

2-7-2024


Automated identification of vehicles in very high-resolution UAV orthomosaics using YOLOv7 deep learning model

Esra YILDIRIM
esrayildirim@gtu.edu.tr

Umut Güneş SEFERCİK
sefercik@gtu.edu.tr

Taşkın KAVZOĞLU
kavzoglu@gtu.edu.tr

Follow this and additional works at: <https://journals.tubitak.gov.tr/elektrik>

 Part of the [Computer Engineering Commons](#), [Computer Sciences Commons](#), and the [Electrical and Computer Engineering Commons](#)

Recommended Citation

YILDIRIM, Esra; SEFERCİK, Umut Güneş; and KAVZOĞLU, Taşkın (2024) "Automated identification of vehicles in very high-resolution UAV orthomosaics using YOLOv7 deep learning model," *Turkish Journal of Electrical Engineering and Computer Sciences*: Vol. 32: No. 1, Article 9. <https://doi.org/10.55730/1300-0632.4060>

Available at: <https://journals.tubitak.gov.tr/elektrik/vol32/iss1/9>

This Article is brought to you for free and open access by TÜBİTAK Academic Journals. It has been accepted for inclusion in Turkish Journal of Electrical Engineering and Computer Sciences by an authorized editor of TÜBİTAK Academic Journals. For more information, please contact academic.publications@tubitak.gov.tr.

Automated identification of vehicles in very high-resolution UAV orthomosaics using YOLOv7 deep learning model

Esra YILDIRIM*, Umut Güneş SEFERCİK, Taşkın KAVZOĞLU

Department of Geomatics Engineering, Faculty of Engineering, Gebze Technical University, Kocaeli, Türkiye

Received: 20.09.2023

Accepted/Published Online: 23.01.2024

Final Version: 07.02.2024

Abstract: The utilization of remote sensing products for vehicle detection through deep learning has gained immense popularity, especially due to the advancement of unmanned aerial vehicles (UAVs). UAVs offer millimeter-level spatial resolution at low flight altitudes, which surpasses traditional airborne platforms. Detecting vehicles from very high-resolution UAV data is crucial in numerous applications, including parking lot and highway management, traffic monitoring, search and rescue missions, and military operations. Obtaining UAV data at desired periods allows the detection and tracking of target objects even several times during a day. Despite challenges such as diverse vehicle characteristics, traffic congestion, and hardware limitations, the detection task must be executed swiftly and accurately. This study successfully achieved automated detection and instance segmentation of parked and moving vehicles across a large university campus by employing the robust learning capabilities of the You Only Look Once version 7 (YOLOv7) deep learning algorithm. The generation of an ultrahigh-resolution orthomosaic of the university campus was accomplished through photogrammetric processing, employing 20-megapixel aerial images obtained from RGB UAV flights with polygonal nadir-view and bundle-grid oblique-view imaging geometries. The vehicle dataset was created by cropping image patches containing vehicle objects from the orthomosaic and manually labeling the boundaries of the vehicle targets using the LabelMe annotation tool. After expanding the dataset by applying data augmentation, the YOLOv7 algorithm was trained and tested using the transfer learning approach. The accuracy metric of precision, recall, and mAP@0.50 scores for the bounding boxes and masks of vehicles were estimated as 99.79, 97.54, and 99.46%, respectively. In addition, the robustness of the trained algorithm was also tested on a short video and (>80%) prediction scores were achieved.

Key words: Vehicle detection, instance segmentation, UAV, orthomosaic, deep learning, YOLOv7

1. Introduction

Currently, there is an increasing curiosity in employing unmanned aerial vehicles (UAVs) for automated target recognition and tracking tasks in various applications including traffic monitoring [1], disaster management [2], and smart agriculture [3]. UAVs are cost-effective, environment-friendly, and provide high-resolution images for a large field of views periodically or in real-time that facilitate target detection. In this context, there are diverse types of UAVs in terms of their weights as UAV1-4 and wing types as fixed-wing, multicopter and VTOL and they can offer satisfying solutions in many areas such as military and commercial operations [4–6]. Moreover, UAVs provide rich image datasets for various image processing applications, thanks to very high-resolution imaging capability from different flying altitudes, viewing angles, and locations [7]. These advantages of UAVs have led to the prevalent adoption of UAV imagery for vehicle detection and tracking studies [8–10].

*Correspondence: esrayildirim@gtu.edu.tr

Considering the increasing traffic density with the rapidly increasing population, automated detection of vehicles from UAV imagery is crucial in numerous applications such as parking lot and highway management, traffic monitoring, and vehicle supervision. Traditional vehicle detection methods mostly rely on manual extraction of vehicle characteristics, such as histogram of oriented gradients (HOG) [11], Haar-like [12], bag-of-words (BoW) [13] and classifying them using classifiers with varying complexity, including support vector machine (SVM) [14], adaptive boosting (AdaBoost) [15], and k-nearest neighbor (kNN) [16]. However, achieving high detection accuracy and real-time recognition performance is challenging for conventional approaches [17, 18].

With the excellent performance of deep learning paradigm, particularly convolutional neural networks (CNNs), in image processing applications, including object detection [19–21], classification [22–24], image retrieval [25, 26], and scene/object segmentation [27, 28], various CNN-based object recognition and tracking algorithms have been introduced and efficiently applied in vehicle target detection applications. CNN-based deep learning algorithms outperform traditional object recognition methods that perform feature extraction based on manual selection and expert experience, by automatically and adaptively extracting features from input data through a self-learning methodology [29]. When it comes to CNN-based target detection methods, they can be categorized into two groups: one-stage detectors and two-stage detectors. In the initial phase, two-stage object detectors produce a set of region proposals, referred to as regions of interest (ROI), by utilizing a region proposal network (RPN) on the input scene. These proposals are then classified, and the bounding box regression is employed to identify the location of the target. Although these detectors achieve high detection and localization accuracy, they suffer from high computational costs and an inability to achieve real-time performance. These detectors mainly include region-based convolutional neural network (R-CNN) [30], Fast R-CNN [31], Faster R-CNN [32], spatial pyramid pooling network (SPP-Net) [33], region-based fully convolutional network (R-FCN) [34], and mask R-CNN [35]. Conversely, one-stage detectors remove the region proposal generation phase and adopt a unified system that class probabilities, and bounding box positions are regressed directly from the input images. The significant advantages of these algorithms are that they are faster and convenient for utilization in real-time applications. Additionally, the widely used representative one-stage detectors can be listed as You Only Look Once (YOLO) series [36–42], single shot multiBox detector (SSD) [43], and RetinaNet [44]. In recent years, both approaches have been employed in vehicle recognition studies. However, it should be mentioned that one-stage models are mostly preferred in real-time vehicle detection and tracking applications [45, 46]. Specifically, among one-stage models, YOLO was selected for the vehicle detection problem in this study due to its superior performance and speed that provides real-time detection, which has been confirmed by previous studies in the literature [47–50].

In the study carried out by Chen et al. [47], a framework based on the YOLOv5 object detector was proposed for vehicle identification from high-resolution UAV imagery. An adaptive clipping algorithm was applied to high-resolution UAV imagery in the preprocessing of the dataset and vehicle detection phases to improve small vehicle detection performance. They utilized the publicly available VisDrone dataset consisting of a large number of drone video frames. With their proposed system, they achieved a 41.7% increase in the vehicle detection performance of the YOLOv5 model. Ammar et al. [48] made a comparison between the performance of the popular Faster R-CNN, YOLOv4, and YOLOv5 frameworks for vehicle recognition, and conducted numerous experiments to examine the impact of different hyperparameter combinations (input size, feature extractor, number of iterations, etc.) on models. They utilized a publicly available Stanford dataset and their own PSU dataset, both of which consist of drone imagery. The results revealed that the two models

outperformed the Faster R-CNN in the recognition of vehicles while there was no significant difference between YOLOv3 and YOLOv4 on both datasets. Amato et al. [49] employed the YOLOv3 model to detect and count vehicles in UAV imagery. They utilized the CARPK dataset containing drone images and the PUCPR+ dataset, which is a large-scale vehicle-counting dataset. The weights of the pretrained YOLOv3 algorithm in the COCO dataset were used, also the model was fine-tuned. They achieved 97% and 95% precision for the CARPK and PUCPR+ datasets, respectively. Benjdira et al. [50] conducted a comparative performance investigation between Faster R-CNN and YOLOv3 for vehicle recognition from UAV imagery. They created a vehicle dataset with the UAV images of the Prince Sultan University campus. Experimental results revealed that the YOLOv3 model exhibited superior in terms of sensitivity and recognition speed.

This research investigates the effectiveness of the cutting-edge YOLOv7 model for automatically detecting and pixel-wise segmenting vehicles using a very high resolution (~ 2.5 cm) UAV orthomosaic and a 30-s video. To construct a vehicle dataset, the UAV aerial photos were acquired in RGB imaging band in Gebze Technical University (GTU) campus, the high quality orthomosaic was generated, and image patches containing the vehicle targets were cropped from the orthomosaic. The vehicle dataset was constructed by manually labeling the vehicles in the cropped images with the polygon shape, followed by a data augmentation process to expand the sample size. Afterwards, the YOLOv7 model was trained and tested using the created corresponding datasets, and its performance was analyzed using commonly used accuracy metrics. The work contributed to the development of an integrated system that works in the form of high-quality UAV orthomosaic generation in any target area and automatic vehicle detection through deep learning. System's real-time applicability in independent video data to design a real-time approach that can be used onboard UAV platforms was additionally analyzed.

