Received: September 29, 2024. Revised: November 30, 2024.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

Innovative Road Object Detection and Distance Estimation Framework Using Monocular Cameras for Advanced Driver Assistance Systems

Omar Bouazizi[1]* **Chaimae Azroumahli[2]** **Aimad El Mourabit[1]**

[1]*Engineering Data and Systems Team, National School of Applied Sciences Tangier, Abdelmalek Essaadi University, Morocco*
[2]*Mathematics, Data Sciences and Emerging Technologies, Faculty of Sciences and Techniques, Settat, Hassan 1st University, Casablanca, Morocco*
* Corresponding author's Email: omar.bouazizi@etu.uae.ac.ma

Abstract: Advanced Driver Assistance Systems (ADAS) rely on accurate road object detection, tracking, and distance estimation to enhance road safety. This paper presents ROD-YOLOv8, an innovative framework that leverages a monocular camera for these tasks within ADAS. By integrating the YOLOv8 model with transfer learning, the framework achieves high accuracy in road object detection. The model was retrained on the BDD100k dataset and further validated using COCO, KITTI, and PASCAL VOC datasets for comparison with existing models, achieving a mean average precision (mAP) of 0.782 and an F1 score of 0.874 on the COCO dataset. Object tracking is maintained through the Bot-Sort algorithm, ensuring consistent tracking of detected objects and providing continuous monitoring and future location prediction. Advanced camera calibration techniques enable accurate distance estimation between the camera and the detected objects, resulting in a mean absolute distance error of just 2.4 meters. Operating at an impressive 83 frames per second (FPS), ROD-YOLOv8 showcases its real-time capabilities, contributing to safer autonomous and assisted driving experiences. The proposed object detection model demonstrates improved performance compared to existing methods, particularly outperforming other methods like Mobile NET and earlier versions of YOLO in terms of precision, recall, and speed, measured in Frames Per Second (FPS), making it highly suitable for integration into ADAS applications.

Keywords: Object detection, Object tracking, Monocular camera, YOLOv8, BotSort, Distance estimation.

1. Introduction

Advanced Driver Assistance Systems (ADAS) has increasingly become popular in recent years, largely for its significant contribution to enhancing driver assistance and road safety [1]. ADAS includes a range of applications designed to increase situational awareness and prevent road accidents [2]. These applications, however, depend fundamentally on the capabilities of ADAS to detect, track, and estimate the distance of road objects.

Deep learning has become increasingly prominent in developing object detection models for ADAS [3−5]. These algorithms enhance ADAS performance by improving the systems' abilities to detect, classify, and track objects. More specifically, Convolutional Neural Networks (CNNs), a specialized type of deep neural network, are considered the reason for this improvement [6]. They automatically and adaptively learn spatial hierarchies of features from input images. In fact, by processing images through convolutional and pooling layers, CNNs iteratively adjust learning weights during training to optimize feature extraction and recognize complex patterns in data [7, 8]. This capability enables models trained with CNN algorithms to leverage vast annotated datasets, thereby significantly enhancing the accuracy and reliability of object detection in ADAS. Among CNN-based algorithms, the You Only Look Once (YOLO) series stands out for real-time object detection due to its high detection accuracy and fast inference speeds. Notable models in this series include YOLOv5 [9]

which introduced techniques like auto-learning bounding box (BBox) anchors, mosaic data augmentation, and adaptive image resizing. YOLOv7 [10] extended Efficient Layer Aggregation Networks, which allowed the model to efficiently learn complex representations. It was also designed with an emphasis on speed, making it one of the fastest and most accurate models for real-time object detection upon release. YOLOv8 [11] presented an anchor-free approach, reducing the need for extensive manual tuning of anchor boxes. YOLOv9 [12] introduced enhancements in multi-task learning, enabling the model to handle diverse detection tasks simultaneously while optimizing resource utilization. Last, YOLOv10 [13], made strides in efficiency by integrating Neural Architecture Search techniques to automatically discover optimal network architectures tailored for specific tasks.

YOLOv8, although it is continually being refined, has shown improvements in computational efficiency and accuracy over its predecessors. For example, previous models such as YOLOv7 utilized anchor boxes to predict BBox and object classes directly from input images in a single step [10]. However, these models required the manual specification of anchor box sizes and shapes, which limited their adaptability across various scenarios. In contrast, YOLOv8 has introduced the use of free anchor boxes, allowing the network to learn optimal parameters during training [14]. This development significantly enhances flexibility and performance in different scenarios, marking a significant progression toward more efficient and adaptable object detection mechanisms for ADAS applications.

Furthermore, the application of transfer learning with YOLOv8 significantly benefits the development of domain-specific object detection models [15]. By retraining pre-trained models on new, target datasets, this approach utilizes the pre-learned features from open-domain datasets, applying them to specific road datasets [16]. This method reduces the computational resources required for training and maintains or even enhances the model's performance. Furthermore, the application of transfer learning with YOLOv8 significantly benefits the creation of domain-specific object detection models [16]. This technique involves retraining pre-existing models on fresh, target datasets, utilizing previously learned features from broader datasets, and adapting them to specialized road datasets. Not only does this approach reduce the computational resources needed for training, but it also sustains and enhances the model's effectiveness [17].

Besides road object detection, distance estimation plays a crucial role in ADAS applications. Distance Estimation methods fall into two categories: sensor-based and vision-based approaches [18]. Sensor-based methods employ a variety of technologies, including radar and LiDAR, to accurately measure distances between the vehicle and other objects. On the other hand, vision-based methods depend solely on vision sensors, primarily cameras. The latest method gained significant attention in recent research due to the widespread availability of cameras in various industries [18, 19]. Within vision-based distance estimation, there are two main approaches: stereo camera-based and monocular camera-based [20, 21]. The stereo camera-based approach aims to determine the depth of each pixel by aligning pairs of stereo images. Conversely, the monocular camera-based approach, which uses a single camera, estimates distances by considering various factors, such as camera settings, real-scale factors, and the width of the vehicle's front [18]. This approach balances accuracy with the benefit of low-cost processing. Several researchers have made contributions to the field of monocular camera-based distance estimation. Ka Seng et al. [22] proposed a novel method for estimating distance in unregulated 3D environments to aid in navigation towards specified target objects using a monocular camera. This approach minimizes the computational load on edge devices by utilizing mathematical models to produce reliable estimations. By incorporating modern object detection models, the authors' method demonstrates a commendable mean absolute percentage error. Similarly, Seungyoo et al. [23] introduced a framework for estimating the distance between a vehicle and objects ahead using an onboard webcam. This method combines an object detector, which identifies object classes and BBox, with a depth estimator to generate the image's depth map, showcasing superior performance over many existing studies. Ahmed et al. [24] developed a single-view geometry-based algorithm for estimating the relative distance between road vehicles, integrating seamlessly with the lane and vehicle detection models of existing ADAS technologies. These works paved the way for more accessible distance estimation methods, promising to impact road safety and efficiency significantly.

In this work, we introduce and study ROD-YOLOv8, a lightweight model capable of detecting road objects, tracking the detected road objects across multiple frames then estimating the distance between a vehicle and the tracked objects using a monocular camera. The primary contributions of this work are depicted in Fig. 1. We present a framework that includes a road object detector, an object tracker, and a distance estimator. The road object detector

identifies objects within a video frame, providing essential information such as object class, BBox coordinates, confidence score, and scale factor. The object tracker then employs the Bot-Sort algorithm across consecutive frames to maintain correspondence between detected objects. Finally, the distance estimator uses a calibration procedure inspired by the HARRIS algorithm. This algorithm enables accurate distance estimation between the detected object and the monocular camera mounted on the moving vehicle, utilizing the pinhole camera model.

