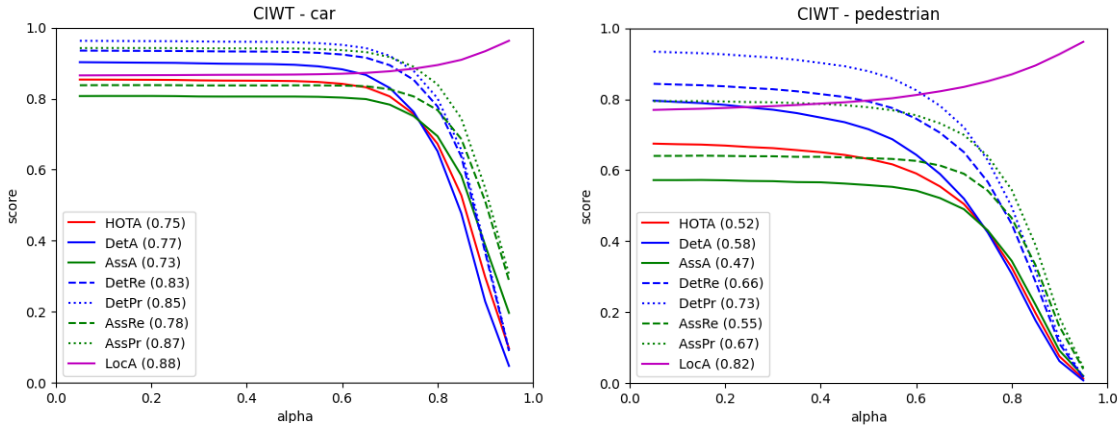


(a) (b)

Figure 4. Original DeepSORT metrics: (a) car and (b) pedestrian

Table 2. Decoupled DeepSORT metrics

Class	MOTA	MOTP	TP	FN	FP	IDSW	Dets	GT_Dets	IDs	GT_IDs
Car	88.238	86.772	22432	1638	947	381 (-53)	23379	24070	823	564
Pedestrian	67.954	79.448	8959	2150	1072	370 (-140)	10031	11109	308	167



(a) (b)

Figure 5. Decoupled DeepSORT metrics: (a) car and (b) pedestrian

Compared with the original DeepSORT metrics, we can see that the scores of MOTA and HOTA (Figure 5(a)) have improved significantly by +19.238 (88.238) and +0.07 (0.75) respectively, and a reduction by -53 (381) in IDswitch for the car class. while for the pedestrian class, the MOTA and HOTA (Figure 5 (b)) scores show an improvement of +25.48 and +0.1 respectively and a reduction of -150 in the IDswitch due to the use of the decoupled DeepSORT which solves this problem by using two parallel architectures, making it impossible to confuse the yolo classes. Table 3 and Figure 6 show the different metrics obtained from the implementation of the $\alpha\beta$ filter based decoupled DeepSORT algorithm.

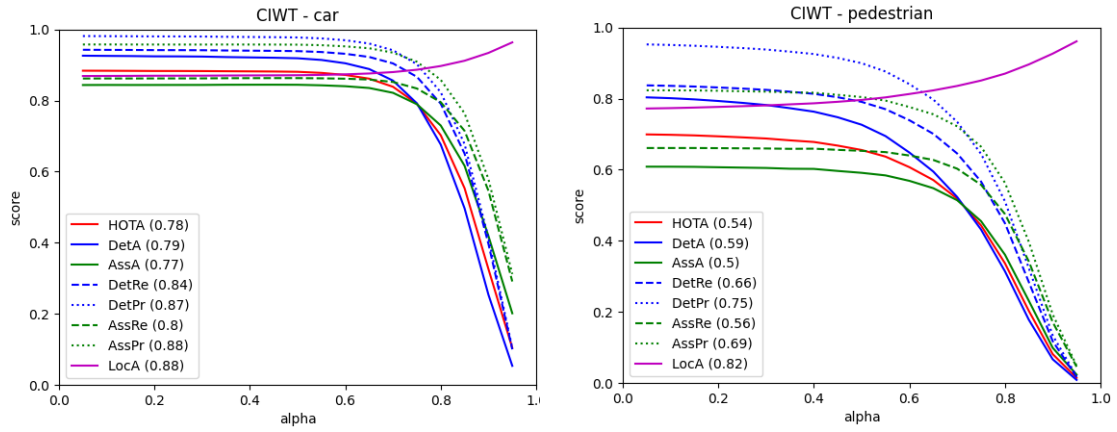
Compared with the original DeepSORT metrics, we can see that the scores of MOTA and HOTA Figure 6(a) have improved significantly by +22.023 and +0.1 respectively, and a reduction by -188 in IDswitch for the car class. while for the pedestrian class, the MOTA and HOTA Figure 6(b) scores show an improvement of +27.073 and +0.12 respectively and a reduction of -172 in the IDswitch. Tables 4 and 5 present a comparison of the metrics obtained with the 3 versions of the DeepSORT algorithm (the original one and the 2 modified versions), for the pedestrians and cars classes.