2. Study area and materials

The study was carried out in the GTU campus area of approximately 2 km², which is very suitable for automated vehicle detection with its large parking lots and roads. The campus is located in Kocaeli, one of the largest metropolitans in the northwest of Türkiye. In the area, several university buildings, forest, agricultural areas, and water bodies exist. The topography is generally flat and the orthometric elevation of the bare ground is between 1 and 50 m. Figure 1 shows the location of the GTU Campus in Türkiye and the original color 3D textured UAV mesh model.

The UAV idea is not new; in fact, it was put forward in 1783 with the use of hot-air balloons. Then, usage for military purposes began in 1849. The civilian use of UAVs started in 2006 and spread with the first camera-equipped UAV by DJI company in 2013 [51–53]. In this study, the orthomosaic, used for automated vehicle detection, was produced by utilizing the aerial photos captured by DJI Phantom IV Pro Version 2 (V2). DJI Phantom IV Pro V2 is the highest resolution UAV of the DJI Phantom series with a 20 MP Sony Exmor red-green-blue (RGB) camera. On the other hand, the device just has a low accuracy consumer-grade global navigation satellite systems (GNSS) receiver only for navigation purposes and does not include a real time kinematic (RTK) GNSS receiver. Therefore, the usage of ground control points (GCP) is mandatory for the orientation of aerial photos. Accordingly, 86 mobile polycarbonate GCPs were precisely located before the planned UAV flights, and coordinates were measured with CHC-i80 GNSS receiver. Specifications of the equipment used for the acquisition and orientation of aerial photos are presented in Table 1.

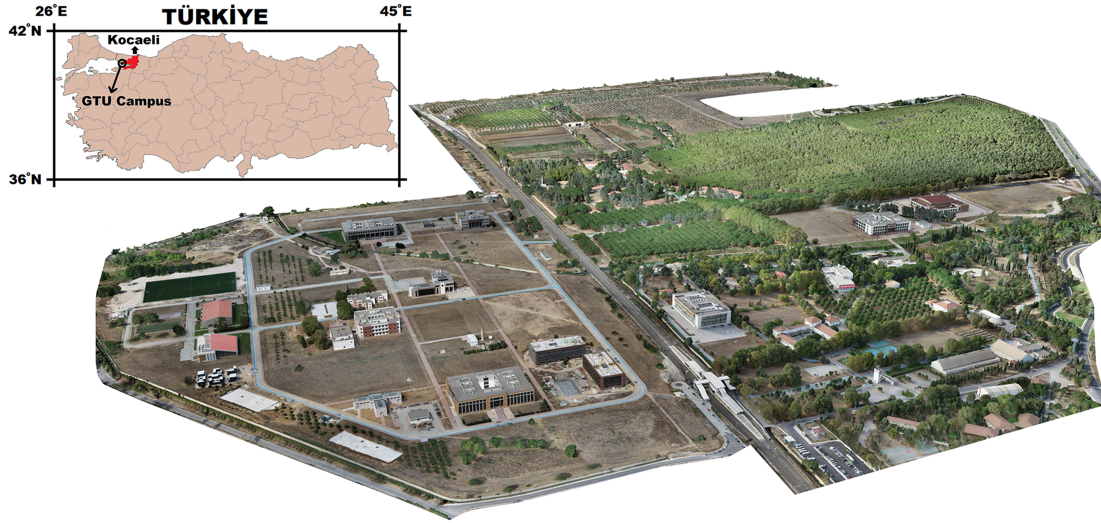




Figure 1. Location of the GTU Campus in Türkiye and original color 3D textured UAV mesh model generated in this study.

Table 1. Specifications of the equipment used for the acquisition and orientation of aerial photos.

			
DJI Phantom IV Pro V2 + Sony Exmor Camera		CHC-i80 GNSS receiver + Mobile GCP	
Specification	Value	Specification	Value
Camera	4K, HD, 1080p, 1" effective pixels 20 MP	GNSS technology	GPS, GLONASS, GALILEO, BeiDou, SBAS, NavIC
Gimbal	3-axis (pitch, roll, yaw)	Operating system	Linux
Flight duration	Max. 30 min	Working modes	Static, VRS RTK, UHF RTK, all surveying modes
Weight	1.375 g	Positioning accuracy RTK	± 0.8 cm H, ± 1.5 cm V with initialization reliability >99.9%
Speed	Max 20 m/s in S-mode	Modem - Bluetooth	4G, 3G, GSM – V4
Wind speed resistance	Max. 10 m/s	Battery	Dual; Static up to 10 h, Cellular receive only up to 9h, UHF receive/transmit up to 6h
Operating temperature	0 °C to 40 °C	Network - RTK	Available
Outdoor positioning module	GPS/GLONASS dual	Internal memory	32 GB
Hover accuracy range	± 0.1 m V, ± 0.5 m H (Vision) ± 0.3 m V, ± 1.5 m H (GPS)	GCP	0.25 m \times 1 m, polycarbonate

3. Methodology

3.1. UAV data acquisition, image orientation and orthomosaic production

The 20 MP UAV aerial photos were captured by different flying geometries as polygonal, bundle-grid (north-south + east-west directions), and circular. The flights were planned and conducted utilizing Pix4D capture software and 8333 aerial photos with ≤ 2.2 cm ground sampling distance (GSD) were acquired by 32 flights. In polygonal and bundle-grid flights, 80 m flying altitude and 80% and 60% front and side overlap ratios were applied, respectively. For circular flights, 30 m flying altitude was applied and 5° capturing angle was determined which means 72 aerial photos ($360^\circ / 5^\circ$) for each flight. While polygonal flights were preferred for agricultural areas and forest, bundle-grid and circular geometries were used in the built-up areas of the campus. The main methodologic stages of VHR UAV orthomosaic production are shown in Figure 2.

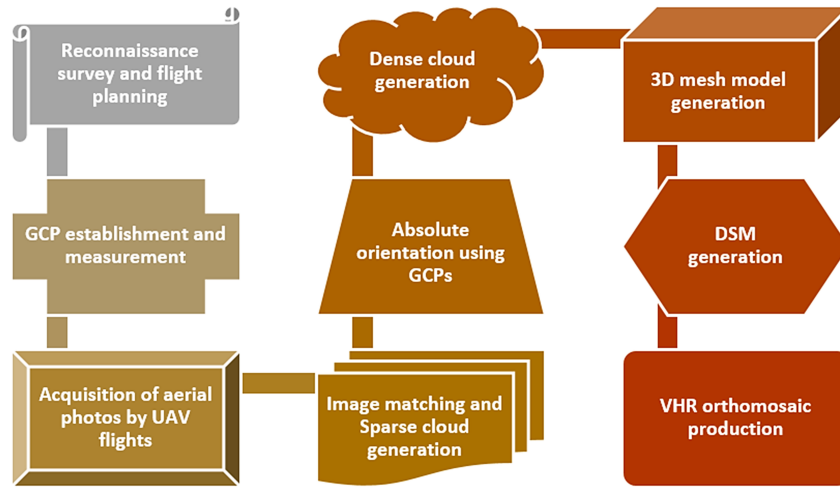


Figure 2. Main methodologic stages of VHR UAV orthomosaic production.

The main advantage of the bundle-grid and circular flights are oblique viewing angles (70°) rather than nadir, utilized in polygonal flights and multiview to the target objects from different sides that improves the quality of image matching and 3D point cloud generation when using structure from motion (SfM) algorithm. The SfM is a low-cost and robust photogrammetric technique that enables the reconstruction of 3D geometry from a series of overlapping mono photos [54]. In this study, the photogrammetric processing of UAV aerial photos was performed utilizing Agisoft Metashape SfM-based software. The UAV aerial photos were oriented by two stage operation as initial (mutual) alignment and absolute orientation utilizing GCPs. Just before the initial orientation, background filtering (masking) process was applied to eliminate the influence of low coherence noisy parts in oblique aerial photos derived from bundle-grid and circular flights. Thus, the accuracy of the initial alignment was increased. In initial alignment, the aerial photos were oriented applying high quality level and area-based crosscorrelation. For the better arrangement of overlapped aerial photos, generic and reference preselection parameters were utilized, and sparse cloud was obtained including ~ 73 million points. In absolute orientation, 86 well-defined GCPs were marked in the related aerial photos, and root mean square error (RMSE) of ± 2 cm (± 0.9 GSD) was achieved considering the following equation where $\hat{X}_i, \hat{Y}_i, \hat{Z}_i$ are the estimated coordinates, X_i, Y_i, Z_i are the actual GCP coordinates acquired by terrestrial GNSS measurements, and n is the number of observations.

$$RMSE = \sqrt{\frac{\sum_{i=1}^n (\hat{X}_i - X_i)^2 + (\hat{Y}_i - Y_i)^2 + (\hat{Z}_i - Z_i)^2}{n}} \quad (1)$$

The 3D dense point cloud of the study area was generated utilizing depth maps, built by extracting depth information from aerial photos with desired overlap. It was comprised of approximately 1.5 billion points and filtered by fencing and classifying noisy points. Dense point clouds visualize the objects in noncontinuous point-based vector format; thus, a more realistic and detailed solid 3D mesh model of the study area was generated utilizing triangulation-based interpolation. In this process, due to the number of predicted faces (triangles) and vast size of polygonal data, mesh decimation was conducted, and the number of generated mesh faces was reduced to approximately 430 million faces [55].