The rest of this paper is organized as follows: Section 2 outlines the approach employed in developing the road object detection model. Section 3 explains the Bot-Sort algorithm applied for object tracking. Section 4 elaborates on the pinhole camera model and HARRIS algorithm utilized for distance estimation. Section 5 summarizes the experimentation and evaluation steps, followed by a discussion of the obtained results. Finally, the conclusion wraps up this study.

2. Road object detection

The first step of our work involves creating the road object detection model. This model detects various road objects, namely cars, people, bikes, motorcycles, buses, riders, trucks, trains, traffic signs, and traffic lights. We retrained a pre-trained YOLOv8 model using the BDD100k dataset to enhance its performance for this specific task. YOLOv8 was selected for its superior balance between speed and accuracy, making it ideal for real-time applications.

Additionally, the BDD100k dataset [25], a comprehensive benchmark for autonomous driving, was leveraged to improve the model's robustness and adaptability in detecting a wide range of road objects and signs under diverse conditions.

The subsequent subsections describe the following approach, including details about the training and validation datasets, re-training parameters, and the metrics used for evaluation purposes.

2.1 Dataset description

Our approach relied on the BDD100k annotated datasets to create the trained models. Table 1 summarizes its statistics. This dataset originally included four coordinates for each BBox: x_{min}, x_{max}, y_{min} , y_{max}. To adapt the format to meet the requirements of YOLOv8, we converted the coordinates to its format. The new coordinates are the width $w_{B.Box}$, the height $h_{B.Box}$, and the coordinates of the BBox x_{center} and y_{center} . They were calculated using the following equations:

$$w_{B.Box} = \frac{x_{min} - x_{max}}{w_{Img}}, h_{B.Box} = \frac{y_{min} - y_{max}}{h_{Img}} \quad (1)$$

$$x_{center} = \frac{x_{min} + x_{max}}{2 * w_{Img}}, y_{center} = \frac{y_{min} + y_{max}}{2 * h_{Img}} \quad (2)$$

where w_{Img} and h_{Img} are the width and height of the input images. The decision to use the BDD100k dataset was justified by the fact that it only contains classes relevant to ADAS. This dataset helped to reduce the number of classes from the original 80 classes of the YOLOv8-n pre-trained model. In addition, we set the training and validation datasets. Regarding the image size, we had to resize the BDD100k images from $1\,280 \times 720$ pixels to a fixed shape of 640×360 pixels.

2.2 Transfer learning with YOLOv8

The current work utilized transfer learning to develop an object detection model using the pre-trained YOLOv8 architecture. As shown in Fig. 1, the process of creating the model included training on the BDD dataset. In addition, to ensure optimal performance suited to the dataset, hyperparameter tuning was carried out using Genetic Evolution and Mutation techniques provided by the developers of YOLO [26]. Similar to its predecessor, YOLOv8 offers various model scales identified as n, s, m, l, and x. Each scale pertains to a network scaling factor intended to balance object detection precision with a lightweight architecture [27]. Larger scales generally deliver higher accuracy, while smaller scales achieve higher FPS [28].

Among these, YOLOv8s is noted for its lightweight and higher FPS at the expense of some accuracy, making it particularly well-suited for real-time applications [28].

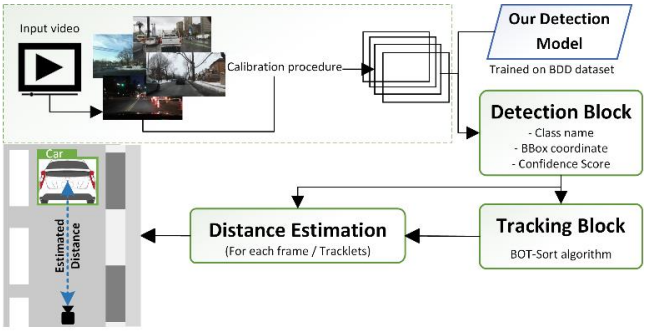


Figure. 1 Overview of Our Proposed Framework for ROD-YOLOv8: Joint Road Object Detection and Distance Estimation

Table 1. Transfer learning Datasets Statistics

Label Name	Train Size	Validation Size
Car	713 211	102 506
Person	11 672	1 597
Bike	100 797	7 210
Motor	3 002	452
Bus	91 349	130 262
Rider	4 517	649
Truck	186 117	26 885
Train	239 686	34 908
Traffic sign	136	15
Traffic Light	29 971	4 245

The initial stage of retraining a YOLOv8s model involves preparing a custom dataset tailored to create a domain-specific object detection model, such as road object detection.This custom dataset consists of images containing detectable objects along with their BBox annotations. Subsequently, the feature extraction process begins, during which pretraining involves freezing the lower layers to retain valuable learned features [29]. Fig. 2 illustrates the standard transfer learning approach used to develop the object detection model while fine-tuning the hyperparameters with genetic evolution and mutation.

For retraining, we initiate by removing the pre-trained models' top layers and replacing them with a new set of convolutional layers tailored for predicting the BBox coordinates and confidence scores of road objects. The updated weights of the final detection models are then fine-tuned through backpropagation, leveraging the target dataset [30]. To ensure the efficiency of transfer learning, it is important to verify that the main objects to detect are well-represented in the training datasets (see Table 1). The principle of genetic evolution and mutation employed for hyperparameter tuning involves the exploration of an optimal set of hyperparameters by iteratively making random adjustments to the initial set, thus generating new candidates for evolution. In this process, we designate the mAP score as the metric for evaluating the model's performance.

To address the issue of reducing the computational cost during training, we took measures to reduce the computational cost during training [11]. One approach was to downscale the size of the input images while being cautious not to lose the feature extraction details crucial for accurate detection and classification. In addition, we set 100 iterations as a tuning budget as it is computationally expensive. The initial hyperparameters used for the creation of the detection model were as follows: a batch size of 4, 50 epochs, and a learning rate of 0.001. The training hardware setup comprised an NVIDIA GeForce RTX 2060 GPU, supported by 12 workers, and 16GB of RAM.

Fig. 3 presents the results obtained during the training of our detection model. In Fig. 3(a), we showcase some output images with their BBox from a training instance. Meanwhile, in Fig. 3(b), we display the distribution of two evaluation metrics: the F1 score and the confidence score. The F1 score serves as a measure of the model's precision and recall balance, while the confidence score represents the model's confidence level in its predictions. The F1 and confidence scores are calculated by YOLO using the following equations:

$$\mathrm{F1} = \frac{2*(Precison*Recall)}{(Precison+Recall)} \tag{3}$$

$$\mathrm{Confidence} = \frac{sum\ of\ Trust\ (Obj)}{sum\ of\ Trust\ (All)} \tag{4}$$