As shown in the comparison, all evaluation metrics are improved by the decoupled DeepSORT algorithm compared to the original version. These metrics are further enhanced when the $\alpha\beta$ filter is integrated into the decoupled DeepSORT architecture. In order to generalize the results of our application, we repeated the tests on another evaluation benchmark (MOTChallenge benchmark [17]). The MOTChallenge focuses on pedestrian tracking only.

The results obtained are presented in the following Figure 7, where Figure 7(a) represents the metrics of the original DeepSORT, Figure 7(b) represents the metrics obtained from the decoupled DeepSORT, and Figure 7(c) represents the metrics obtained from the decoupled DeepSORT based on the $\alpha\beta$ filter. According to the Figure 7 and Tables 6 and 7, we can conclude that the results are consistent with those obtained from the KITTI benchmark [15]. Figure 8 show some tracking results using the original DeepSORT before and after occlusion from the KITTI object tracking evaluation dataset [16].

Table 3. Decoupled DeepSORT based $\alpha\beta$ filter metrics

Class	MOTA	MOTP	TP	FN	FP	IDSW	Dets	GT_Dets	IDs	GT_IDs
Car	90.382	87.085	22630	1440	494	246 (-188)	23124	24070	745	564
Pedestrian	69.547	79.434	8931	2178	835	338 (-172)	9766	11109	256	167



(a) (b)

Figure 6. Metrics of the $\alpha\beta$ filter based decoupled DeepSORT : (a) cars and (b) pedestrians

Table 4. Comparison of the metrics obtained with the 3 versions of the DeepSORT algorithm, for the cars' class

Metric (%)	Original DeepSORT	Decoupled DeepSORT	Decoupled DeepSORT based $\alpha\beta$ filter
HOTA	68.359	74.767 (+6.408)	77.645 (+9.286)
DetA	68.007	77.032 (+9.025)	79.296 (+11.289)
AssA	69.093	73.074 (+3.981)	76.502 (+7.409)
DetRe	73.712	82.754 (+9.042)	83.742 (+10.03)
DetPr	83.455	85.2 (+1.745)	87.168 (+3.713)
AssRe	73.694	77.82 (+4.126)	80.209 (+6.515)
AssPr	85.374	86.728 (+1.354)	88.342 (+2.998)
LocA	87.966	88.099 (+0.133)	88.385 (+0.419)

Table 5. Comparison of the metrics obtained with the 3 versions of the DeepSORT algorithm, for the pedestrians' class

Metric (%)	Original DeepSORT	Decoupled DeepSORT	Decoupled DeepSORT based $\alpha\beta$ filter
HOTA	42.474	52.506 (+10.032)	54.019 (+11.545)
DetA	44.682	58.606 (+13.924)	59.142 (+14.46)
AssA	40.576	47.297 (+6.721)	49.589 (+9.013)
DetRe	59.805	66.285 (+6.78)	65.835 (+6.03)
DetPr	57.792	73.408 (+15.616)	74.888 (+17.096)
AssRe	46.445	54.441 (+7.996)	56.331 (+9.886)
AssPr	65.463	68.092 (+2.629)	68.928 (+3.465)
LocA	81.742	82.042 (+0.3)	82.067 (+0.325)