To achieve a true orthomosaic in orthogonal geometry, a digital surface model (DSM) is required. DSM is a digital cartographic representation of earth's surface with X, Y planimetric coordinates and altitude Z. It determines all natural and human-made objects as a 2.5D solid model in different spatial resolutions depending on the data acquisition technique. Using UAV technology, the highest resolution DSMs in comparison with other airborne or space-borne remote sensing techniques can be generated by means of very low flying altitudes. In principle, the optimal original grid spacing of DSMs derived from remotely sensed data is $3 \times \text{GSD}$; otherwise, interpolation causes substantial loss of vertical accuracy due to an insufficient amount of data [56–58]. As a result, the DSM was generated with approximately 6 cm considering approximately 2 cm GSD of UAV aerial photos. Finally, approximately 2.5 cm resolution gap-free true orthomosaic (Figure 3), a powerful base for automated vehicle detection, was produced using high resolution DSM and hole filling process.



Figure 3. Produced approximately 2.5 cm orthomosaic of GTU campus area.

3.2. Vehicle dataset preparation

To establish a dataset of vehicles, we extracted image samples featuring vehicle objects from the generated RGB orthomosaic data. Handling and processing very high-resolution UAV data is both labor-intensive and time-consuming because of their intricate details. On the other hand, convolutional neural networks require a large amount of computation together with large training samples compared to machine learning methods. To address constraints related to GPU memory, equal-sized patches were cropped from UAV orthomosaic data. A total of 200 image patches, each with dimensions of 512×512 pixels, were collected from the study area. Figure 4 provides visual representations of typical vehicle images from this dataset. It is apparent that this dataset encompasses a variety of vehicle objects with different characteristics, including types, colors, pixel sizes, orientations, lighting conditions, and scenes. It should be noted that all aerial photos including vehicles ranging from one to ten were in RGB format with an 8-bit radiometric and approximately 2.5 cm spatial resolution.



Figure 4. Representative images from the UAV vehicle dataset.

3.3. Image annotation

In object detection studies, image labeling has a pivotal role before training deep learning models to build a robust and accurate model. The CNN model learns from the labeled features during the training phase and produces outcomes based on the quality of these features. Hence, the labeling preciseness of the features profoundly influences the accuracy of the model. In this context, ground-truth labels should be correctly annotated and avoid ambiguity. In this study, LabelMe, an open-source tool designed for image labeling, was employed to label the vehicle targets that exist in the images [59]. For the vehicle instance segmentation task, each vehicle in the images was manually annotated using a polygon shape. As a result of the labeling, ground-truth masks for each vehicle in the images were obtained. The image annotation process in the LabelMe tool and the instance segmentation mask generated for that image are demonstrated in Figure 5. Since the instance segmentation identifies each object in the given image as a distinct sample, regardless of its corresponding class information, each vehicle target was masked in different colors in the generated ground-truth mask.

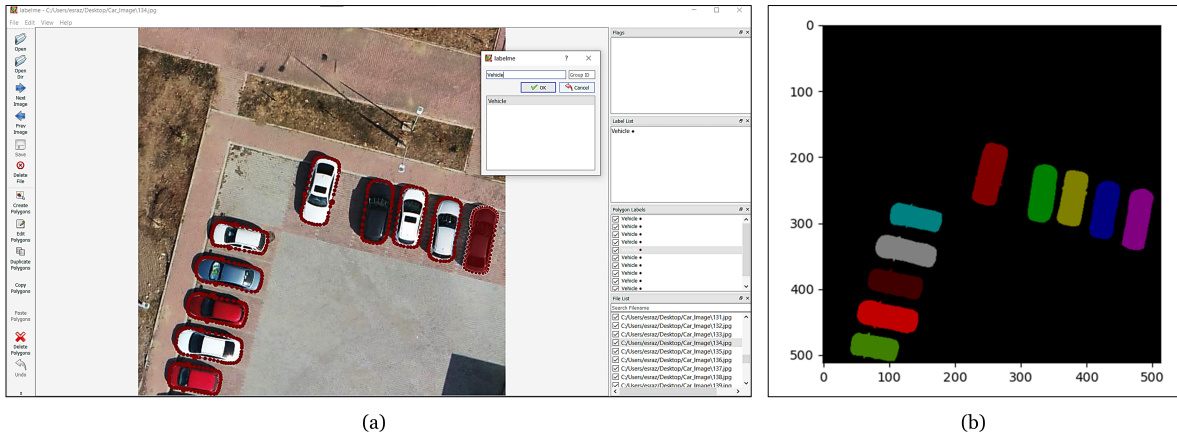


Figure 5. (a) Labeling of an image in the LabelMe tool, (b) generated ground-truth mask.

After the image annotation process, a total of 494 vehicle objects were labeled in 200 aerial photos. Then, the created dataset was divided into three datasets: 70% of the dataset was utilized for training, 20% for validation, and the remaining 10% for testing the model's performance. Consequently, the training, validation, and testing datasets contained 306, 122, and 60 vehicle instances, respectively.

3.4. Data augmentation

Deep learning models require larger training samples than the other algorithms contingent upon the complexity of the deep neural network. The effectiveness of a target recognition model based on deep learning is heavily influenced by the quality of the utilized training data. More specifically, the amount and diversity of the dataset employed improve the capability of the algorithm for generalization and its training performance, thus helping to avoid overfitting problem. Therefore, data augmentation techniques are generally utilized as a critical preprocessing stage to boost the robustness of the models. Considering these requirements, various data augmentation methods were applied to the training dataset in this study. Different variations of the existing images were thus produced synthetically. The applied data augmentation strategies included horizontal flipping, adding salt and pepper noise to 5% of the pixels in the image, and randomly cropping between 0% and 50% of the image. The comparison of the original, horizontally rotated, noise added, and cropped images are shown in Figure 6. After data augmentation, a vehicle dataset containing a total of 480 images and 1055 vehicle instances was obtained.

3.5. Dataset characteristics

To further analyze the characteristics of the dataset, the label distribution of the vehicle instances was visually analyzed. As illustrated in Figure 7a, the dataset comprises only one vehicle class with more than 800 instances in total. The red bounding boxes show the labels of the vehicle targets. Considering normalized image sizes, the x and y values of the center points of the bounding boxes for instances in the dataset vary from 0.0 and 1.0. On the other hand, the width of the targets in the dataset varies between 0.0 and 0.5, while the height varies between 0.0 and 0.4. The transition from light blue to dark blue in the graphs indicates a more concentrated distribution. Accordingly, it was seen that small-sized vehicle targets with a width and height less than 0.25 in

the dataset were concentrated. It was also observed that the targets were relatively concentrated in the center of the image. Furthermore, Figure 7b displays the label correlogram of the vehicle dataset. As indicated by the correlation statistics, the width and height of the bounding boxes delimited the vehicle samples in the dataset were mostly concentrated at 0.1 and 0.2 values. While the y values of the center point of the bounding boxes are concentrated at 0.5, the x values are about equally distributed in the normalized image sizes.

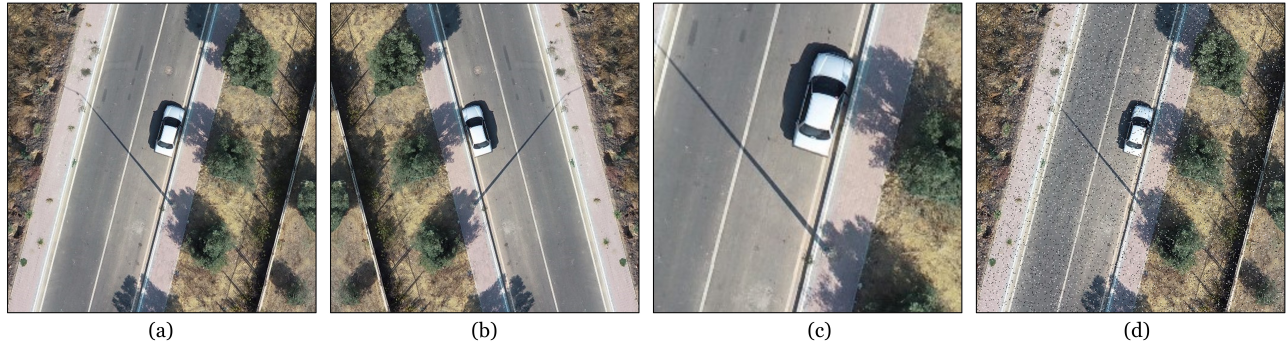


Figure 6. Visual representation of the data augmentation methods utilized in the study: (a) original image, (b) horizontal flipped, (c) random cropped, (d) random noise added.