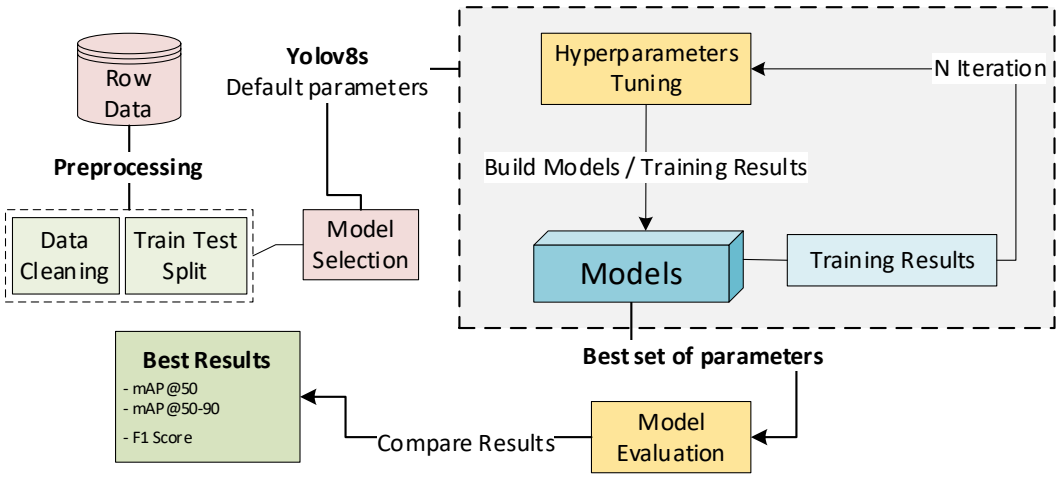
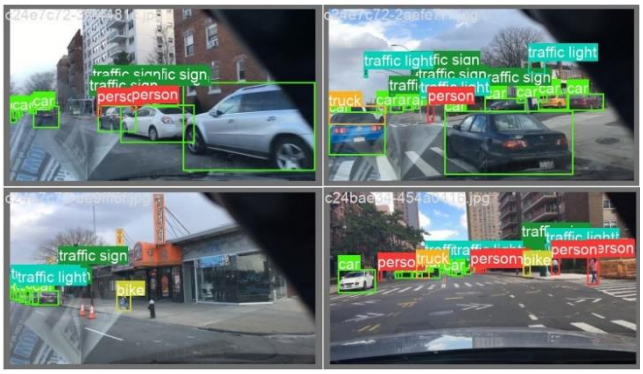
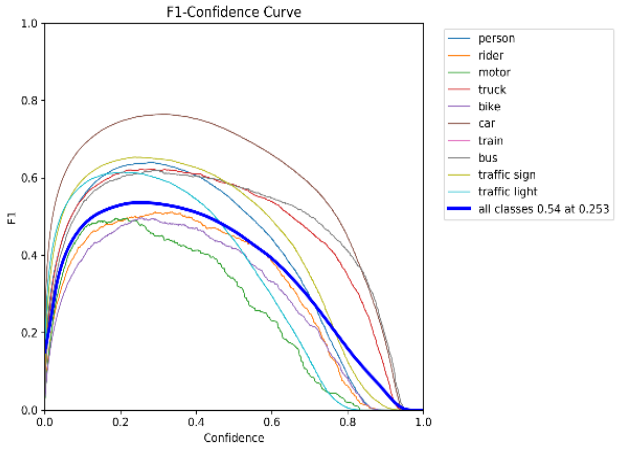


Figure. 2 Process of retraining the road object detection model with hyperparameter evolution



(a)



(b)

Figure. 3 Training Results of the Object Detection Model: (a) Inference on Test Images During Training; (b) F1-Confidence Curve for each Class

The visualization of these two scores offers insights into the model's performance at varying confidence thresholds and provides a comprehensive overview of its detection accuracy and reliability. The training of our detection model gives high-density regions around specific confidence values for the different classes, affirming the model's detection high performance.

Fig. 4 displays the confusion matrix generated by YOLO after training both detection models. This visualization was instrumental in evaluating the performance of both models, as it allowed us to compare the model's predictions with the true labels from the dataset. By analyzing the confusion matrix, we identified particular classes that the model had difficulty predicting accurately, notably the "train" class. To optimize our model's performance, we recognized the need for data augmentation specifically for the train labels in our training dataset.

3. Object tracking

Object Tracking is a pertinent task for ADAS applications. It aims at locating objects within video frames [31]. Various object-tracking algorithms have been developed for ADAS. These algorithms typically utilize a combination of sensor data, such as lidar or camera inputs, and mathematical models to detect the position of a defined object in a sequence of frames [1].

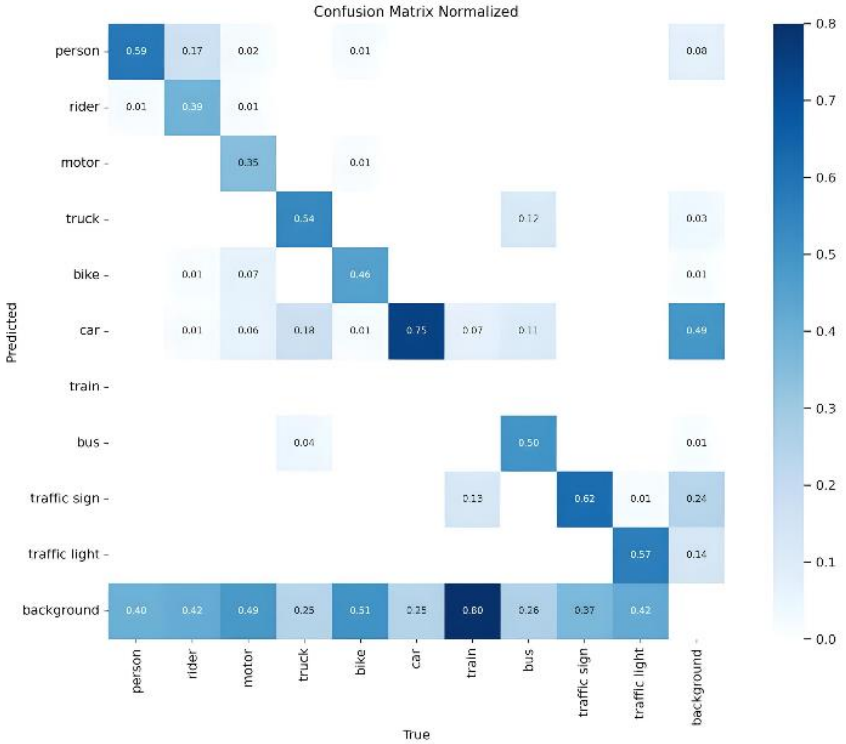


Figure. 4 Road Object Confusion Matrix for Detection

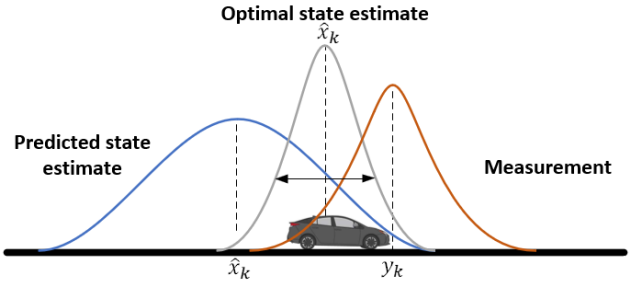


Figure. 5 Kalman Filter optimal state estimator

One commonly used approach in these algorithms is the Kalman Filter (KF), which operates by iteratively estimating the state of a dynamic system from a series of noisy measurements. The state vector $x_k = [x_c(k), y_c(k), w(k), h(k), \dot{x}_c(k), \dot{y}_c(k), \dot{w}(k), \dot{h}(k)]^T$ depicts the BBox coordinates. The KF maintains two main Equations; the predicted state $\hat{x}_{k|k-1}$ at the measurement of the timestep k to $k-1$, and the measurement update $\hat{x}_{k|k}$ that corrects the predicted state based on the corrected measurement y_k [20]. The predicted state $\hat{x}_{k|k-1}$ is calculated using the following equation:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k\_1|k-1} + B_k u_k + w_k \quad (5)$$

where F_k, B_k, u_k and w_k denote the state transition matrix, control input matrix, control input, and process noise respectively.