Table 6. MOTChallenge benchmark's metrics with 3 version of DeepSORT

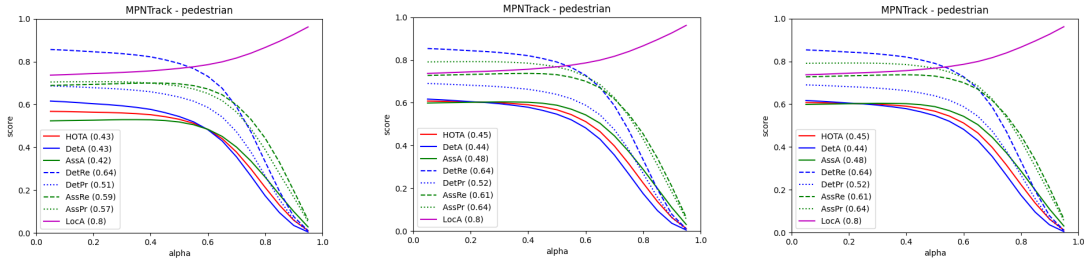
Algorithm	MOTA	MOTP	TP	FN	FP	IDSW	Dets	GT_Dets	IDs	GT_IDs
Original Deep-SORT	33.171	76.761	31767	8138	18009	512	49776	39905	794	500
Decoupled Deep-SORT	34.399	76.746	31682	8223	17681	274 (-238)	49363	39905	873	500
Decoupled Deep-SORT based $\alpha\beta$	34.399	76.746	31682	8223	17681	274 (-238)	49363	39905	873	500

In the original DeepSORT algorithm, both classes are tracked simultaneously by the same DeepSORT. Occlusion can cause confusion between the two classes, i.e., a pedestrian can be classified as a car and vice versa. This confusion is translated by an IDSwitch. As shown in the Figure 8, the system confuses their

identities: the pedestrian (yellow bounding box - Figure 8(a)) is incorrectly tracked as the car (blue bounding box - Figure 8(b)). Decoupled DeepSORT solves this problem by using two separate tracking systems, making it impossible to confuse the two classes. As shown in the Figure 9, both the pedestrian and car have maintained their correct identities after the occlusion (Figures 9(a) and 9(a)). Figure 9 shows some tracking results using the decoupled DeepSORT based on Kalman filter before and after occlusion from the KITTI object tracking evaluation dataset [16].

Table 7. Comparison of the MOTChallenge benchmark's metrics obtained with the 3 versions of the DeepSORT algorithm for the pedestrians's class

Metric	Original DeepSORT	Decoupled DeepSORT	Decoupled DeepSORT based $\alpha\beta$ filter
HOTA (%)	42.607	45.436 (+2.829)	45.436 (+2.829)
DetA (%)	43.486	43.616 (+0.13)	43.616 (+0.13)
AssA (%)	42.122	47.705 (+5.583)	47.705 (+5.583)
DetRe (%)	64.151	63.99 (-0.161)	63.99 (-0.161)
DetPr (%)	51.43	51.729 (+0.299)	51.729 (+0.299)
AssRe (%)	58.514	61.335 (+2.821)	61.335 (+2.821)
AssPr (%)	57.345	64.052 (+6.707)	64.052 (+6.707)
LocA (%)	79.799	79.833 (+0.043)	79.833 (+0.043)



(a) (b) (c)

Figure 7. MOTChallenge benchemark's metrics for the pestrian class: (a) original DeepSORT, (b) decoupled DeepSORT, and (c) decoupled DeepSORT based on $\alpha\beta$ filter

(a) (b)

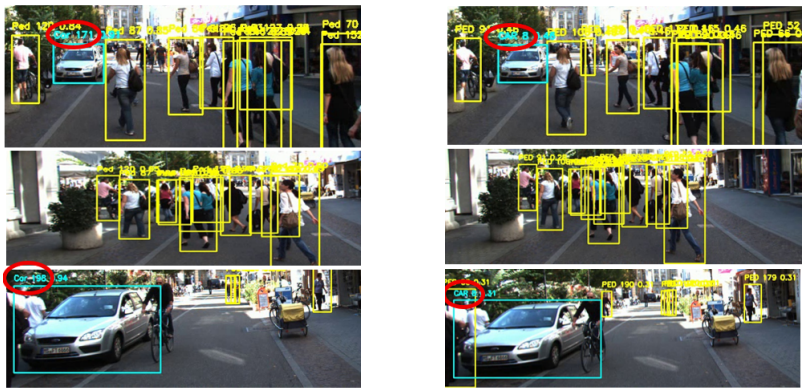
Figure 8. Original DeepSORT track results: (a) before occlusion and (b) after occlusion