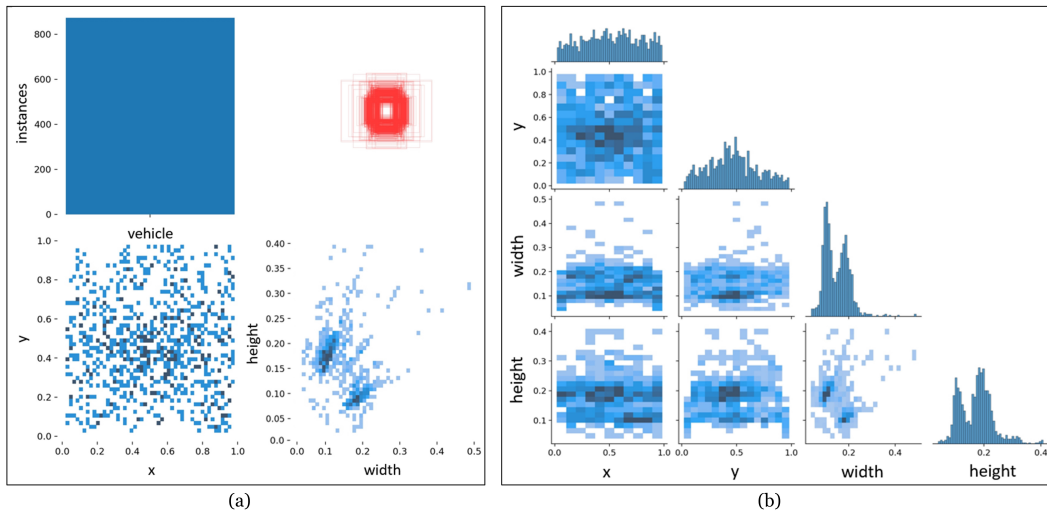


Figure 7. Visualization of vehicle dataset: (a) label distribution, (b) label correlation.

3.6. YOLOv7 algorithm

YOLO [36] is a state-of-the-art deep learning-based target detection algorithm introduced by Redmon et al. It addresses object recognition tasks as a regression problem. It is one of the most widely utilized algorithms for the identification of objects because of its speed and accuracy trade-off. The YOLO specifies an approach where the recognition task is completed in a single forward propagation through a CNN. The core concept of YOLO in target recognition involves dividing input images into a grid of $S \times S$ cells thus enabling faster inference (Figure 8). Each grid cell is supposed to predict the targets whose center falls into that cell, with a B

bounding box. Along with the bounding boxes, confidence scores are also calculated, reflecting how confident the predicted bounding boxes contain an object. Thus, each bounding box predicted by the model contains five different values including b_x , b_y , b_w , b_h , and p_c . Among these, b_x and b_y denote the x and y coordinates of the predicted bounding box's center point relative to the grid cell, while b_w and b_h correspond to the width and height of the bounding box relative to the entire image. Also, p_c signifies the confidence score associated with the bounding box. Additionally, each grid element calculates the C conditional class probabilities. The final model output is a tensor with the shape $S \times S \times (B * 5 + C)$. However, because multiple bounding boxes may represent the same target in the input image, a nonmaximum suppression (NMS) algorithm is utilized to remove these duplicate detections.

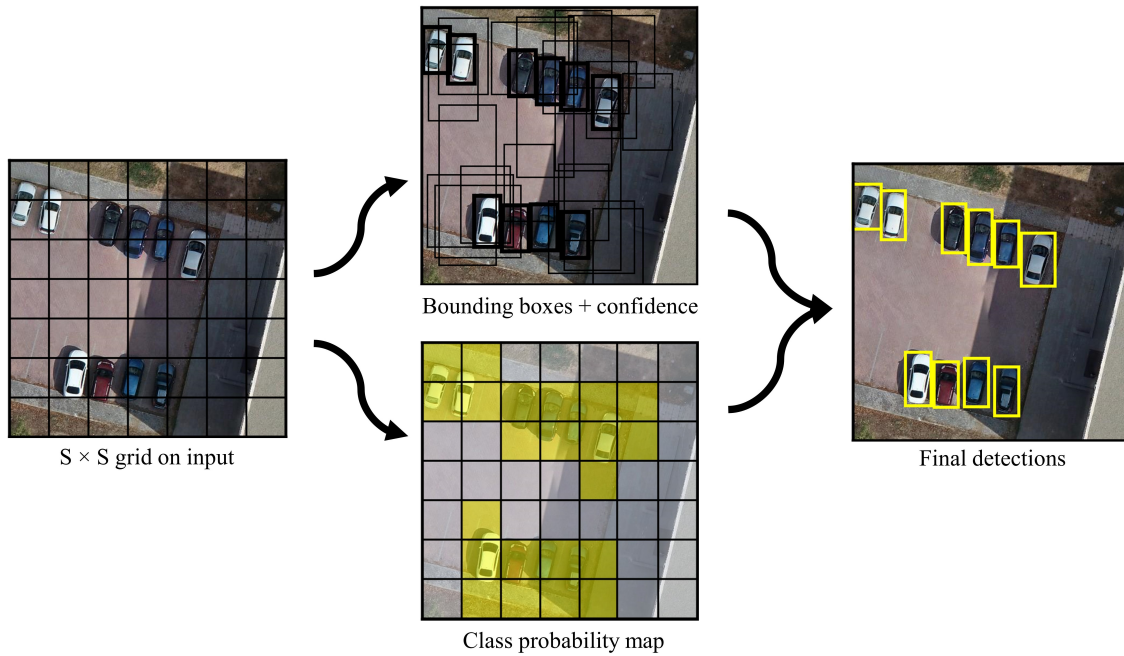


Figure 8. The detection pipeline of YOLO.

NMS is employed as a postprocessing technique for filtering out redundant bounding boxes to obtain final detections in object detection models. For this, NMS discards bounding boxes with lower confidence scores using a predefined overlap threshold value, and just keeps the most accurate bounding box [60]. Algorithm 1 gives the pseudo code of this procedure.

The YOLO network is trained using the following, multipart loss function:

Algorithm 1: Nonmaximum suppression.

Input: $B = \{b_1, \dots, b_N\}$, $C = \{c_1, \dots, c_N\}$, t
 B is the list of initial detection boxes
 C contains corresponding detection scores
 t is the NMS threshold

```

begin
     $F \leftarrow \{\}$ 
    while  $B \neq \text{empty}$  do
         $m \leftarrow \text{argmax } C$ 
         $M \leftarrow b_m$ 
         $F \leftarrow F \cup M$ ;  $B \leftarrow B - M$ 
        for  $b_i$  in  $B$  do
            if  $\text{iou}(M, b_i) \geq t$  then
                 $B \leftarrow B - b_i$ ;  $C \leftarrow C - c_i$ 
            end
        end
    end
    return  $F, C$ 
end
    
```

$$\begin{aligned}
 \text{Loss} &= \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\
 &+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\
 &+ \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \\
 &+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2 \\
 &+ \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2,
 \end{aligned} \tag{2}$$

where $\mathbb{1}_i^{\text{obj}}$ denotes if object appears in cell i and $\mathbb{1}_{ij}^{\text{obj}}$ denotes that the j th bounding box predictor in cell i is responsible for that prediction. Moreover, λ_{coord} increases the loss from bounding box coordinate predictions, and λ_{noobj} decreases the loss from confidence predictions for bounding boxes that do not contain objects.

Several subsequent versions of YOLO have been released in the past years, and each has distinctive architectural reforms and features [36–42]. In July 2022, YOLOv7 was introduced, with few improvements significantly increasing detection accuracy and speed. The architectural changes of the YOLOv7 algorithm include extended efficient layer aggregation network (E-ELAN), and model scaling for concatenation-based models. In addition, the bag-of-freebies utilized in the YOLOv7 are planned reparameterized convolution,

coarse label assignment for the auxiliary head and fine label assignment for the lead head, batch normalization in conv-bn-activation topology, implicit knowledge, and exponential moving average [42]. The architecture of the YOLOv7 network, comprising of input, backbone, neck, and head parts, is demonstrated in Figure 9. In the input part, the image is rescaled to ensure uniform pixel dimensions before being fed into the backbone network. On the other hand, the backbone consists of several E-ELAN, CBS, and MP1 modules. The CBS module extracts image features at different scales and comprises convolution, batch normalization layers, and sigmoid-weighted linear unit activation function (SiLU). Also, the E-ELAN is composed of a few CBS modules and allows the algorithm to converge and learn more efficiently by maintaining the original gradient path. The MP1 module consists of a max pooling layer and a CBS module.

The neck of the model is comprised of the path aggregation feature pyramid network (PA-FPN), including EH-ELAN, SPPCSPC, CAT, MP2, and UP modules. PA-FPN is utilized because of its ability to accurately preserve spatial information which aids in the proper localization of pixels. The SPPCSPC module consists of CBS, CAT, and max pooling modules. The difference between MP2 and MP1 is that the number of channels is doubled in MP2. The CAT module performs concatenation operations that fuse the features from previous branches, and the UP module comprises up-sampling and CBS modules. Moreover, in the head part, the YOLOv7 network applies a REP (RepConv) module to modify the number of image channels for the outputs at three distinct scales from the neck part, and subsequently applies 1×1 convolution for predicting confidence, class, and anchor frame.