The measurement update $\hat{x}_{k|k}$ is calculated using:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(y_k - H_k \hat{x}_{k|k-1}\right) \quad (6)$$

where K_k, H_k and y_k present the Kalman gain state, the measurement matrix, and the corrected measurement update at the time k respectively.

Fig. 5 illustrates how the KF combines the predictions of the system dynamics with the measurements, estimating a joint probability distribution over the variables for each timeframe, and providing an optimal estimate of the system state [32].

The SORT algorithm and its variants, such as Deep-SORT and Bot-SORT, are recognized as some of the earliest tracking algorithms employed in ADAS [33].

They utilize the KF for frame-by-frame data coherence, in conjunction with the Hungarian algorithm, which aims to assign a set of objects to another set of objects while minimizing the total cost or distance, as depicted in the following equation:

$$MIN_{cost} = \min_P \sum_{i,j} C_{ij} P_{ij} \quad (7)$$

Where P is the permutation matrix, and C_{ij} is the cost of assigning the i^{th} object to the j^{th} track.

As depicted in Fig. 1, our approach leverages YOLOv8 for object detection and tracking, which offers the use of two tracking algorithms: Bot-Sort and Byte-Track. In our case, we aimed to achieve a balance between real-time performance for ADAS, accuracy, and computational resources. Following a comprehensive literature review, we inferred that Byte-Track, with its reliance on CNNs [34], demands significant computational resources for training and inference, thereby constraining its applicability in resource-limited environments or scenarios with stringent real-time requirements [27, 35, 36]. Thus, we opted for the Bost-Sort algorithm [37]. Bot-Sort operates by decentralizing decision-making processes DMP as represented by the following equation:

$$DMP = \underset{i}{\operatorname{argmin}} \sum_j C_{ij} \quad (8)$$

This decentralized approach handles large amounts of data and adapts to changes in object trajectories or environmental conditions, making it suitable for scenarios with a high density of moving objects like road scenes [38]. Moreover, Bot-Sort implements the Camera Motion Compensation solution to avoid the problem of the BBox stabilization and rectify the KF calculations. The overall pipeline of the object tracking approach of our methodology is depicted in Fig. 6.

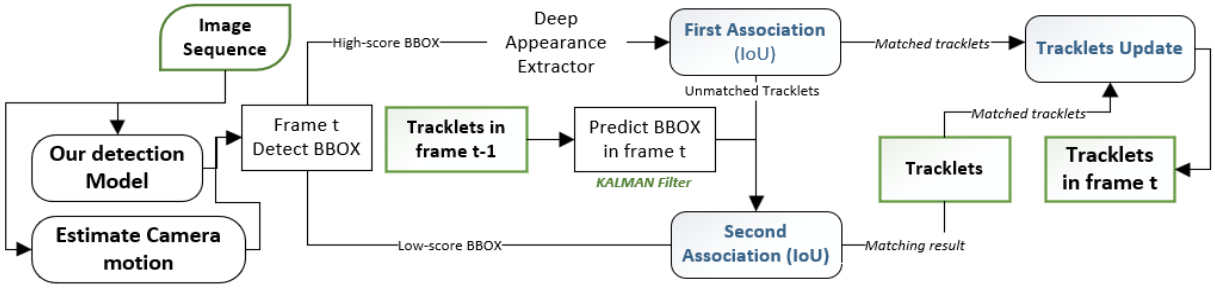


Figure. 6 Bot-Sort Tracker Pipeline

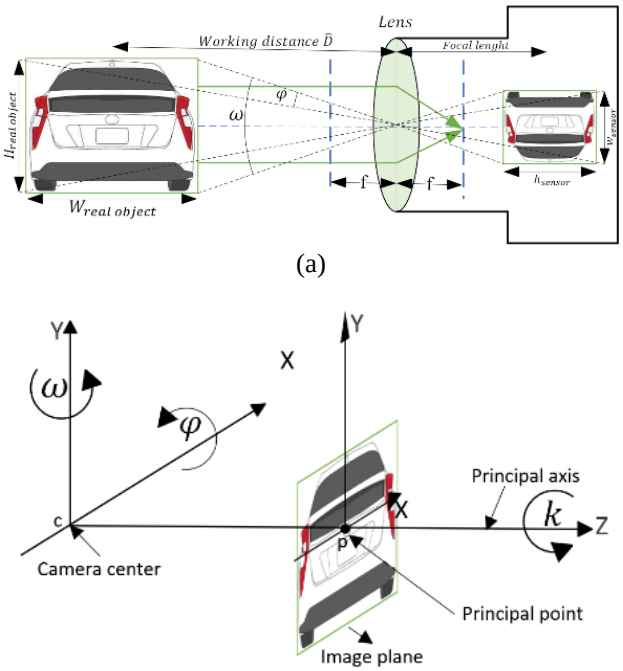


Figure. 7 (a) Schematic diagram of the imaging geometry (b) Geometric schematic and coordinates system of the pine-hole model

4. Distance estimation using a pinhole model

Distance estimation is a fundamental task in ADAS applications. It aims to accurately determine the distance between a vehicle and nearby road objects. Various methodologies for distance estimation rely on a fusion of sensors typically integrated into vehicles. However, the use of monocular cameras for distance estimation has gained significant attention due to their inherent flexibility [18, 19, 23]. This methodology can employ optical flow, depth information extracted from images, and geometric principles.

Employing a monocular camera for distance estimation should consider several key parameters, including the camera's focal length, the actual size of objects, and the resolution of the captured images. Additionally, accurate distance estimation requires proper camera calibration and perspective correction to account for camera angles and perspective distortions that can introduce measurement inaccuracies. This means converting 2D detected object coordinates to 3D real-world coordinates [18, 39]. Since using a monocular camera leads to the loss of 3D information. To calculate the distance, we need to calibrate the camera to access the matrix of intrinsic parameters to recover this 3D information.

4.1 Pinhole imaging model

The pinhole imaging model is considered one of the commonly used methods to estimate a distance using a monocular camera and a simplified camera projection model.

Fig. 7 provides a geometric schematic representation of this model's principles. This geometric model can be used to determine the 3D geometric position of a real-world road object using its corresponding detected road object in the image. While the parameters of the geometric model are the intrinsic parameters of the camera. The following equation:

$$D = \frac{fW}{w} \tag{9}$$

can be used to estimate the distance D according to the pinhole imaging model. f is the focal camera length, W is the real-world width of an object, while w is the BBox width of the imaging plane that is converted to the pixel coordinate system of an image. Further, to estimate W, camera calibration is used to convert a 2D detected object to its corresponding 3D geometric position of an object on the surface to finally estimate the real-world distance [40].

4.2 Camera calibration and coordinates for distance estimation

Our work was inspired by the work described in [18], where they use the HARRIS corner detector algorithm [41] to extract sub-pixel coordinates and Zhang's method [42] to calibrate the camera and determine the pose of the pinhole camera. The HARRIS algorithm begins by shifting a processing sub-window across an image centered on a point and expanding. If the change exceeds a specified threshold, the point is identified as a corner point.

