IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Vision-based Moving UAV Tracking by Another UAV on Low-Cost Hardware and A New Ground Control Station

EMRE ÇİNTAŞ[1,2], (Student Member, IEEE), BARIŞ ÖZYER[1], AND EMRAH ŞİMŞEK[1,2]

[1]Department of Computer Engineering, Atatürk University, 25240 Erzurum, Turkey
[2]Department of Computer Engineering, Erzurum Technical University, 25050 Erzurum, Turkey

Corresponding author: Emre Çintaş (e-mail: emrecintas@atauni.edu.tr)

This work was supported by the Atatürk University Scientific Research Projects Coordination Unit under the project FBA-2020-8447.

ABSTRACT Automatic flying target detection and tracking in video sequences acquired from a camera mounted on another Unmanned Aerial Vehicle (UAV) is a challenging task due to the presence of non-stationary cameras in the system, dynamic motion of the moving target, and high-cost computation for real-time applications. In this paper, our aim is to automatically detect and track moving UAV by another one while simultaneously flying in the air. In order to provide efficiently in real-time applications, we develop a vision-based low-cost hardware system integrated with an independent ground control station. We initially created a new public dataset called ATAUAV that includes different types of UAV images obtained from videos recording in our experiments and searches on Google Images for the training process. Deep learning-based YOLOv3-Tiny (You Only Look Once) is used for target detection with the highest accuracy and fastest results. Kernelized Correlation Filter (KCF) adapted with YOLO, which runs on low-cost hardware, is used for real-time detected target tracking. We compared the performance of the proposed approach with different tracking algorithms. Experimental results show that the proposed approach provides the highest accuracy rate as 82.7% and a mean fps speed as 29.6 on CPU. The dataset can be downloaded at http://cogvi.atauni.edu.tr/ResearchLab/PageDetail/Our-ATAUAVs-Dataset-86.

INDEX TERMS Artificial neural networks, computer vision, kcf, object detection, object recognition, target tracking, unmanned aerial vehicles, yolo

I. INTRODUCTION

Most recently developed UAVs are equipped with a static or nonstationary camera; the operator can watch the camera footage in real-time through video streaming while the UAV is in flight [1], [2]. Advancing embedded high-speed cameras, these robotic systems enable the automatic detection and tracking of moving objects via ground control stations (GCSs) [3], such as cars [3], humans [4], and aircraft [5], in different applications [6]. However, tracking an object in video frames captured by fast-moving cameras or other targets remain a challenging problem in computer vision [1]. The reason is that the complication of camera movement may lead to complex changes in illumination, rotation, and scale invariance between images. In addition, background clutter, low resolution, aerodynamic effect, occlusion, low contrast, image noise, and small size of the target object worsen the problem.

In recent years, correlation filter (CF)-based methods [7], [8] are widely used in tracking objects under controlled conditions using static cameras. The correlation is a similarity measure between two consecutive frames. A CF efficiently distinguishes between the background and the target object in the Fourier domain. Kernelized CF (KCF) [9] is a well-known CF tracker-based method. The KCF algorithm successfully works on hundreds of frames per second (fps) because the training samples are created with cyclic shifts that convert the data matrix to circulant matrices [10]. Depending on the property of the circulant matrices, the solution of the problem is converted into a Discrete Fourier Transform (DFT) domain. Therefore, the data matrix prevents the reversal process and reduces the complexity by several degrees of magnitude. In addition, HOG [11], [12], LBP [13], and SIFT [14] features are used to increase a CF's tracking accuracy. In [15], the authors proposed a special

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033481, IEEE Access

filter called aberrance repressed CF to track human activity videos captured from UAV. This filter can prevent image anomalies caused by dynamic scene noises while tracking an object. An automatic spatio-temporal regularization framework [16] was proposed to restrict local CF learning that tracks cars, bicycles, and other objects with UAVs. In [17], the authors proposed an adaptive algorithm that combines CF tracking response based on multi-channel HOG features and enhanced color histogram tracking response to track small targets, such as trucks, cars, and bikes, with high accuracy. Although these filter-based methods perform fast response in static cameras and controlled conditions, they are not successful in practical applications. Several algorithms, such as CSRT [18], TLD [19], MIL [20], and BACF [21] have been proposed to track objects in controlled environments. However, the fps rates of these algorithms are very low compared with that of KCF.