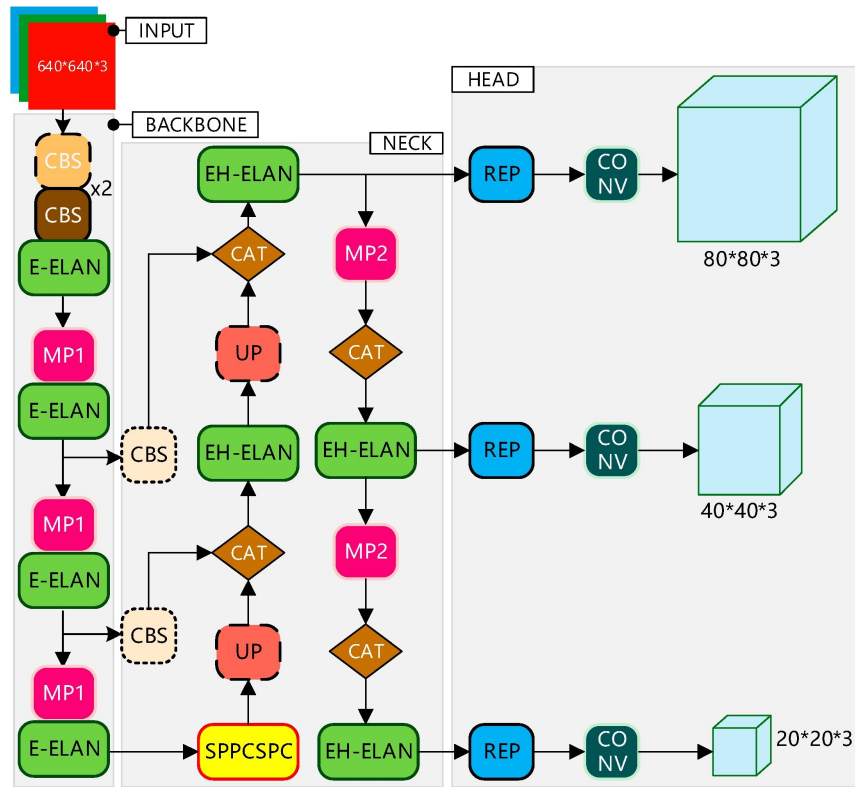


Figure 9. YOLOv7 network structure diagram [61].

Table 2. Hyperparameter configuration for the experiment.

Hyperparameters	Value
Image size (pixel)	640×640
Epochs	100
Batch size	16
Optimizer	SGD
Initial learning rate	0.01
Momentum	0.937
Weight decay	0.0005

3.7. Design and implementation

All experiments in the study were carried out using the Python programming language and the PyTorch deep learning framework. Also, the training and validation process of the YOLOv7 was conducted in the cloud-based Google Colab environment which offers access to the NVIDIA Tesla T4 GPU. Due to the small dataset utilized in the study, a transfer learning approach was employed. Transfer learning enables the reuse of weights learned by models trained on large datasets including COCO and ImageNet in the training of the target model with a smaller dataset. Furthermore, training the network from scratch, where the network weights of the model are randomly initialized, is a very time-consuming process. Instead, with transfer learning, using weights learned by a pretrained network allows the model to converge faster. Therefore, the training of the YOLOv7 on the vehicle dataset was initialized by using the weights that the model had previously learned in the COCO dataset. The training hyperparameters of the model employed in the study are given in Table 2. These include an input image size of 640×640 pixels, an initial learning rate of 0.01, a momentum of 0.937, weight decay set at 0.0005, SGD as the optimizer, a batch size of 16, and 100 training epochs.

4. Experimental results

To avoid underfitting and overfitting of the model, whole training and validation processes were monitored (Figure 10). There exist four types of losses generated during the training and validation of the YOLOv7: bounding box loss, segmentation loss, and objectness loss. Objectness loss measures the probability of objects being present within the predicted bounding box; a higher objectness score implies a greater probability of the bounding box containing an object. Bounding box loss indicates how effectively the predicted bounding box encompasses an object. Furthermore, segmentation loss measures the difference, on a per-pixel basis, between the predicted target mask and the actual ground-truth target mask. During the YOLOv7 training and validation stages, three loss graphs demonstrated a decreasing trend, and no overfitting was monitored in the model. As illustrated in Figure 10a, it is obvious that the training loss curve of the algorithm converged in the initial phases of the training. Furthermore, as depicted in Figure 10b, it can be observed that the validation loss curve of the algorithm converged towards the end of the training process. After the completion 100 training epochs, both the training and validation loss curves achieved their minimum values.

For qualitative performance analysis of the model, precision, recall, and mean average precision (mAP) accuracy assessment metrics were utilized. Precision refers to how many of the objects predicted by the algorithm are the actual vehicle, and recall refers to how many of all vehicles are predicted by the algorithm. The precision and recall are computed based on true positive (TP), true negative (TN), false positive (FP), and false negative (FN) detections. The mAP is computed as the average of the APs across various c classes, with

AP representing the area under the precision-recall curve. In the calculation of the mAP, two intersections over union (IoU) thresholds were employed: 0.50 (mAP@0.50) and a range from 0.5 to 0.95 with increments of 0.05 (mAP@0.50:0.95). The IoU quantifies the degree of overlap between the bounding box predicted by the model and the actual ground truth bounding box. All these metrics are computed as follows:

$$Precision = \frac{TP}{TP + FP} , \quad (3)$$

$$Recall = \frac{TP}{TP + FN} , \quad (4)$$

$$AP = \int_0^1 P(R)dR , \quad (5)$$

$$mAP = \frac{1}{c} \sum_{i=1}^c AP_i . \quad (6)$$

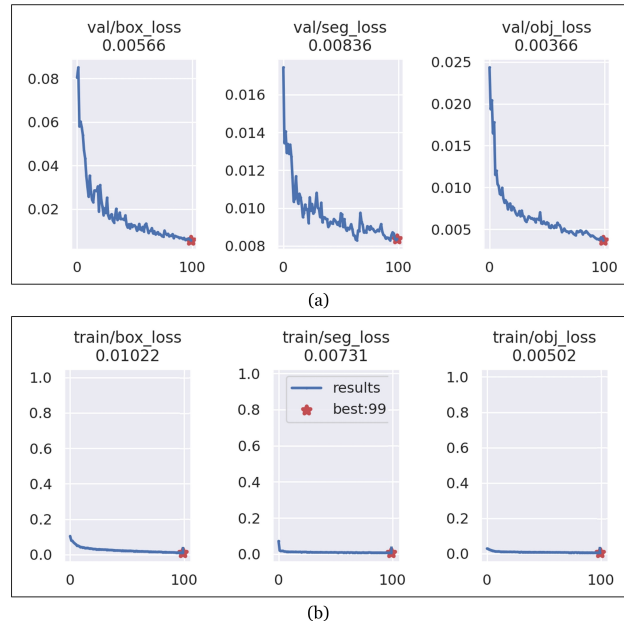


Figure 10. The loss graphs of the YOLOv7 model: (a) training loss graphs, (b) validation loss graphs.

As a result of the model evaluation, accuracy metrics were calculated for both the predicted bounding boxes and the segmentation masks. The accuracy analysis results of the model are visualized in Figure 11. For both bounding boxes and segmentation masks, the precision, recall, and mAP@0.50 metrics were calculated as 99.79, 97.54, and 99.46%, respectively. In addition, the mAP@0.50:0.95 metric reached 97.31% for bounding boxes and 93.09% for segmentation masks. Since predicting the mask of the vehicles on a per-pixel basis was a more complex problem than predicting the bounding box, and the mAP@0.50:0.95 metric required challenging IoU threshold values, the segmentation masks reached a lower value than the bounding boxes. Considering the computational load of the algorithm, the whole training and validation process took about 49 min.

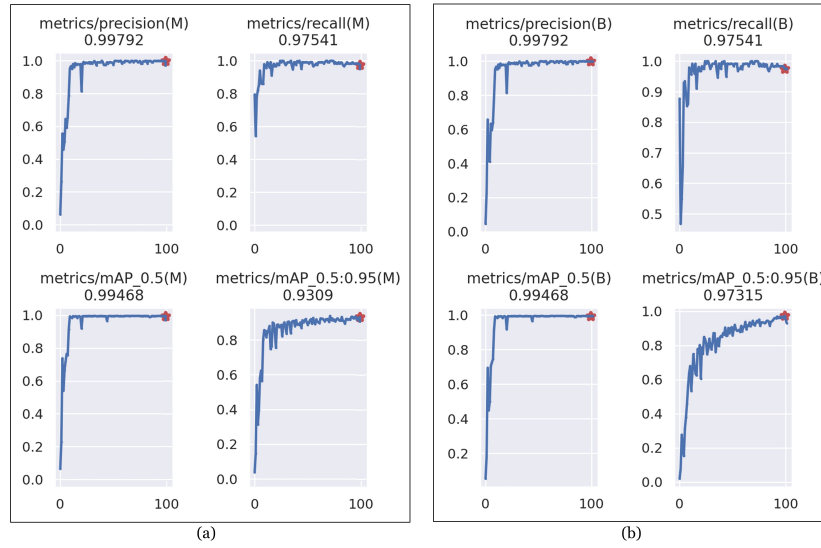


Figure 11. Result of the model evaluation metrics for (a) bounding boxes, (b) segmentation masks.