Calibration determines the desired number of corner points, ensuring balanced distribution across the image. To manage densely clustered points in certain regions, a maximum limit is set for corner points per area, preventing excessive low-threshold corner points elsewhere. In Zhang's calibration method, pixel coordinates (x, y) of the corner points on a calibration board are obtained using an image detection algorithm. The calibration board's checkerboard grid serves as the world coordinate system with known physical coordinates W. By predefining the calibration board's coordinate system and grid size, it is possible to calculate the physical coordinates of each corner point in the world system. This information is then used to calibrate the camera

and determine its internal and external reference matrices [18, 19].

To calculate the distance, the transformation between camera coordinates and image coordinates is required. The following equation:

$$x = Kx_c \tag{10}$$

shows how the coordinates of the actual measurement x are calculated, where x_c is the camera coordinates and K is presented in the equation below:

$$K = \begin{pmatrix} f_x & 0 & x_{center} \\ 0 & f_y & y_{center} \\ 0 & 0 & 1 \end{pmatrix} \tag{11}$$

In this equation, Kdenotes a matrix derived from the literature [18], with f_x and f_y representing the camera's focal length coordinates, and x_{center} and y_{center} are part of the BBox coordinates of the detected object.

Following previous works [18, 19], estimating the distance requires the conversion between camera coordinates and image coordinates. The following equation:

$$X_c = [\, R \mid T \,]X_w \tag{12}$$

calculates the real-world coordinates $X_c = (x_c, y_c, z_c)^T$ and the transformation of camera coordinates. The pose of an object, relative to the camera coordinate system, could be described in terms of the translation vector T and the rotation matrix R as described in the following equation:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} \cos\omega & 0 & -\sin\omega \\ 0 & 1 & 0 \\ \sin\omega & 0 & \cos\omega \end{bmatrix} \times \begin{bmatrix} \cos\kappa & -\sin\kappa & 0 \\ \sin\kappa & \cos\kappa & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

Our distance estimation approach comprises the following steps: initially, we obtain the image coordinates x through Eq. (10) and a projection function as described in [18]. Subsequently, we determine X_c using an unprojection function from image coordinates to camera coordinates. Finally, distance estimation is represented by the Euclidean distance of the 3D point X_c :

$$\widehat{D} = \|X_c\| = \sqrt{{x_c}^2 + {y_c}^2 + {z_c}^2} \tag{14}$$

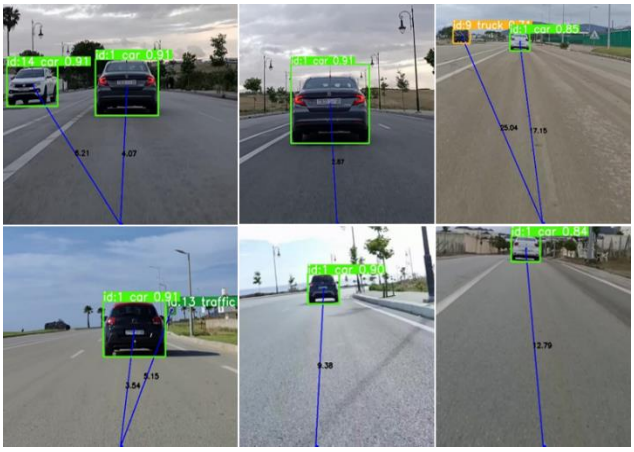


Figure. 8 Inference with the ROD-YOLOv8 Framework

5. Experimental results

All experiments were conducted on systems running Ubuntu 20.04. The system was developed using the Python programming language. The first machine used for testing is equipped with an NVIDIA GeForce RTX 2060 GPU and 32GB of RAM. The second machine is a Jetson Xavier AGX with 16GB of RAM.

To evaluate the effectiveness of ROD-YOLOv8, we conducted a thorough evaluation, which included examining the object detection model's performance, assessing the accuracy of distance estimation, and analyzing the system's FPS. Examples of inference using our system ROD-YOLOv8 are presented in Fig.8.

5.1 Object detection

The evaluation of the object detection system was conducted using precision, recall, F1 score (as described in Section 1.1), and the mAP as depicted in:

$$mAP = \frac{\sum_i^n AP_i}{n} \tag{15}$$

where AP_i represents the average precision for class i and n is the number of classes. These metrics were chosen for their ability to measure the correctness of predictions and the model's capacity to detect relevant objects within the dataset.

Specifically, the mAP provides a holistic view of the model's precision and recall performance across multiple classes, offering insight into its ability to identify and classify objects within the 10 chosen categories accurately [7]. The testing was performed using three additional validation datasets: COCO [43], KITTI [44], and PASCAL VOC [45].

Table 2. Comparison Between the Performance of Our Object Detection Model and Other Existing Models.

Model	Dataset	mAP	Precision	Recall	F1
ROD-YOLOv8 [Ours]	COCO	**78.25**	87.4	83.3	**87.4**
YOLOv5n [26]	COCO	78.7	85.2	69.0	76.2
YOLOv5s [26]	COCO	**83.8**	86.6	77.4	81.7
YOLOv7-tiny [10]	COCO	74.2	**88.2**	76.1	81.7
ROD-YOLOv8 [Ours]	PASCAL VOC	**64.26**	51.22	78.49	**61.98**
YOLOv3 [46]	PASCAL VOC	55.81	42.29	68.48	52.29
MobileNetV1-YOLOv3 [46]	PASCAL VOC	6.27	21.95	17.90	19.72
MobileNetV2-YOLOv3 [46]	PASCAL VOC	13.26	27.48	28.34	27.90
ROD-YOLOv8 [Ours]	KITTI	79.65	**68.43**	**96.14**	**79.95**
YOLOv5n [47]	KITTI	73.1	51	93	-
YOLOv5s [47]	KITTI	78.5	59.4	93.4	-
YOLOv5m [47]	KITTI	78.2	61.5	93.6	-
YOLOv5l [47]	KITTI	**81.7**	64.1	93.6	-

Before testing, the class labels of these datasets were preprocessed and adapted to align with the label format of the BDD100k dataset, which was used for retraining the model. This preprocessing step ensured consistency across all datasets, allowing for a fair evaluation of the model's performance in diverse traffic scenarios.

The evaluation results of ROD-YOLOv8, as shown in Table 2, demonstrate its strong performance and suitability for ADAS. Our model achieved a mAP of 78.25 and an F1 score of 87.4 on the COCO dataset, both of which are competitive with, and in some cases superior to, other state-of-the-art models in the literature. Notably, ROD-YOLOv8 exhibits a balanced performance with high precision (87.4%) and recall (83.3%), outperforming several models such as YOLOv5n and YOLOv7-tiny, particularly in terms of F1 score, which is crucial for applications requiring both accuracy and consistency. When compared to models tested on the COCO dataset, ROD-YOLOv8 offers a strong balance between precision and recall, achieving an overall higher F1 score than YOLOv7-tiny and YOLOv5s. Although YOLOv5s has a slightly higher mAP, our model's improved F1 score underscores its reliability and robustness in real-world scenarios. These results affirm that ROD-YOLOv8 is a competitive and effective model for road object detection, with particular strengths in precision and overall accuracy, making it well-suited for integration into ADAS frameworks.