Several deep learning-based algorithms, such as YOLO [22], SSD [23], and convolutional neural network (CNN) [24] have recently been utilized to develop robust object tracking methods in real-time applications because of their high-accuracy results [1], [26]. In [22], the authors used YOLOv3 for detection and a Kalman filter algorithm for tracking pedestrians in video frames. In [23], a real-time human tracking system based on the SSD architecture was developed for AR Drone 2. A multi-block SSD algorithm was proposed to detect and recognize unauthorized persons' entrance into railway stations by video surveillance [24]. An interesting study was proposed in [25], wherein cattle raised in farms are detected in real-time videos captured by a drone using a CNN. In [26], the authors developed a GPU-based new tracking algorithm with deep regression networks called GOTURN.

Although deep learning algorithms yield accurate results in object detection and tracking problems, they require high-performance computational systems. In real-time applications, deep learning algorithms should be run on GPUs since the FPS rates of GPUs are higher than those of CPUs. However, GPUs are more costly. It is also **risky** due to the possible falling and breaking cases of UAVs during experiments. Therefore, low-cost CPU-based hardware designs are a more appropriate platform than high-cost GPUs for facilitating high-speed processes in UAVs.

Given the above motivation, we propose a method that accurately detects and tracks moving UAVs in videos on **CPU**-based **low-cost hardware** with a **mean fps of ~30**. The proposed method combines the deep learning-based object detection algorithm YOLO [27] and the tracking algorithm KCF [9]. Therefore, in this study,

- We introduce a novel UAV dataset for validation of the proposed method and use of future works;
- We propose a novel tracking method that works on real-time low-cost hardware systems with collected datasets; and
- We develop a computer vision-based new GCS for controlling UAVs and monitoring the tracking performance of UAVs over time.

The rest of this paper is organized as follows: In Section 2, the methodology, the UAV dataset, and the designed GCS are described in detail. In Section 3, the experimental results are discussed. We conclude the paper and give future directions of study in Section 4.

II. METHODOLOGY

In this section, we initially present the study problem by giving a scenario for evaluating the performance of the method. The proposed method then is described, followed by a brief discussion on the theory of detection and tracking algorithms. The proposed GCS is described to control and illustrate the system in detail.

A. PROBLEM DEFINITION

Assume that the target UAV is departing from point (w_i) and follows route (P_T). Another UAV simultaneously follows route (P_U) and tracks the detected target UAV, as shown in Fig. 1. The target UAV under **controlled** conditions (stable and without considerable changes in color, texture, light, etc.) starts a maneuvering movement while approaching point (w_{i+1}). However, when the moving target starts dynamically moving from point (w_{i+1}) to point (w_{i+n}), it cannot be tracked under the **uncontrolled** conditions due to illumination, aerodynamic effect and the other variations of flight, which cause texture and appearance distortion in video frames.

In this context, V represents the velocity of the air flowing around the solid object, α refers to the angle of attack of the aircraft, L is the aerodynamic force which is perpendicular to the air velocity (lift), D is the component parallel to the air velocity (drag), and finally, F_A is the aerodynamic effect force resultant affecting a solid object [28].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033481, IEEE Access