To analyze the performance of the algorithm in detail the F1-confidence (Equation 7), and precision-recall (PR) curves were analyzed (Figure 12). The F1-confidence curve is a measure of the precision and recall values at any threshold. As seen in Figure 12a, when the confidence value optimizing precision and recall was 0.844, the model attained an F1 score of 99%, indicating high accuracy.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (7)$$

The precision-recall curve illustrated in Figure 12b indicates the changes in the precision values of the algorithm as recall values increase during the training phase. In this case, the model achieved a mAP of 99.5% with a threshold of 0.5 for a single vehicle class. In addition, the curve concentrates in the upper right corner, indicating that the model can predict vehicles with high performance.

After the training and validation stages of the model, 20 UAV test images that were not introduced to the model before were fed to the trained model, and the detection and segmentation results were visually analyzed (Figure 13). As shown in the testing results, YOLOv7 successfully detected the vehicles in the images and produced the instance segmentation masks. The bounding boxes and pixel-wise instance segmentation masks that the algorithm predicts for vehicle objects appropriately represent the boundaries of the objects. Besides, the model successfully detected vehicle objects with different background conditions such as parking lots, roads, and shadows. The model was also robust when it comes to different vehicle types, sizes, and orientations. Although several vehicle targets were located close to one another, the model could draw the object boundaries correctly. Moreover, an examination of the confidence scores located in the upper left part of the bounding box delimited the targets indicated that the model consistently achieved confidence scores exceeding 90%. It can be inferred from the produced results that the model had a high level of confidence in its predictions. Additionally, the model exhibited an average prediction speed of about 62.3 milliseconds (ms) per testing image. Considering all these results, the YOLOv7 model proved to be an accurate and real-time detector for vehicle recognition applications.

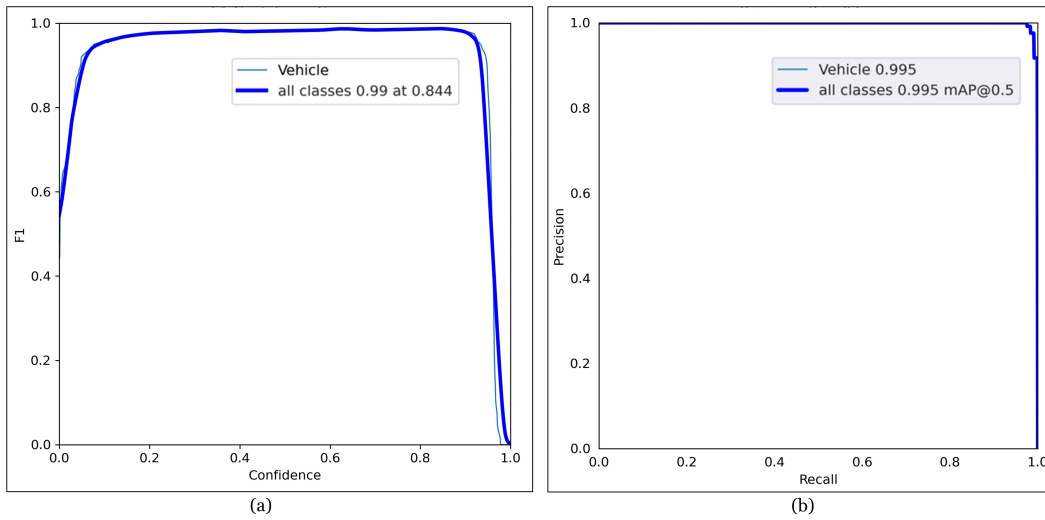


Figure 12. Learning process curves of the model: (a) F1-confidence curve, (b) precision-recall curve.

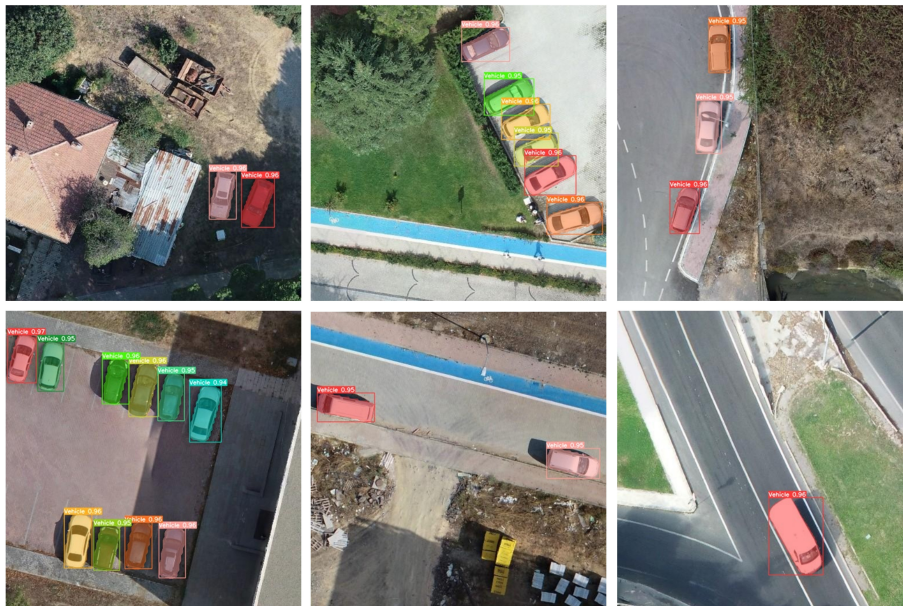


Figure 13. Sample vehicle detection results on the UAV test images.

To investigate the real-time vehicle identification and tracking performance of the YOLOv7 on video frames, a vehicle video was also fed into the model. The length of the video was 30 s and contained 726 frames. As a result of the testing process, the model achieved 0.5 ms preprocessing, 19.5 ms inference, and 1.0 ms postprocessing speed per frame. In the preprocessing stage, the image was converted to PyTorch tensors, and the pixel values were normalized to 0.0–1.0. It should be mentioned that inference time is the processing time performed within the model. In the postprocessing stage, a large number of overlapping bounding boxes produced by the model are eliminated by the NMS algorithm, keeping the most appropriate high-confidence

bounding box for the object. The sample frames generated as a result of testing the model on the video with two tasks, vehicle detection and vehicle tracking, are illustrated in Figure 14. When the vehicle identification and instance segmentation results of the model were examined, it was seen that it successfully detected and located the vehicles in the video frame (Figure 14a). As a result of the analysis performed with the confidence threshold value of 0.7, it was observed that the algorithm detected all the vehicle instances in the frame with a confidence score of more than 80%. However, the model had a locating problem in predicting a truck-type vehicle. The main reason was that there was no such type of vehicle sample in the UAV dataset in which the model was trained. On the other hand, as a result of vehicle detection and tracking of the model, there was a similar problem in the video frame (Figure 14b). Apart from vehicle detection, in this task, the way traveled by the vehicle was tracked starting from the center point of the bounding boxes that delineate the vehicles. Thereby, the blue lines in the video frame demonstrate the track where vehicles were traveling along the road. Accordingly, it was observed that the YOLOv7 exhibited superior performance in real-time moving vehicle detection from videos, which was a challenging task, in addition to vehicle detection and segmentation from UAV images.

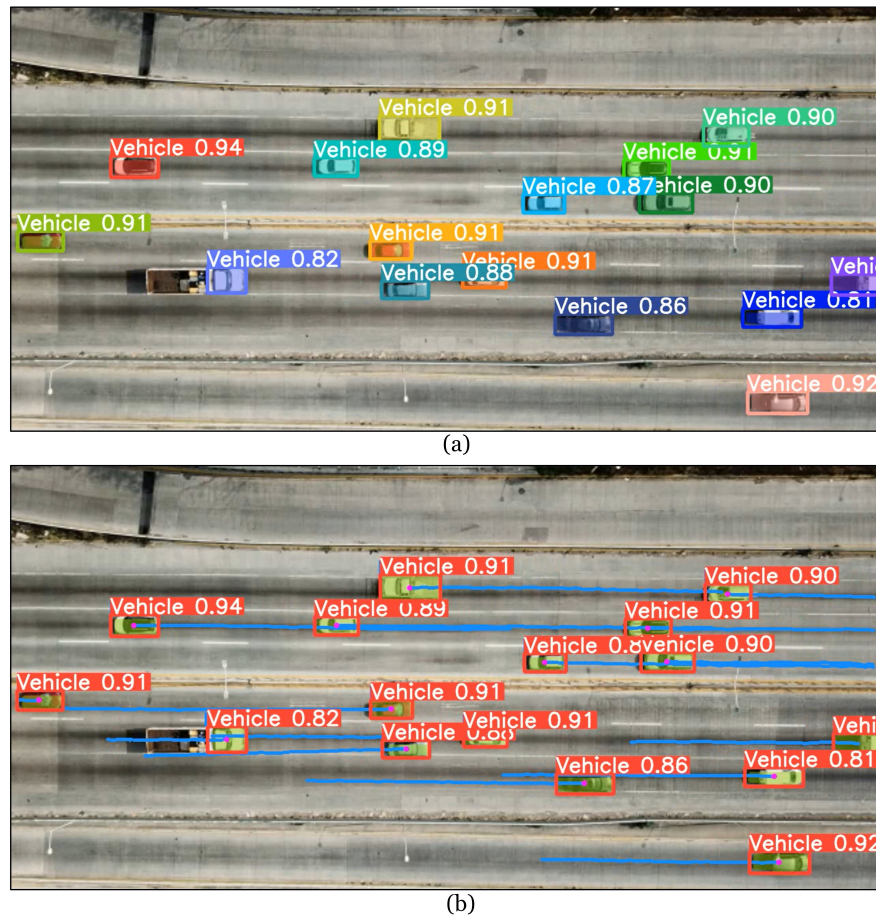


Figure 14. Testing results of the model in the video frames: (a) vehicle detection, (b) vehicle detection and tracking.