In addition, ROD-YOLOv8 achieves the highest mAP (64.26%) and F1 Score (61.98%) on PASCAL VOC, indicating strong object detection capabilities with balanced precision and recall. On the KITTI dataset, ROD-YOLOv8 achieves high precision (68.43%) and recall (96.14%), resulting in a competitive F1 Score of 79.95%, closely rivaling YOLOv5l in mAP. These results suggest that ROD-YOLOv8 is an effective, versatile model for object detection across various environments, especially suited for applications like autonomous driving where detection accuracy is essential.

5.2 Distance estimation

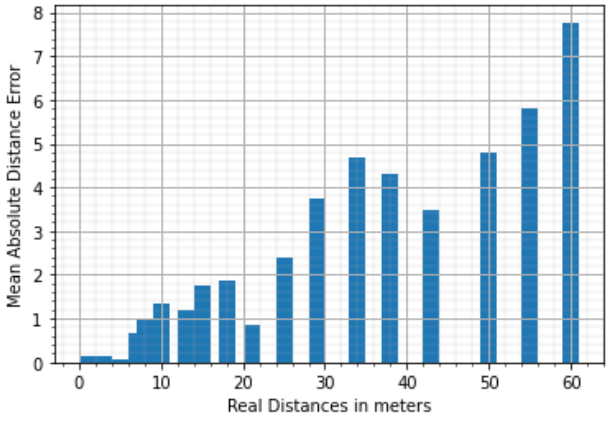
The evaluation of our distance estimation model's performance adhered to the criteria outlined in [48], utilizing two principal metrics: the Mean Absolute Distance Error (MAD) and the Mean Relative Distance Error (MRD), as defined in the following equations:

$$MAD = \frac{1}{n}\sum_{i=1}^{n}\left|d_i - \widehat{d}_i\right| \tag{16}$$

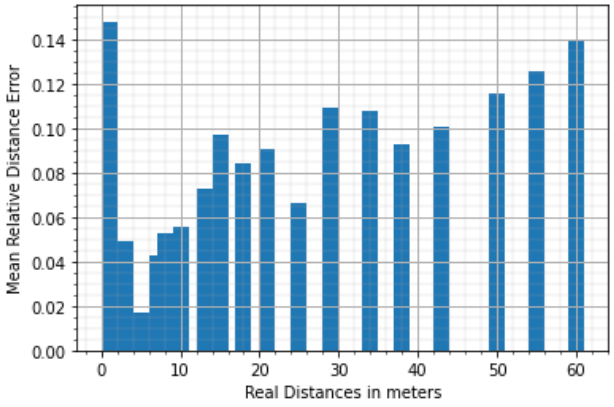
$$MRD = \frac{1}{n}\sum_{i=1}^{n}\frac{|d_i - \widehat{d}_i|}{\max(d_i,1)} \tag{17}$$

where n is the number of instances, and $d_i, \widehat{d}_i$ are respectively the real distance and the predicted distance for instance i. The MAD measures the average absolute error between the estimated distances and the ground truth distances. This metric gives a clear indication of the average deviation of our distance estimates from the actual values. The MRD however offers a normalized perspective by considering the relative error concerning the ground truth distances, it provides insights into the proportional accuracy of our estimates across different distance ranges.

For our system, the MAE achieved a value of 2.412, while the MRD was constrained to a value of 0.083. Fig. 9 illustrates the absolute and relative distance errors, revealing that MAD increases in a polynomial trend, with an exception at the 60-meter



(a)



(b)

Figure. 9 The absolute and relative distance error results. (a) Dependency of MAD Error on the object's real distance. (b) Dependency of MRD Error on the object's real distance

mark, indicating a higher error margin for objects at a distance higher than 50 meters. The MRD, conversely, remained nearly constant across all distances, except at one meter. This anomaly can be attributed to the challenge posed by minor distance errors at close range, such as a 20 cm error at one meter translating to an MRD of 0.15, compounded by the inherent inaccuracies in measuring short distances.

Further exploration into the ROD-YOLOv8's distance estimation performance unveiled errors particularly with classes like traffic signs and vehicles in other lanes, primarily attributed to deformation. Road objects in an image plan can undergo deformation, such as vehicles turning or pedestrians walking, which poses detection and tracking challenges for ADAS. To address these issues, perspective transformation, associated with changes in viewpoint, can be employed as an effective solution [1]. Although this transformation does not maintain parallelism, length, or angle, it ensures that lines remain straight, thereby offering a potential solution to deformation-induced errors.

5.3 FPS and real-time application

The final phase of our evaluation focused on assessing the real-time performance of the ROD-YOLOv8 system by measuring its FPS. Our road object detection model alone achieved an impressive 97 FPS on the Jetson Xavier AGX, demonstrating its capability for high-speed processing essential for real-time applications. However, when integrated with distance estimation, the FPS dropped to 83, which, although lower, still satisfies the real-time performance criteria crucial for ADAS applications.

Comparatively, as shown in Table 3, the ROD-YOLOv8 model outperforms several existing models such as YOLOv4 and Faster R-CNN in terms of FPS,

Table 3. Comparison of FPS Performance Between ROD-YOLOv8 and Other Existing Models.

Model	FPS	Hardware
Faster R–CNN [49]	6	NVIDIA GeForce 1060 GPU
YOLOv4 [49]	35	NVIDIA GeForce 1060 GPU
YOLOv4 CSPDarknet-53 [49]	38	Nvidia GeForce RTX 2080 Ti
SSD VGG-16 [49]	43	Nvidia GeForce RTX 2080 Ti
Faster R–CNN [49]	92	Nvidia GeForce RTX 2080 Ti
ResNet50 [49]	25	Nvidia GeForce RTX 2080 Ti
YOLOv2 DarkNet-19 [49]	45.5	Nvidia GeForce RTX 2080 Ti
YOLOv5[50]	43.59	Jetson Xavier AGX
YOLOv5[50]	23.17	Jetson Xavier Nx
YOLOv5[50]	6.4	Jetson Nano
YOLOv7 [35]	53	NVIDIA Tesla V100
ROD-YOLOv8 [Ours: Detection]	**97**	Jetson Xavier AGX
ROD-YOLOv8 [Ours: Detection + Distance estimation]	83	Jetson Xavier AGX
ROD-YOLOv8 [Ours: Detection]	50	Nvidia GeForce RTX 2060
ROD-YOLOv8 [Ours: Detection + Distance estimation]	41	Nvidia GeForce RTX 2060

even when using different hardware configurations. For instance, our model's performance on the Jetson Xavier AGX surpasses YOLOv4's 35 FPS on the GeForce 1060 and aligns closely with the high FPS achieved by Faster R-CNN on more advanced hardware.

These results highlight the ROD-YOLOv8's potential to deliver real-time object detection and distance estimation, critical for enhancing the reliability and safety of ADAS technologies.

Despite the challenges of integrating detection and distance estimation, the system maintains robust performance, making it a strong candidate for practical deployment in safety-sensitive environments.

6. Conclusion

In this paper, we presented ROD-YOLOv8, a framework designed to enhance road safety and navigation in autonomous and assisted driving environments. The framework integrates three key components: road object detection, object tracking, and distance estimation. The detection model accurately identifies and classifies road objects, providing essential data such as BBox coordinates and confidence scores. The Bot-Sort-based tracking model ensures consistent object identification across frames, crucial for predicting object movements. The distance estimation model, using class-specific calibration, reliably estimates the distance between objects and the vehicle-mounted monocular camera. ROD-YOLOv8 demonstrated strong performance, achieving a mAP of 78.2, an F1 score of 87.4, and an FPS of 97 on the Jetson Xavier AGX.