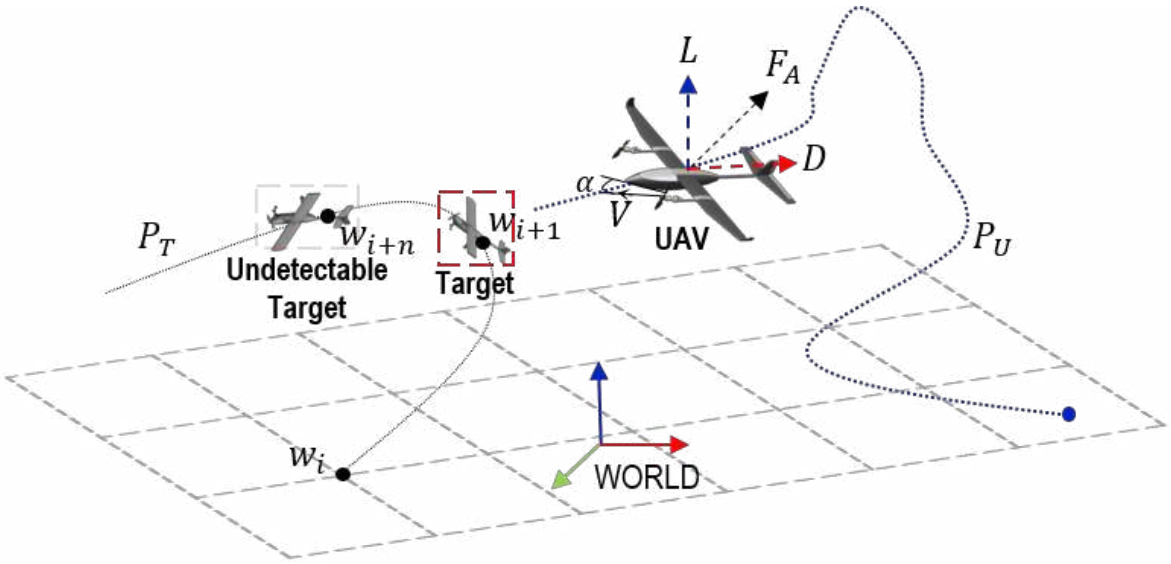

FIGURE 1. **Problem formulation. Failure to track unmanned aerial vehicles in real-time due to the dynamic movements, pose-texture differences, aerodynamic effects, and other variations**

B. OBJECT DETECTION BASED ON YOLO

YOLO [27] is a convolutional neural network architecture proposed for real-time detection of objects in high-speed videos. YOLO network turns the detection problem into a regression problem that image is used only as an input network once instead of not processed separately for each class. Therefore, it is possible to determine the location of detected objects in the video frames which is considerably faster than the traditional object detection algorithms. Decreasing detection time significantly affects the **time consumption**, **energy consumption, and efficiency of UAVs**.

It is well known that YOLO is a convolutional neural network that consists of convolution, pooling, and fully-connected layers [29]. In a traditional convolutional neural network, $z_{i,j,n}^{m}$ is the feature value of the m layer n feature map of the position (i, j) calculated as following [30],

$$z_{i,j,n}^{m} = w_n^{m^T} x_{i,j}^{m} + b_n^{m} \quad (1)$$

w_n^m is the weight vector, b_n^m is bias and $x_{i,j}^m$ is the input part of the m layer at the position (i, j) [30]. The ReLu (Rectified Linear Unit) activation function is used to detect the non-linear features calculated by,

$$a_{i,j,k} = \max(z_{i,j,k}, 0) \quad (2)$$

where $z_{i,j,k}$ is the activation input at the position (i, j) at the k channel [30]. The pooling layer is used after a convolutional layer, reducing the size of feature maps and network parameters [30]. Pooling function $y_{i,j,n}^m$ is calculated by the following equation,

$$y_{i,j,n}^{m} = \text{pool}\,(a_{f,g,n}^{m}), \forall\, (f,g) \in R_{ij} \quad (3)$$

where $a_{i,j,n}^m$ is the feature map, and R_{ij} refers to the local neighborhoods of the position (i, j). In the last stage of the network, several fully-linked layers are used to convert 2D feature maps into a 1D feature vector [29]. In this output layer, the SVM algorithm which can be generally combined with the softmax operator or CNN is used for classification. The best parameters for the classification process are obtained by minimizing the loss function. For N input-output relationships $\{(x^{(n)}, y^{(n)});\ n \in [1, \cdots, N]\}$, $x^{(n)}$ is the n input data, $y^{(n)}$ is class tags, and $o^{(n)}$ is the output of the convolutional neural network and N is the number of samples. The loss function L is calculated as follows [30]:

$$L = \sum_{n=1}^{N} l = (\theta; y^{(n)}, o^{(n)}) \quad (4)$$

In this study, several experiments were carried out for the selection of YOLO parameters. The YOLOv3 consists of two main networks: i) Feature extraction network: ***(416 x 416 x 3)*** resized images are used as an input. ii) Object detector network: (13 x 13 x 225) and (26 x 26 x 225) feature map scales are merged with an upsampled ***13×13*** feature map. We should here emphasize that the object detection stage for each frame, YOLO runs at 30+ fps on high-end GPUs and requires more processing power compared to the object tracking method **increasing** the UAV **energy consumption**. However, compared to the earlier YOLO version, the speed can be increased up to 100+ fps on the GPU. In the object tracking process, there is no need for an object detection layer in every image frame. Therefore, there is a need for an object tracking method such as KCF running in the CPU. The KCF, which has the highest speed in the literature, can run at 100+ fps on the CPU. Therefore, the process requirement of KCF is much less than the other methods.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033481, IEEE Access