5. Discussions and conclusion

Automated vehicle detection is a challenging task, limited by several performance reducing factors such as different characteristics of vehicles, traffic density, and limited computer hardware capacity. Currently, issues such as traffic monitoring and parking management in metropolitans highly depend on automated vehicle detection. In addition, it is indispensable in search and rescue missions in accidents and disasters and, of course, for military purposes. It is a significant fact that deep learning integration in automated vehicle detection produces efficient results, especially when high-resolution data is used. In remote sensing, the highest spatial resolution data can only be achieved by unmanned aerial vehicle (UAV) technology by means of low flight altitudes. In addition, the multiaspect and oblique viewing capabilities of UAVs enable the creation of realistic 3D mesh models of objects and high-quality orthomosaics. In this study, parking and moving vehicles were automatically detected utilizing 2.5 cm GSD orthomosaic derived from 20 MP aerial photos captured by different geometry 32 UAV flights. In the identification of vehicles in the campus orthomosaic images, the popular deep learning-based YOLOv7 algorithm was utilized. Moreover, each detected vehicle was pixel-wise masked by the instance segmentation feature of YOLOv7. Thus, each vehicle within one class in an image was segmented as different samples and distinguished from the other. The mAP, precision, and recall accuracies were estimated as 99.79, 97.54, and 99.46%, respectively. A study, conducted by Chen et al. [47], employed YOLOv5 to detect vehicles from the publicly available VisDrone dataset and they obtained mAP, precision, and recall scores of 89.6, 91.9, and 82.5%, respectively. Amato et al. [49] employed the YOLOv3 model to detect and count vehicles in UAV imagery. They utilized the two public dataset, and achieved 97% precision and 95% recall for the CARPK dataset and 95% precision and 86% recall PUCPR+ dataset. Ammar et al. [50] conducted a comparative experiment between the performance of the Faster R-CNN, YOLOv3, and YOLOv4 algorithms for the detection of cars from UAV imagery. In the current research, the findings of the previous study were extended using the YOLOv7 algorithm, and performance investigation was performed for both vehicle detection and tracking. They utilized a publicly available Stanford dataset and their own PSU dataset, both of which consist of drone imagery. While they achieved the highest recall and F-1 scores on the PSU dataset using YOLOv4, they obtained the best values with Faster R-CNN on the Standford dataset. However, the recall and F-1 scores obtained in both datasets were lower than 98%. Additionally, the efficacy of the model predictions was also validated by F1-confidence and precision-recall curves in this study. As a secondary step of the analysis, the trained model was applied to a 30-s video and observed that the confidence scores for all predicted vehicles were over 80%. One of the most important findings to conclude from this study is the substantial impact of the quality and characteristics of the dataset on the performance of the deep learning algorithm. More specifically, the representative capability of the training instances has a critical effect on the accuracy and transferability of the algorithm. The employed model only failed in cases of trucks that did not exist in the training dataset as there were no trucks in the parking lots of the university campus. This is obviously a developing issue for future studies with the extension of the dataset by adding different types of cars (e.g., trucks). Given the significance of the computational cost of the deep learning methods, the training of YOLOv7 only took less than one hour and predicted cars with about 21 ms per frame of test video, indicating real-time performance. In a nutshell, the results demonstrated that the usage of a very high-resolution UAV orthomosaic and YOLOv7 deep learning algorithm together reveals high performance in automated vehicle detection.

References

- [1] Kainz O, Dopiriak M, Michalko M, Jakab F, Nováková I. Traffic monitoring from the perspective of an unmanned aerial vehicle. *Applied Sciences* 2022; 12 (16): 7966. doi: 10.3390/app12167966
- [2] Ding J, Zhang J, Zhan Z, Tang X, Wang X. A precision efficient method for collapsed building detection in post-earthquake UAV images based on the improved NMS algorithm and Faster R-CNN. *Remote Sensing* 2022; 14 (3): 663. doi: 10.3390/rs14030663
- [3] Zhang X, Han L, Dong Y, Shi Y, Huang W et al. A deep learning-based approach for automated yellow rust disease detection from high-resolution hyperspectral UAV images. *Remote Sensing* 2019; 11 (13): 1554. doi: 10.3390/rs11131554
- [4] Kose O, Oktay T. Simultaneous design of morphing hexarotor and autopilot system by using deep neural network and SPSA. *Aircraft Engineering and Aerospace Technology* 2023; 95 (6): 939-949. doi: 10.1108/AEAT-07-2022-0178
- [5] Şahin H, Kose O, Oktay T. Simultaneous autonomous system and powerplant design for morphing quadrotors. *Aircraft Engineering and Aerospace Technology* 2022; 94 (8): 1228-1241. doi: 10.1108/AEAT-06-2021-0180
- [6] Kose O, Oktay T. Simultaneous quadrotor autopilot system and collective morphing system design. *Aircraft Engineering and Aerospace Technology* 2020; 92 (7): 1093-1100. doi: 10.1108/AEAT-01-2020-0026
- [7] Ammar A, Koubaa A, Ahmed M, Saad A, Benjdira B. Vehicle detection from aerial images using deep learning: A comparative study. *Electronics* 2021; 10 (7): 820. doi: 10.3390/electronics10070820
- [8] Bouguettaya A, Zarzour H, Kechida A, Taberkit AM. Vehicle detection from UAV imagery with deep learning: A review. *IEEE Transactions on Neural Networks and Learning Systems* 2021; 33 (11): 6047-6067. doi: 10.1109/tnnls.2021.3080276
- [9] Srivastava S, Narayan S, Mittal S. A survey of deep learning techniques for vehicle detection from UAV images. *Journal of Systems Architecture* 2021; 117: 102152. doi: 10.1016/j.sysarc.2021.102152
- [10] Wang X. Vehicle image detection method using deep learning in UAV video. *Computational Intelligence and Neuroscience* 2022; 8202535. doi: 10.1155/2022/8202535
- [11] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05); San Diego, CA, USA; 2005. pp. 886-893. doi: 10.1109/CVPR.2005.177
- [12] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features. In: 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'01); Kauai, HI, USA; 2001. pp. 511-518. doi: 10.1109/CVPR.2001.990517
- [13] Fei-Fei L, Perona P. A Bayesian hierarchical model for learning natural scene categories. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05); San Diego, CA, USA; 2005. pp. 524-531. doi: 10.1109/CVPR.2005.16
- [14] Mountrakis G, Im J, Ogole C. Support vector machines in remote sensing: A review. *ISPRS Journal of Photogrammetry and Remote Sensing* 2011; 66 (3): 247-259. doi: 10.1016/j.isprsjprs.2010.11.001
- [15] Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 1997; 55 (1): 119-139. doi: 10.1006/jcss.1997.1504
- [16] Peterson LE. K-nearest neighbor. *Scholarpedia* 2009; 4(2): 1883. doi: 10.4249/scholarpedia.1883
- [17] Kavzoglu T, Erdemir MY, Tonbul H. Classification of semiurban landscapes from very high-resolution satellite images using a regionalized multiscale segmentation approach. *Journal of Applied Remote Sensing* 2017; 11 (3): 035016-035016. doi: 10.1117/1.JRS.11.035016