(a) (b)

Figure 9. Decoupled DeepSORT based on Kalman filter track results: (a) before and (b) after occlusion

Although the decoupled DeepSORT method based on Kalman filter has fixed ID switching between the two classes, it still struggles to reassociate the same ID to an object after it has been occluded in challenging

scenarios. As shown in Figure 10, a car is initially tracked with ID 171. After being completely obscured by pedestrians, it reappears and is assigned a new ID 198 (Figure 10(a)), as it is mistakenly considered a new object. Replacement of Kalman filter with an $\alpha\beta$ filter significantly reduced this type of failure. In the same scenario, as shown in Figure 10(b), the same car is now consistently tracked with ID 8 before and after the occlusion, successfully maintaining its correct identity demonstrating improved tracking consistency.



(a) (b)

Figure 10. Track results: (a) decoupled DeepSORT based on Kalman filter and (b) decoupled DeepSORT based on $\alpha\beta$ filter

6. CONCLUSION

In this paper, we presented an enhancement to the detection-tracking framework and apply it to pedestrians and cars monitoring. The detection was performed using the powerful YOLOv8m algorithm, trained on the KITTI dataset. The model achieved a mAP of 97.3% for cars and 87.1% for pedestrians, while tracking was based on the well-known DeepSORT algorithm. The initial HOTA score was 68.359 and 42.474 for cars and pedestrian classes, respectively, in KITTI benchmark and 42.607 for pedestrian class in MOTchallenge-Benchmark. To improve tracking performance, we introduced two key modifications to the original DeepSORT. Firstly, class-specific tracking, we assigned a dedicated tracker to each object class. This modified version, referred to as decoupled DeepSORT, significantly reduced the number of identity switches (IDSW): by 12.21% for the car class and 46,48% for the pedestrian class in KITTI benchmark and 27% in MOTchallenge benchmark. Consequently, the HOTA metric improved by 10.032% for pedestrians and 6.408% for cars. Secondly, $\alpha\beta$ Filter Integration, we replaced the Kalman filter with the $\alpha\beta$ filter in the tracking module. This substitution not only reduced computational overhead but also led to further improvements in tracking metrics. Specifically, the $\alpha\beta$ filter-based decoupled DeepSORT architecture enhanced the HOTA score by 11.545% for pedestrians and 9.286% for cars in KITTI benchmark and 2.829% for Pedestrians in MOTchallenge benchmark, compared to the original DeepSORT. The final HOTA score achieved 77.645 and 54.019 for cars and pedestrians classes respectively in the KITTI benchmark and 45.436 for pedestrians class in MOTchallenge benchmark. Our application performs well in the supervision of public places (e.g. airports, motorways, and metro stations) as it corresponds to the dataset's parameters (angle, field of vision, object shape, and object visibility). However, in the case of drone applications, the parameters change completely, which significantly reduces the performances of the system.

FUNDING INFORMATION

No funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Lakhdar Djelloul Mazouz	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	
Abdessamad Kaddour Trea	✓		✓						✓					
Tarek Amiour	✓		✓						✓					
Abdelaziz Ouamri		✓		✓	✓		✓			✓		✓	✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal Analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject Administration
Fu : **Fu**nding Acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The authors confirm that the data supporting the findings of this study are available within the article.

REFERENCES

[1] K. Y. Chiu and S. F. Lin, "Lane detection using color-based segmentation," *in IEEE Intelligent Vehicles Symposium, Proceedings, IEEE*, 2005, pp. 706–711, doi: 10.1109/IVS.2005.1505186.

[2] S. M. Muddamsetty, D. Sidibé, A. Trémeau, and F. Mériaudeau, "Salient objects detection in dynamic scenes using color and texture features," *Multimedia Tools and Applications*, vol. 77, no. 5, pp. 5461–5474, Mar. 2018, doi: 10.1007/s11042-017-4462-y.