C. KCF TRACKING ALGORITHM

In the Kernelized Correlation Filter (KCF) [9], the relationship of the circulant matrix with the discrete Fourier transform is revealed so that tracking can be done quickly. The rows of the circulant matrix consist of the target model and the cyclical shifts of the model. Ridge regression for the learning of image windows is defined in the frequency domain by discrete Fourier transform. The weight parameters obtained after learning the image windows are multiplied by the test images in the frequency domain, and the possible position of the target is found. In the learning stage, three-dimensional color or oriented gradient histogram is used as feature vectors. The ridge regression problem is linearized by the kernel trick as follows.

As shown in Equation (5), y_i represents a target, $f(z) = w \top z$ represents function and x_i refers to examples, and their regression targets minimize the square error on y_i.

$$\min_{w} \sum_{i} (f(x_i) - y_i)^2 + \lambda \, [w]^2 \tag{5}$$

where λ is the regularization parameter for overfitting. The frequency representation is described as:

$$\hat{w} = \frac{\hat{x}^* \odot \hat{y}}{\hat{x}^* \odot \hat{x} + \lambda} \tag{6}$$

where $\hat{w}$ specifies the Fourier transform of w, $\hat{x}^*$ is the complex conjugate of $\hat{x}$, and $\odot$ refers to the intelligent result as an element-wise production.

As far as we know that the KCF algorithm is the fastest one in the literature for object tracking in video frames and provides high accuracy under controlled conditions in general. However, in cases where the target is overlapped and the target is out of sight because it only searches around its previous position, the follow-up task will fail due to no self-correction mechanism. In addition to its flaws, the aspect ratio does not change dynamically.

D. PROPOSED METHOD

Fig. 2. shows the block diagram of the proposed method that detects and tracks a moving object in real-time. The proposed method includes the YOLOv3-Tiny [27] network first when it starts an instant UAV search process in the first frame. If there is no UAV object in the scene, the algorithm works stably. If a UAV can be detected and recognized in each 1 and 30 x n frames, the bounding box information of the relevant object is given to the KCF [9] algorithm and the tracking process is started. Then, in every 30 x n frames and in cases where the KCF algorithm fails, the processes in the control phase are activated. In the control phase, the Euclidean distance between the central coordinates of the final results is estimated by the YOLO. KCF algorithms are examined and the closest coordinates are found on a metric basis. The new calculated coordinates are given to the KCF algorithm through the update process and the KCF algorithm is retrained.

The proposed real-time method automatically detects and tracks the UAV apparent in an image sequence through object detection, localization, and correction process. The details of the proposed method are shown in Algorithm 1. YOLO central coordinates x_{D_j}, y_{D_j} are updated in every thirty frames as a balance value between accuracy and speed.