The system's distance estimation showed a MAD error of 2.412, with most errors occurring for distant or deformed objects beyond 50 meters. In summary, ROD-YOLOv8 represents a significant advancement in road object detection and distance estimation using monocular cameras, offering a more accurate and reliable solution for ADAS technologies, thereby contributing to safer and more efficient driving experiences.

Data Availability

The source codes and datasets generated and analyzed during this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare that they have no conflict of interest.

Author Contributions

Omar Bouazizi contributed to the conceptualization, methodology, software development, validation, formal analysis, and writing of the original draft. Chaimae Azroumahli was involved in the conceptualization, methodology, and writing – review & editing. Aimad El Mourabit contributed to the conceptualization, methodology, supervision, and writing – review & editing.

References

[1] V. Malligere Shivanna and J. I. Guo, "Object Detection, Recognition, and Tracking Algorithms for ADASs—A Study on Recent Trends", *Sensors 24*, No. 1: 249, doi: 10.3390/s24010249.

[2] P. Shunmuga Perumal et al., "LaneScanNET: A deep-learning approach for simultaneous detection of obstacle-lane states for autonomous driving systems", *Expert Syst. Appl.*, Vol. 233, p. 120970, 2023, doi: 10.1016/j.eswa.2023.120970.

[3] G. S. A. Mohammed, N. M. Diah, Z. Ibrahim, and N. Jamil, "Vehicle detection and classification using three variations of you only look once algorithm", *Int. J. Reconfigurable Embed. Syst.*, Vol. 12, No. 3, pp. 442-452, 2023, doi: 10.11591/ijres.v12.i3.pp442-452.

[4] L. Dai et al., "A deep learning system for detecting diabetic retinopathy across the disease spectrum", *Nat. Commun.*, Vol. 12, No. 1, p. 3242, 2021, doi: 10.1038/s41467-021-23458-5.

[5] S. Antar, H. K. Hussein, M. H. Abdel-rahman, and F. Ghaleb, "Robust Object Recognition with Deep Learning on a Variety of Datasets", *Int. J. Intell. Eng. Syst.*, Vol. 16, No. 4, pp. 436-449, 2023, doi: 10.22266/ijies2023.0831.35.

[6] M. T. Ahad, Y. Li, B. Song, and T. Bhuiyan, "Comparison of CNN-based deep learning architectures for rice diseases classification", *Artif. Intell. Agric.*, Vol. 9, pp. 22-35, 2023, doi: 10.1016/j.aiia.2023.07.001.

[7] O. Bouazizi, C. Azroumahli, A. El Mourabit, and M. Oussouaddi, "Road Object Detection using SSD-MobileNet Algorithm : Case Study for Real-Time ADAS Applications", *J. Robot. Control*, Vol. 5, No. 2, pp. 551-560, 2024, doi: 10.18196/jrc.v5i2.21145.

[8] M. S. Sawah, S. A. Taie, M. H. Ibrahim, and S. A. Hussein, "Deep Neural Network-based Approach for Accurate Vehicle Counting", *Int. J. Intell. Eng. Syst.*, Vol. 16, No. 2, pp. 255-267, 2023, doi: 10.22266/ijies2023.0430.21.

[9] T. Mahendrakar et al., "Performance Study of

YOLOv5 and Faster R-CNN for Autonomous Navigation around Non-Cooperative Targets", In: *Proc. of IEEE Aerosp. Conf. Proc.*, Vol. 2022, pp. 1-12, 2022, doi: 10.1109/AERO53065.2022.9843537.

[10] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors", *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver*, BC, Canada, pp. 7464-7475, 2023, doi: 10.1109/cvpr52729.2023.00721.

[11] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-Time Flying Object Detection with YOLOv8", *arXiv preprint*, Vol. 2305.09972, pp. 1-15, 2023.

[12] C.-Y. Wang, I.-H. Yeh, and H.-Y. M. Liao, "YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information", In: *Proc. of Leonardis*, A., Ricci, E., Roth, S., Russakovsky, O., Sattler, T., Varol, G. (eds) Computer Vision – ECCV 2024. ECCV 2024. Lecture Notes in Computer Science, Vol. 15089. Springer, Cham. https://doi.org/10.1007/978-3-031-72751-1_1.

[13] A. Wang et al., "YOLOv10: Real-Time End-to-End Object Detection", *arXiv preprint arXiv:2405.14458*, 2024.

[14] X. Zhai, Z. Huang, T. Li, H. Liu, and S. Wang, "YOLO-Drone: An Optimized YOLOv8 Network for Tiny UAV Object Detection", *Electron.*, Vol. 12, No. 17, 2023, doi: 10.3390/electronics12173664.

[15] I. A. Alnajjar, L. Almazaydeh, A. A. Odeh, A. A. Salameh, K. Alqarni, and A. A. B. Atta, "Anomaly Detection Based on Hierarchical Federated Learning with Edge-Enabled Object Detection for Surveillance Systems in Industry 4.0 Scenario", *Int. J. Intell. Eng. Syst.*, Vol. 17, No. 4, pp. 649-665, 2024, doi: 10.22266/IJIES2024.0831.49.

[16] U. Sirisha, S. P. Praveen, P. N. Srinivasu, P. Barsocchi, and A. K. Bhoi, "Statistical Analysis of Design Aspects of Various YOLO-Based Deep Learning Models for Object Detection", *Int. J. Comput. Intell. Syst.*, Vol. 16, No. 1, pp. 1-29, 2023, doi: 10.1007/s44196-023-00302-w.

[17] T. H. Wu, T. W. Wang, and Y. Q. Liu, "Real-Time Vehicle and Distance Detection Based on Improved Yolo v5 Network", In: *Proc. of 2021 3rd World Symposium on Artificial Intelligence*, WSAI 2021, Institute of Electrical and Electronics Engineers Inc., 2021, pp. 24-28. doi: 10.1109/WSAI51899.2021.9486316.

[18] H. Liang, Z. Ma, and Q. Zhang, "Self-Supervised Object Distance Estimation Using a Monocular Camera", *Sensors*, Vol. 22, No. 8, 2022, doi: 10.3390/s22082936.

[19] S. H. Qi, J. Li, Z. P. Sun, J. T. Zhang, and Y. Sun, "Distance Estimation of Monocular Based on Vehicle Pose Information", *J. Phys. Conf. Ser.*, Vol. 1168, No. 3, 2019, doi: 10.1088/1742-6596/1168/3/032040.

[20] Y. Yuan, Y. Wu, L. Zhao, H. Chen, and Y. Zhang, "Multiple object detection and tracking from drone videos based on GM-YOLO and multi-tracker", *Image Vis. Comput.*, Vol. 143, p. 104951, 2024, doi: 10.1016/j.imavis.2024.104951.

[21] S. Li, C. Lyu, B. Xia, Z. Chen, and L. Zhang, "TAMDepth: self-supervised monocular depth estimation with transformer and adapter modulation", *Vis. Comput.*, 2024, doi: 10.1007/s00371-024-03332-3.

[22] K. S. Chou et al., "A Lightweight Robust Distance Estimation Method for Navigation Aiding in Unsupervised Environment Using Monocular Camera", *Appl. Sci.*, Vol. 13, No. 19, 2023, doi: 10.3390/app131911038.

[23] S. Lee, K. Han, S. Park, and X. Yang, "Vehicle Distance Estimation from a Monocular Camera for Advanced Driver Assistance Systems", *Symmetry (Basel).*, Vol. 14, No. 12, 2022, doi: 10.3390/sym14122657.