[3] L.-W. Tsai, J.-W. Hsieh, and K.-C. Fan, "Vehicle detection using normalized color and edge map," *IEEE Transactions on Image Processing*, vol. 16, no. 3, pp. 850–864, Mar. 2007, doi: 10.1109/TIP.2007.891147.

[4] R. Kalsotra and S. Arora, "Morphological based moving object detection with background subtraction method," *in 4th IEEE International Conference on Signal Processing, Computing and Control, ISPCC 2017*, IEEE, Sep. 2017, pp. 305–310, doi: 10.1109/ISPCC.2017.8269694.

[5] C. Q. Lai and S. S. Teoh, "A review on pedestrian detection techniques based on histogram of oriented gradient feature," in *2014 IEEE Student Conference on Research and Development, SCOReD 2014,* IEEE, Dec. 2014, pp. 1–6, doi: 10.1109/SCORED.2014.7072948.

[6] L. D. Mazouz, A. Meche, A. Ouamri, and A. W. A. Darna, "Two-stage HOG/SVM for license plate detection and recognition," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 34, no. 1, pp. 210–223, Apr. 2024, doi: 10.11591/ijeecs.v34.i1.pp210-223.

[7] F. N. Kılıçkaya, M. Taşyürek, and C. Öztürk, "Performance evaluation of YOLOv5 and YOLOv8 models in car detection," *Imaging and Radiation Research*, vol. 6, no. 2, p. 5757, Jul. 2024, doi: 10.24294/irr.v6i2.5757.

[8] F. Tang, F. Yang, and X. Tian, "Long-distance person detection based on YOLOv7," *Electronics (Switzerland)*, vol. 12, no. 6, p. 1502, Mar. 2023, doi: 10.3390/electronics12061502.

[9] R. Patel and P. Meduri, "Car detection based algorithm for automatic parking space detection," in *Proceedings - 19th IEEE International Conference on Machine Learning and Applications, ICMLA 2020, IEEE*, Dec. 2020, pp. 1418–1423, doi: 10.1109/ICMLA51294.2020.00220.

[10] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila, "EuroCity persons: A novel benchmark for person detection in traffic scenes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 8, pp. 1844–1861, Aug. 2019, doi: 10.1109/TPAMI.2019.2897684.

[11] B. Benjdira, T. Khursheed, A. Koubaa, A. Ammar, and K. Ouni, "Car detection using unmanned aerial vehicles: Comparison between faster R-CNN and YOLOv3," in *2019 1st International Conference on Unmanned Vehicle Systems-Oman, UVS 2019*, IEEE, Feb. 2019, pp. 1–6, doi: 10.1109/UVS.2019.8658300.

[12] C. E. Kim, M. M. Dar Oghaz, J. Fajtl, V. Argyriou, and P. Remagnino, "A comparison of embedded deep learning methods for person detection," *VISIGRAPP 2019 - Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, vol. 5, pp. 459–465, Jan. 2019, doi: 10.5220/0007386304590465.

[13] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images," *arxiv*, Jun. 2016.

[14] M. Gao, "YOLO and COCO dataset detective porfermance," *World Journal of Information Technology* , vol. 2, no. 2, 2024, doi: 10.61784/wjit3002.

[15] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Kitti Dataset," 2012. [Online]. Available: http://www.cvlibs.net/datasets/kitti/ (accessed May 10, 2025).

[16] J. Luiten and A. Hoffhues, "TrackEval." [Online]. Available: https://github.com/JonathonLuiten/TrackEval (accessed May 10, 2025).

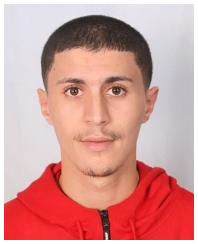
[17] P. Dendorfer et al., "MOTChallenge: a benchmark for single-camera multiple target tracking," *International Journal of Computer Vision*, vol. 129, no. 4, pp. 845–881, Apr. 2021, doi: 10.1007/s11263-020-01393-0.