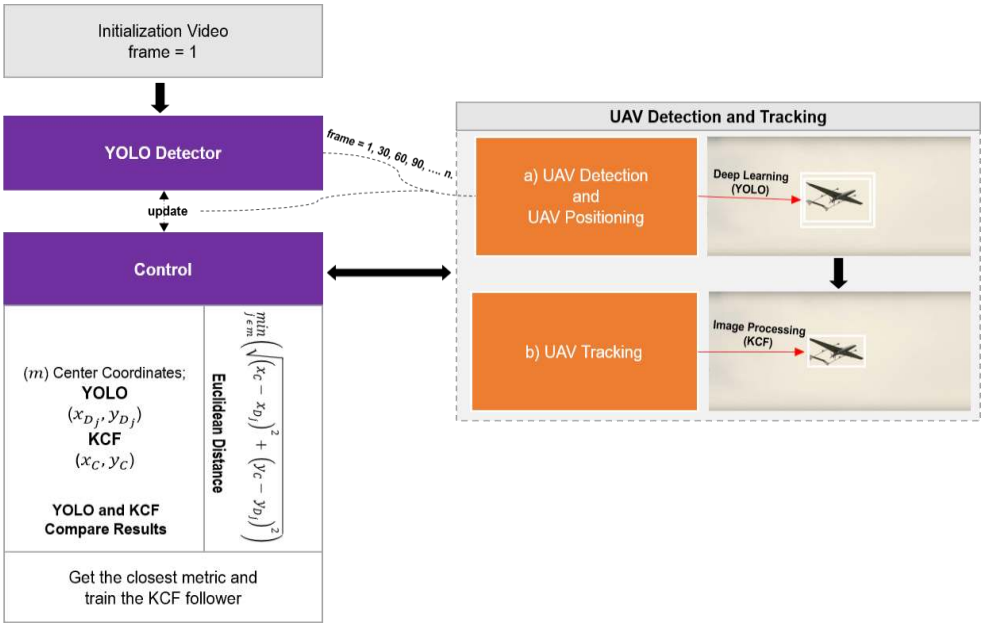

FIGURE 2. The algorithm framework

The YOLO algorithm only needs to be optimized to focus on **object detection** in the video frame and operate at high speed on machines without graphics cards. In cases where the KCF method fails due to problems caused by changes in scaling, lighting, transformation, and closure in the video frames, there is a need for an algorithm to find the position of the moving object again. For this reason, while YOLO is activated in the detection and localization of an object in every thirty frames, the KCF method is used only for **tracking** the detected object. As a result, a new approach has been designed that combines YOLO for object detection and KCF for tracking together in every 30 frames. In order to achieve a successful result, the YOLO network was first trained with a dataset composed of different UAV images captured from the real-time experiments. Then, considering the sudden frame jumps and acrobatic movements, it is predicted that the results of YOLO object detection may not coincide with the results of the KCF tracker. By checking the mentioned prediction in every thirty frames, as indicated in Eq. (7), the closest one on the metric basis was determined by using Euclidean distance between the (m) central coordinates of the results predicted by YOLO where (x_{D_j}, y_{D_j}) and the central coordinates of the last result produced by KCF (x_C, y_C), and the KCF tracker was trained.

$$\min_{j \in m}\left(\sqrt{\left(x_C - x_{D_j}\right)^2 + \left(y_C - y_{D_j}\right)^2}\right) \tag{7}$$

Determining the ideal threshold value to be thirty as a result of the trials and running of YOLO object detection by the algorithm once, in only thirty frames enabled the proposed method to run fast and stable without a graphic card. Consequently, the proposed method took advantage of accurate object detection and dynamic positioning from YOLO while using the KCF for speed and locking to a single target.

ALGORITHM 1. **UAV Tracking with Deep Learning and Image Processing**

Input: image *I*, center coordinates $x_C, y_C, x_{D_j}, y_{D_j}$, *x*, *y*, width *w*, height *h*, frame id F_{id}

Output: target state $\hat{s}_t$

```
Set F_id to zero and tracker is YOLO
while true
    F_id = F_id + 1
    if mod(F_id, 30) = 0 or if (F_id == 1) then
        tracker is YOLO
        x_Dj, y_Dj, w, h = Get UAV Detection and Positioning Bounding Box
    else
        tracker is KCF
        x_c, y_c, w, h = Get UAV Tracking Bounding Box
    end
    Control:
        x, y, w, h = Compare results for YOLO and KCF using equation (7)
    if (w>0) then
        Train the KCF follower with (x, y, w, h information)
        UAV Tracking
    end
end
```