- [18] Kavzoglu T, and Tonbul H. A comparative study of segmentation quality for multi-resolution segmentation and watershed transform. In: 8th International Conference on Recent Advances in Space Technologies (RAST); İstanbul, Türkiye; 2017. pp. 113-117. doi: 10.1109/RAST.2017.8002984
- [19] Yildirim E, Kavzoglu T. Ship detection in optical remote sensing images using YOLOv4 and Tiny YOLOv4. In: Ben Ahmed M, Boudhir AA, Karaş İR, Jain V, Mellouli S (eds). Innovations in Smart Cities Applications Volume 5. SCA 2021. Lecture Notes in Networks and Systems, vol 393. Springer, Cham, 2022, pp. 913-924. doi: 10.1007/978-3-030-94191-8_74
- [20] Coşkun D, Karaboğa D, Baştürk A, Akay B, Nalbantoğlu OU et al. A comparative study of YOLO models and a transformer-based YOLOv5 model for mass detection in mammograms. Turkish Journal of Electrical Engineering and Computer Sciences 2023; 31 (7): 1294-1313. doi: 10.55730/1300-0632.4048
- [21] Yildirim E, Kavzoglu T. Detection of collapsed buildings from post-earthquake imagery using Mask Region-Based Convolutional Neural Network. In: 7th Intercontinental Geoinformation Days (IGD); Peshawar, Pakistan; 2023. pp. 119-122.
- [22] Kavzoğlu T, Yılmaz EÖ. Analysis of patch and sample size effects for 2D-3D CNN models using multiplatform dataset: hyperspectral image classification of ROSIS and Jilin-1 GP01 imagery. Turkish Journal of Electrical Engineering and Computer Sciences 2022; 30 (6): 2124-2144. doi: 10.55730/1300-0632.3929
- [23] Shabaz M, Soni M. Cognitive digital modelling for hyperspectral image classification using transfer learning model. Turkish Journal of Electrical Engineering and Computer Sciences 2023; 31(6): 1039-1060. doi: 10.55730/1300-0632.4033
- [24] Chen G, Chen Q, Long S, Zhu W, Yuan Z et al. Quantum Convolutional Neural Network for image classification. Pattern Analysis and Applications 2023; 26 (2): 655-667. doi: 10.1007/s10044-022-01113-z
- [25] Gordo A, Almazán J, Revaud J, Larlus D. Deep image retrieval: Learning global representations for image search. In: 14th European Conference on Computer Vision (ECCV'16); Amsterdam, The Netherlands; 2016: 241-257.
- [26] Öztürk Ş, Alhudhaif A, Polat K. Attention-based end-to-end CNN framework for content-based X-ray image retrieval. Turkish Journal of Electrical Engineering and Computer Sciences 2021; 29 (8): 2680-2693. doi: 10.3906/elk-2105-242
- [27] Badrinarayanan V, Kendall A, Cipolla R. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence 2017; 39 (12): 2481-2495. doi: 10.1109/TPAMI.2016.2644615
- [28] Bagheri F, Tarokh MJ, Ziaratban M. Skin lesion segmentation by using object detection networks, DeepLab3+, and active contours. Turkish Journal of Electrical Engineering and Computer Sciences 2022; 30 (7): 2489-2507. doi: 10.55730/1300-0632.3951
- [29] Yildirim E, Nazar M, Sefercik UG, Kavzoglu T. Stone Pine (Pinus Pinea L.) detection from high-resolution UAV Imagery using deep learning model. In: IEEE International Geoscience and Remote Sensing Symposium (IGARSS'22); Kuala Lumpur, Malaysia; 2022. pp. 441-444. doi: 10.1109/IGARSS46834.2022.9883964
- [30] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR); Columbus, OH, USA; 2014. pp. 580-587. doi: 10.1109/CVPR.2014.81
- [31] Girshick R. Fast R-CNN. In: IEEE International Conference on Computer Vision (ICCV); Washington, DC, USA; 2015. pp. 1440-1448. doi: 10.1109/ICCV.2015.169
- [32] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence 2017; 39 (6): 1137-1149. doi: 10.1109/TPAMI.2016.2577031

- [33] He K, Zhang X, Ren S, Sun J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2015; 37 (9): 1904-1916. doi: 10.1109/TPAMI.2015.2389824
- [34] Dai J, Li Y, He K, Sun J. R-FCN: Object detection via region-based fully convolutional networks. In: 30th Annual Conference on Neural Information Processing Systems (NIPS'16); Barcelona, Spain; 2016. pp. 379-387.
- [35] He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 2020; 42 (2); 386-397. doi: 10.1109/TPAMI.2018.2844175
- [36] Redmon J, Divvala S, Girshick R, Farhadi A. You Only Look Once: Unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'16); Las Vegas, NV, USA; 2016. pp. 779-788. doi: 10.1109/CVPR.2016.91
- [37] Redmon J, Farhadi A. YOLO9000: Better, faster, stronger. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'17); Honolulu, Hawaii; 2017. pp. 7263-7271. doi: 10.1109/CVPR.2017.690
- [38] Redmon J, Farhadi A. YOLOv3: An incremental improvement. arXiv: 1804.02767v1, 2018. Available: <https://doi.org/10.48550/arXiv.1804.02767>
- [39] Bochkovskiy A, Wang CY, Liao HYM. YOLOv4: Optimal speed and accuracy of object detection. arXiv: 2004.10934, 2020. Available: <https://doi.org/10.48550/arXiv.2004.10934>
- [40] Jocher G. YOLOv5 by Ultralytics. <https://github.com/ultralytics/yolov5>, 2020. Accessed: June 06, 2023.
- [41] Li C, Li L, Jiang H, Weng K, Geng Y et al. YOLOv6: A single-stage object detection framework for industrial applications. arXiv: 2209.02976, 2022. Available: <https://doi.org/10.48550/arXiv.2209.02976>
- [42] Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv: 2207.02696, 2022. Available: <https://doi.org/10.48550/arXiv.2207.02696>
- [43] Liu W, Anguelov D, Erhan D, Szegedy C, Reed S et al. SSD: Single shot multibox detector. In: European Conference on Computer Vision (ECCV'16); Amsterdam, The Netherlands; 2016. pp. 21-37.
- [44] Lin TY, Goyal P, Girshick R, He K, Dollár P. Focal loss for dense object detection. In: IEEE International Conference on Computer Vision; Venice, Italy; 2017. pp. 2980-2988. doi: 10.1109/ICCV.2017.324
- [45] Li S, Yang X, Lin X, Zhang Y, Wu J. Real-time vehicle detection from UAV aerial images based on improved YOLOv5. *Sensors* 2023; 23 (12): 5634. doi: 10.3390/s23125634
- [46] Xie X, Yang W, Cao G, Yang J, Zhao Z et al. Real-time vehicle detection from UAV imagery. In: 2018 IEEE Fourth International Conference on Multimedia Big Data (BigMM); Xi'an, China; 2018. pp. 1-5. doi: 10.1109/BigMM.2018.8499466
- [47] Chen Z, Cao L, Wang Q. YOLOv5-based vehicle detection method for high-resolution UAV images. *Mobile Information Systems* 2022; 2022: 1828848. doi: 10.1155/2022/1828848
- [48] Ammar A, Koubaa A, Ahmed M, Saad A, Benjdira B. Vehicle detection from aerial images using deep learning: A comparative study. *Electronics* 2021; 10 (7): 820. doi: 10.3390/electronics10070820
- [49] Amato G, Ciampi L, Falchi F, Gennaro C. Counting vehicles with deep learning in onboard UAV imagery. In: 2019 IEEE Symposium on Computers and Communications (ISCC); Barcelona, Spain; 2019. pp. 1-6. doi: 10.1109/ISCC47284.2019.8969620
- [50] Benjdira B, Khurshed T, Koubaa A, Ammar A, Ouni K. Car detection using Unmanned Aerial Vehicles: Comparison between Faster R-CNN and YOLOv3. In: 2019 1st International Conference on Unmanned Vehicle Systems-Oman (UVS); Muscat, Oman; 2019. pp. 1-6. doi: 10.1109/UVS.2019.8658300
- [51] Chen S, Laefer DF, Mangina E. State of technology review of Civilian UAVs. *Recent Patents Engineering* 2016; 10 (3): 160-174. doi: 10.2174/1872212110666160712230039

- [52] Udeanu G, Dobrescu A, Oltean M. Unmanned aerial vehicle in military operations. *Scientific Research and Education in the Air Force* 2016; 18 (1): 199-206. doi: 10.19062/2247-3173.2016.18.1.26
- [53] Kretov A, Glukhov V, Tikhonov A. Conceptual assessment of the possibility of using cryogenic fuel on unmanned aerial vehicles. *Drones* 2022; 6 (8): 217. doi: 10.3390/drones6080217
- [54] Westoby MJ, Brasington J, Glasser NF, Hambrey MJ, Reynolds JM. Structure-from-Motion'photogrammetry: A low-cost, effective tool for geoscience applications. *Geomorphology* 2022; 179: 300-314. doi: 10.1016/j.geomorph.2012.08.021
- [55] Sefercik UG, Kavzoglu T, Nazar M, Atalay C, Madak M. Creation of a virtual tour .exe utilizing very high-resolution RGB UAV data. *International Journal of Environment and Geoinformatics* 2022; 9 (4): 151-160. doi: 10.30897/ijegeo.1102575
- [56] Baltasvias E. A comparison between photogrammetry and laser scanning. *ISPRS Journal of Photogrammetry and Remote Sensing* 1999; 54 (2-3): 83-94. doi: 10.1016/S0924-2716(99)00014-3
- [57] Jacobsen K. Characteristics of nearly world-wide available digital height models. In: *10th Seminar on Remote Sensing and GIS Applications in Forest Engineering*; Curitiba, Brazil; 2012. pp. 15-18.
- [58] Sefercik UG, Glennie C, Singhanian A, Hauser D. Area-based quality control of airborne laser scanning 3D models for different land classes using terrestrial laser scanning: sample survey in Houston, USA. *International Journal of Remote Sensing* 2015; 36 (23): 5916-5934. doi: 10.1080/01431161.2015.1110260
- [59] Wada CY, Labelme: Image polygonal annotation with Python. <https://github.com/wkentaro/labelme>, 2023. Accessed: July 07, 2023.
- [60] Bodla N, Singh B, Chellappa R, Davis LS. Soft-NMS – Improving object detection with one line of code. In: *IEEE International Conference on Computer Vision (ICCV)*; Venice, Italy; 2017. pp. 5561-5569. doi: 10.1109/ICCV.2017.593
- [61] Qiu Z, Bai H, Chen T. Special vehicle detection from UAV Perspective via YOLO-GNS based deep learning network. *Drones* 2023; 7 (2): 117. doi: 10.3390/drones7020117