[18] J. Terven, D.-M. Córdova-Esparza, and J.-A. Romero-González, "A comprehensive review of YOLO achitectures in computer vision: from YOLOv1 to YOLOv8 and YOLO-NAS," *Machine Learning and Knowledge Extraction*, vol. 5, no. 4, pp. 1680–1716, Nov. 2023, doi: 10.3390/make5040083.
[19] M. Hussain, "YOLO-v1 to YOLO-v8, the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, vol. 11, no. 7, p. 677, Jun. 2023, doi: 10.3390/machines11070677.
[20] R. Pereira, G. Carvalho, L. Garrote, and U. J. Nunes, "Sort and Deep-SORT based multi-object tracking for mobile robotics: evaluation with new data association metrics," *Applied Sciences (Switzerland)*, vol. 12, no. 3, p. 1319, Jan. 2022, doi: 10.3390/app12031319.
[21] F. Yang, X. Zhang, and B. Liu, "Video object tracking based on YOLOv7 and DeepSORT," *arXiv*, Jul. 2022, doi: 10.48550/arXiv.2207.12202.
[22] Ying-Zu Lin, Soon-Jyh Chang, Yen-Ting Liu, Chun-Cheng Liu, and Guang-Ying Huang, "A 5b 800MS/s 2mW asynchronous binary-search ADC in 65nm CMOS," in *2009 IEEE International Solid-State Circuits Conference - Digest of Technical Papers*, IEEE, Feb. 2009, p. 80–81,81a, doi: 10.1109/ISSCC.2009.4977317.
[23] N. Wojke and A. Bewley, "Deep cosine metric learning for person re-identification," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE, Mar. 2018, pp. 748–756, doi: 10.1109/WACV.2018.00087.
[24] L. D. Mazouz, A. Meche, and M. Dahmani, "An efficient real time implementation of a fast IMM for tracking a manoeuvring target," in *2019 International Conference on Advanced Electrical Engineering, ICAEE 2019*, IEEE, Nov. 2019, pp. 1–6, doi: 10.1109/ICAEE47123.2019.9014753.
[25] J. Hui, "mAP (mean Average Precision) for Object Detection," 2018. [Online]. Available: https://medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173 (accessed Aug. 10, 2025).
[26] J. Luiten et al., "HOTA: A higher order metric for evaluating multi-object tracking," *International Journal of Computer Vision*, vol. 129, no. 2, pp. 548–578, Feb. 2021, doi: 10.1007/s11263-020-01375-2.

BIOGRAPHIES OF AUTHORS

Lakhdar Djelloul Mazouz was born in Algeria. He received the engineer degree in electronics in 2003, the degree of Magister in Modern Communication Techniques in 2008 and Ph.D. degree in image processing in 2024, from the University of Sciences and Technology of Oran (USTO-MB), Algeria. He is actually an Associate Professor at USTO-MB university. His research interests are concerned with signal and image processing, deep learning, artificial intelligence, tracking, telecommunications, and real-time implementation. He can be contacted at email: lakhdar.djelloul@univ-usto.dz.

Abdessamad Kadour Trea was born in 2002 in Algeria. He received a license in Telecommunications in 2023 and a Master's degree in Networks and Telecommunications in 2025, both from the University of Sciences and Technology of Oran (USTO-MB), Algeria. His research interests include image processing, deep learning, artificial intelligence, network technologies, and telecommunication. He can be contacted at email: samadkdr27@gmail.com



Tarek Amiour was born in 2001 in Algeria. He received a license in Telecommunications in 2023 and a Master's degree in Networks and Telecommunications in 2025, both from the University of Sciences and Technology of Oran (USTO-MB), Algeria. His research interests include image processing, deep learning, artificial intelligence, network technologies, and telecommunication. He can be contacted at email: amiourtarek05@gmail.com

Abdelaziz Ouamri was born in Algeria. He received the Engineer diploma degree in Electrical Engineering from ENSI (CAEN), a DEA degree in automatic and signal processing from the University Paris XIin1979, a Doctor Engineer in 1981, and a Ph.D. degree in signal processing from the University ParisXI in 1986, France. He is currently a Professor at the University of Sciences and Technology of Oran, Algeria. His research interests are focused on high-resolution spectral array processing methods, and detection and tracking moving objects. In 1990 he received the senior award from the IEEE Signal Processing Society in the Spectrum Estimation Technical area. He can be contacted at email: ouamri@yahoo.com.