1) DATASET COLLECTION

Although there are many public data sets of human [4], car [3] for target tracking, there is no data set consisting only of UAVs concerning our specific problem. For that reason, the ATAUAV dataset which is one of our contributions is developed as a benchmark dataset for future scientific studies. Fig. 3. shows some sample images of our UAV dataset. Approximately 4500 images are collected from videos captured during test flights with our developed UAVs at Ataturk University. The videos are captured under an unconstrained environment with different weather conditions such as cloudy, cloudless, and rainy. The rest of the images are collected from searches on Google Images for the training process. We have a total of 10000 UAV images consisting of various UAVs in the dataset. All UAV images are labeled frame by frame to obtain ground truth data that includes the centroid position of the detected object and Δ_x width and Δ_y height distance parameters of bounding boxes. The number of images in the datasets is increased by 35% percent with the data augmentation technique and used in test and train. Besides, in order to prevent over-fitting, the images with moving objects such as humans and cars are added %10 percentage to the dataset. 4000 images of the ATAUAV dataset were used to train the YOLOv3-Tiny network and

6000 images were used to the test of models. Fig. 3. shows samples images from our dataset.

2) GROUND CONTROL STATION (GCS) DESIGN AND IMPLEMENTATION

The unmanned aerial vehicle (UAV) can be either controlled by a ground control station (GCS) by a pilot or autonomously controlled by embedded systems without a pilot [31]. The main purpose of our proposed and designed a new GCS is due to the lack of unique ground control station software which is **computerized vision-based** and able to **track targets at a low-cost** today. Fig. 4. shows the CPU-based ground station software diagram. GCS controls the UAV movement along the basis axis. According to our tracking algorithm implemented at this station, based on the UAV movement axis control parameters, **X**-axis is used to control the roll (φ) and yaw (ψ) positions, **Y**-axis is used to control the altitude **Z** [26]. All these control parameters are tuned by PID controllers. The distance and position of a target object provided by the proposed tracking algorithm are used as input through the MAVLink [32-33] protocol. A dynamic front-facing camera is used to calculate the distance and position of the target. When the dynamic camera is mounted such that the camera looks down and the top of the image points to the front of the UAV, the roll, yaw, and pitch angles are defined as follows:

Roll (φ);

- This φ value is usually 0°

Yaw (ψ);

- If ψ = 0° and direct the camera towards to ground refers the top of the image points to the north
- If ψ = 90° and direct the camera towards nadir (i.e. nadir), refers to the top of the image points to the east
- If ψ = 270° and direct the camera towards nadir(i.e. nadir), refers top of the image points to the west

Pitch (p);

- If p = 0°, it means that the camera is looking down
- If p = 90°, it means that the camera is looking forward

The communication between UAV designed by uniquely and GCS is provided by the MAVLink [32-33] protocol that allows wireless bidirectional communication with UAV.