[24] A. Ali, A. Hassan, A. R. Ali, H. Ullah Khan, W. Kazmi, and A. Zaheer, "Real-time vehicle distance estimation using single view geometry", In: *Proc. of* 2020 IEEE Winter Conf. Appl. Comput. *Vision*, WACV 2020, pp. 1100-1109, 2020, doi: 10.1109/WACV45572.2020.9093634.

[25] F. Yu et al., "BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning", In: *Proc. of IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 2633-2642, 2018, doi: 10.1109/CVPR42600.2020.00271.

[26] S. Xu, Z. Guo, Y. Liu, J. Fan, and X. Liu, "An Improved Lightweight YOLOv5 Model Based on Attention Mechanism for Face Mask Detection", *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, Vol. 13531 LNCS, pp. 531-543, 2022, doi: 10.1007/978-3-031-15934-3_44.

[27] Z. Wang, Z. Hua, Y. Wen, S. Zhang, X. Xu, and H. Song, "E-YOLO: Recognition of estrus cow based on improved YOLOv8n model", *Expert Syst. Appl.*, Vol. 238, p. 122212, 2024, doi: 10.1016/j.eswa.2023.122212.

[28] H. Lou et al., "DC-YOLOv8: Small-Size Object

Detection Algorithm Based on Camera Sensor", *Electron.*, Vol. 12, No. 10, pp. 1-14, 2023, doi: 10.3390/electronics12102323.

[29] I. Athanasiadis, P. Mousouliotis, and L. Petrou, "A Framework of Transfer Learning in Object Detection for Embedded Systems", *arXiv preprint*,2018.

[30] Z. Chen, G. Qiu, P. Li, L. Zhu, X. Yang, and B. Sheng, "MNGNAS: Distilling Adaptive Combination of Multiple Searched Networks for One-Shot Neural Architecture Search", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 45, pp. 1-20, 2023, doi: 10.1109/TPAMI.2023.3293885.

[31] V. Krishnan, G. Vaiyapuri, and A. Govindasamy, "Hybridization of Deep Convolutional Neural Network for Underwater Object Detection and Tracking Model", *Microprocess. Microsyst., Vol. 94, No. August*, p. 104628, 2022, doi: 10.1016/j.micpro.2022.104628.

[32] A. A. Shetty, N. T. Hegde, A. C. Vaz, and C. R. Srinivasan, "Multi Cost Function Fuzzy Stereo Matching Algorithm for Object Detection and Robot Motion Control", *J. Robot. Control*, Vol. 4, No. 3, pp. 356-370, 2023, doi: 10.18196/jrc.v4i3.17041.

[33] M. Yang et al., "Hybrid-SORT: Weak Cues Matter for Online Multi-Object Tracking", 2023.

[34] X. Lin, S. Sun, W. Huang, B. Sheng, P. Li, and D. D. Feng, "EAPT: Efficient Attention Pyramid Transformer for Image Processing", *IEEE Trans. Multimed.*, Vol. 25, No. 1941-0077, pp. 50-61, 2023, doi: 10.1109/TMM.2021.3120873.

[35] A. Thakuria et al., "Improving the network architecture of YOLOv7 to achieve real-time grading of canola based on kernel health", *Smart Agric. Technol.*, Vol. 5, p. 100300, 2023, doi: 10.1016/j.atech.2023.100300.

[36] T. Li, Z. Li, Y. Mu, and J. Su, "Pedestrian multi-object tracking based on YOLOv7 and BoT-SORT", In: *Proc. of SPIE 12754, Third International Conference on Computer Vision and Pattern Analysis (ICCPA 2023)*, 127541I (1 August 2023); https://doi.org/10.1117/12.2684256.

[37] N. Aharon, R. Orfaig, and B.-Z. Bobrovsky, "BoT-SORT: Robust Associations Multi-Pedestrian Tracking", *arXiv preprint arXiv:2206.14651*, 2022.

[38] Y. Zhao, H. Zhang, P. Lu, P. Li, E. Wu, and B. Sheng, "DSD-MatchingNet: Deformable sparse-to-dense feature matching for learning accurate correspondences", *Virtual Real. Intell. Hardw.*, Vol. 4, No. 5, pp. 432-443, 2022, doi: 10.1016/j.vrih.2022.08.007.

[39] I. A. Taqi and M. G. Abdul-Haleem, "An Efficient Cryptosystem for Image Using 1D and 2D Logistic Chaotic Maps", *Int. J. Intell. Eng. Syst.*, Vol. 16, No. 4, pp. 125-136, 2023, doi: 10.22266/ijies2023.0831.11.

[40] C. Li et al., "RADepthNet: Reflectance-aware monocular depth estimation", *Virtual Real. Intell. Hardw.*, Vol. 4, No. 5, pp. 418-431, 2022, doi: 10.1016/j.vrih.2022.08.005.

[41] Y. B. Li and J. J. Li, "Harris corner detection algorithm based on improved contourlet transform", *Procedia Eng.*, Vol. 15, pp. 2239-2243, 2011, doi: 10.1016/j.proeng.2011.08.419.

[42] W. Burger, "Zhang's Camera Calibration Algorithm : In-Depth Tutorial and Implementation", *Technical Report HGB16-05*, University of Applied Sciences Upper Austria, School of Informatics, Communications and Media, Dept. of Digital Media, Hagenberg, Austria, 2016, doi: 10.13140/RG.2.1.1166.1688/1.

[43] T. Y. Lin et al., "Microsoft COCO: Common objects in context", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, pp. 740-755, 2014, doi: 10.1007/978-3-319-10602-1_48.

[44] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A Novel Dataset and Benchmarks for Urban Scene Understanding in 2D and 3D", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 45, No. 3, pp. 3292-3310, 2023, doi: 10.1109/TPAMI.2022.3179507.

[45] E. M., V.-G. L., W. C. K. I., W. J., and Z. A., "The Pascal Visual Object Classes (VOC) Challenge", *Int. J. Comput. Vis.*, Vol. 88, No. 2, pp. 303-338, 2010.

[46] H. Zhao et al., "Mixed YOLOv3-LITE: A Lightweight Real-Time Object Detection Method", *Sensors*, Vol. 20, No. 7, p. 1861, 2020, doi: 10.3390/s20071861.

[47] I. Azurmendi, E. Zulueta, J. M. Lopez-Guede, and M. González, "Simultaneous Object Detection and Distance Estimation for Indoor Autonomous Vehicles", *Electronics*, Vol. 12, No. 23, p. 4719, 2023, doi: 10.3390/electronics12234719.

[48] M. Vajgl, P. Hurtik, and T. Nejezchleba, "Dist-YOLO: Fast Object Detection with Distance Estimation", *Appl. Sci.*, Vol. 12, No. 3, pp. 1-13, 2022, doi: 10.3390/app12031354.

[49] N. Youssouf, "Traffic sign classification using CNN and detection using faster-RCNN and

YOLOV4", *Heliyon*, Vol. 8, No. 12, 2022, doi: 10.1016/j.heliyon.2022.e11792.

[50] E. Guney, C. Bayilmis, and B. Cakan, "An Implementation of Real-Time Traffic Signs and Road Objects Detection Based on Mobile GPU Platforms", *IEEE Access*, Vol. 10, pp. 86191-86203, 2022, doi: 10.1109/ACCESS.2022.3198954.