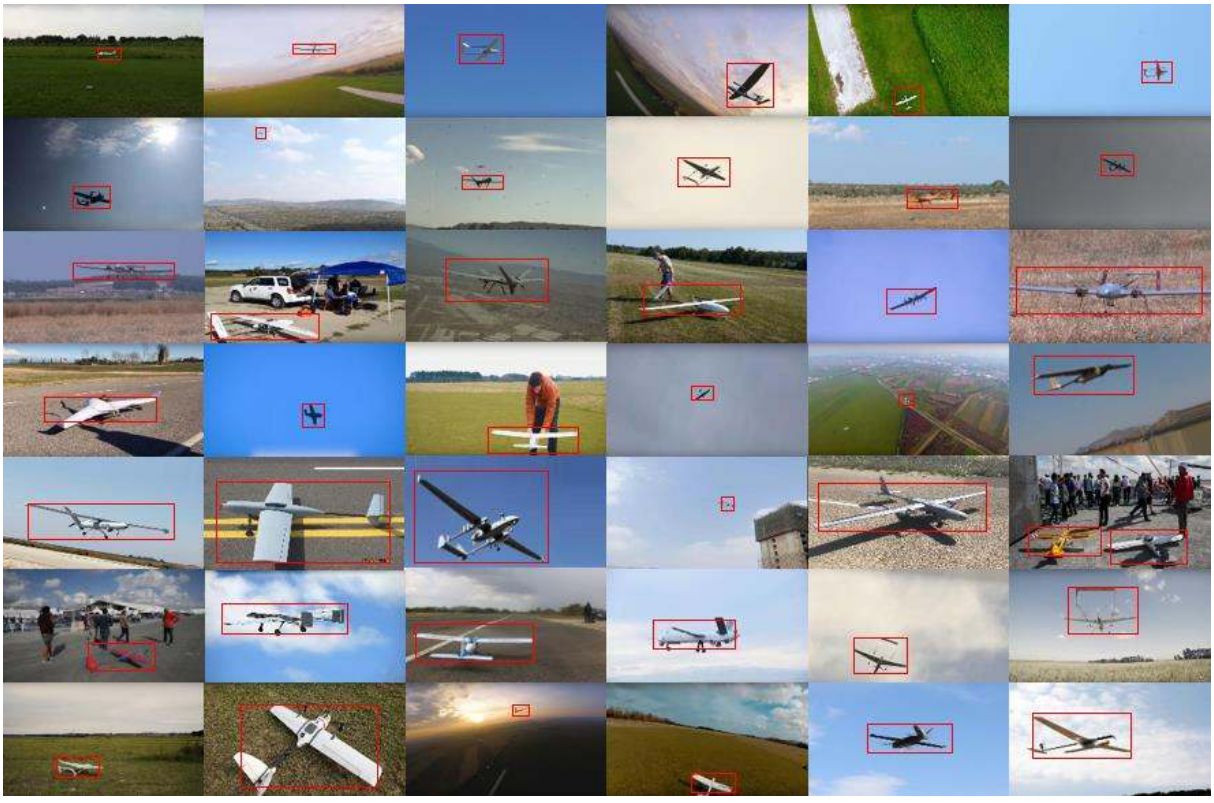

FIGURE 3. Sample frames from our ATAUAV dataset. The red bounding box indicates the ground truth annotation

While GCS can send commands to control UAV position and direction, the UAV sends back information of its status [34]. Then, the target detected by the proposed method we recommend is displayed safely with the class name on the ground station monitor. We observed that the proposed method is **fast** and **smooth** in this system to detect and track a UAV target.

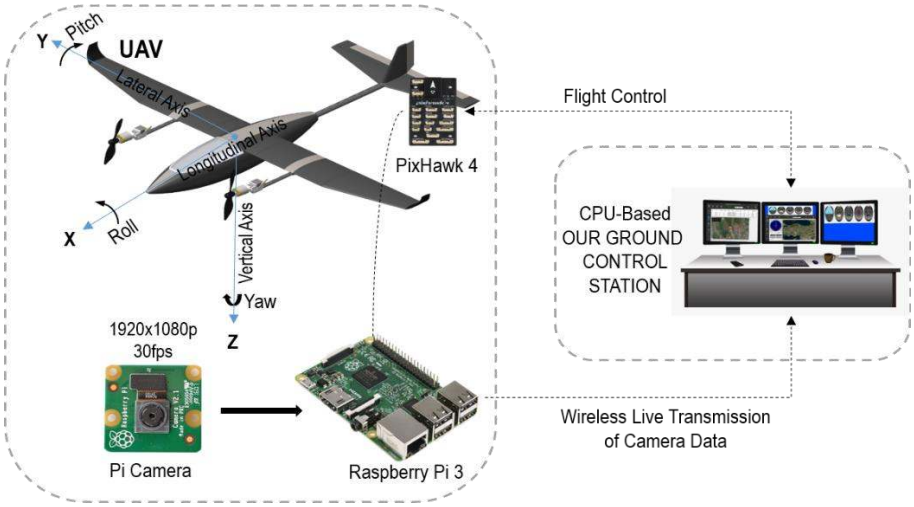

FIGURE 4. **UAV detection and tracking using the CPU-based ground control station from an UAV and basic axis of movement of UAV**

III. EXPERIMENTS AND RESULTS

Experiments were carried out for the YOLOv3-Tiny network. While training the model, Google Colab was utilized and the NVIDIA Tesla P100 graphics card was used. While interpreting the results, Intel® Core ™ i5-7400 CPU 3.00GHz processor and Raspberry Pi 4, 4GB was used. In the experiment, as a measurement parameter, the precision and recall values of the UAV class with 6000 test images were calculated respectively. Then, **F1-score** is calculated using Eq. (8) and represented in Table 1. As seen from Table 1, the highest F1-score (0.93) is achieved in the YOLOv3-Tiny with *416 × 416* resized input image model. As it is known that a higher F1-score leads to more accurate classification results in the system.

TABLE I
THE RESULTS OF 5 MODELS IN THE DATASET (t = 0.25)

Model	Precision	Recall	F1-score	mAP (%)	Avg IoU (%)	Iteration
YOLOv3-Tiny 288x288	0.91	0.88	0.89	90.62	69.9	11000
YOLOv3-Tiny 352x352	**0.93**	0.84	0.88	90.93	**71.0**	12000
YOLOv3-Tiny 416x416	**0.95**	**0.91**	**0.93**	**95.22**	**74.6**	**10000**
YOLOv3-Tiny 480x480	0.90	0.90	**0.90**	92.29	67.1	11000
YOLOv3-Tiny 544x544	0.90	0.85	0.88	91.55	66.8	11000

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{8}$$

For the UAV using the YOLOv3-Tiny *416 × 416* models, the highest mean success value among the 5 models is 0.7456. YOLOv3-Tiny *416 × 416* is more robust than the other models. Besides, when comparing the methods, the OPE (One Pass Evaluation Protocol) was used because the precision scores should be taken into consideration for a fair comparison. The overlap between the position (R) produced by the method for the objects detected and the real position (ground-truth) (β) of the target objects manually determined by humans was taken into account for calculating success scores and the IoU (Intersection over Union) ratio was calculated by,

$$IoU = \frac{area|\beta \cap R|}{area|\beta \cup R|} \tag{9}$$

When IoU value is obtained less than 50% percentage, the position turned is defined as false (false positive - FP). In contrast, it is defined as true (true positive - TP) while if the method could not produce a position, it was counted as missing (false negative - FN). This process was repeated by the amount of the frames in the whole data set, and the ratio of the successful frames to the whole frame was found and achieved as a success score. Then, the precision-recall curves were used to determine the mean precision-mAP. The iteration values of the highest mAP scores are given in Table 1.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033481, IEEE Access

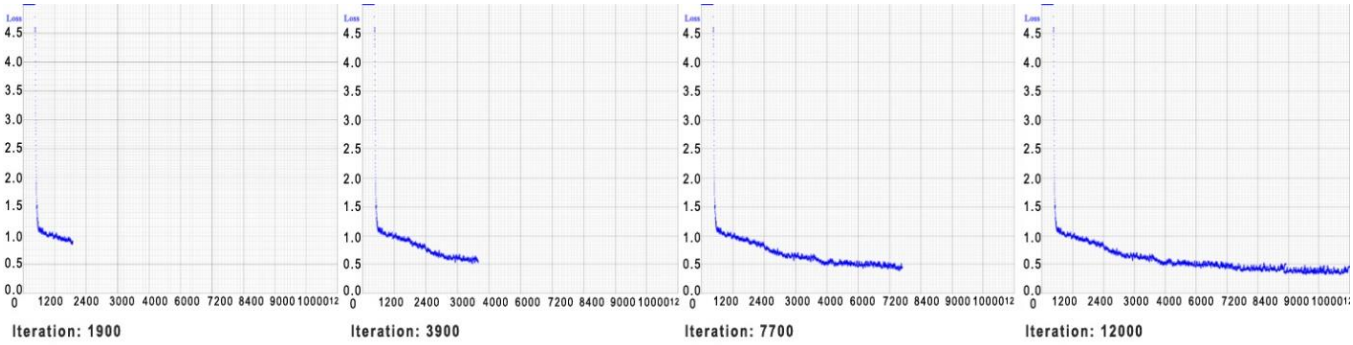

FIGURE 5. The success rate of the dataset created

Fig. 5. shows the success rate of the data set in different iterations. The performance of the proposed method is compared with the GOTURN, CRST, and KCF algorithms as shown in Fig. 6. The sample frames are selected randomly under different scale variations and lighting changes. As shown from the figure, all algorithms perform well under the controlled environment but the CRST and KCF algorithm fails in scale variations and cluttered background.

The proposed method was compared with KCF [9], GOTURN [26], CSRT [18], TLD [19], MIL [20], and BACF [21] on real-time live video, which was used in similar studies in the literature. In this comparison, the ATAUAV dataset was used for YOLOv3-Tiny as a training process. Since the UAVHDB [35] data set could not be reached, this data set could not be tested on the proposed method.

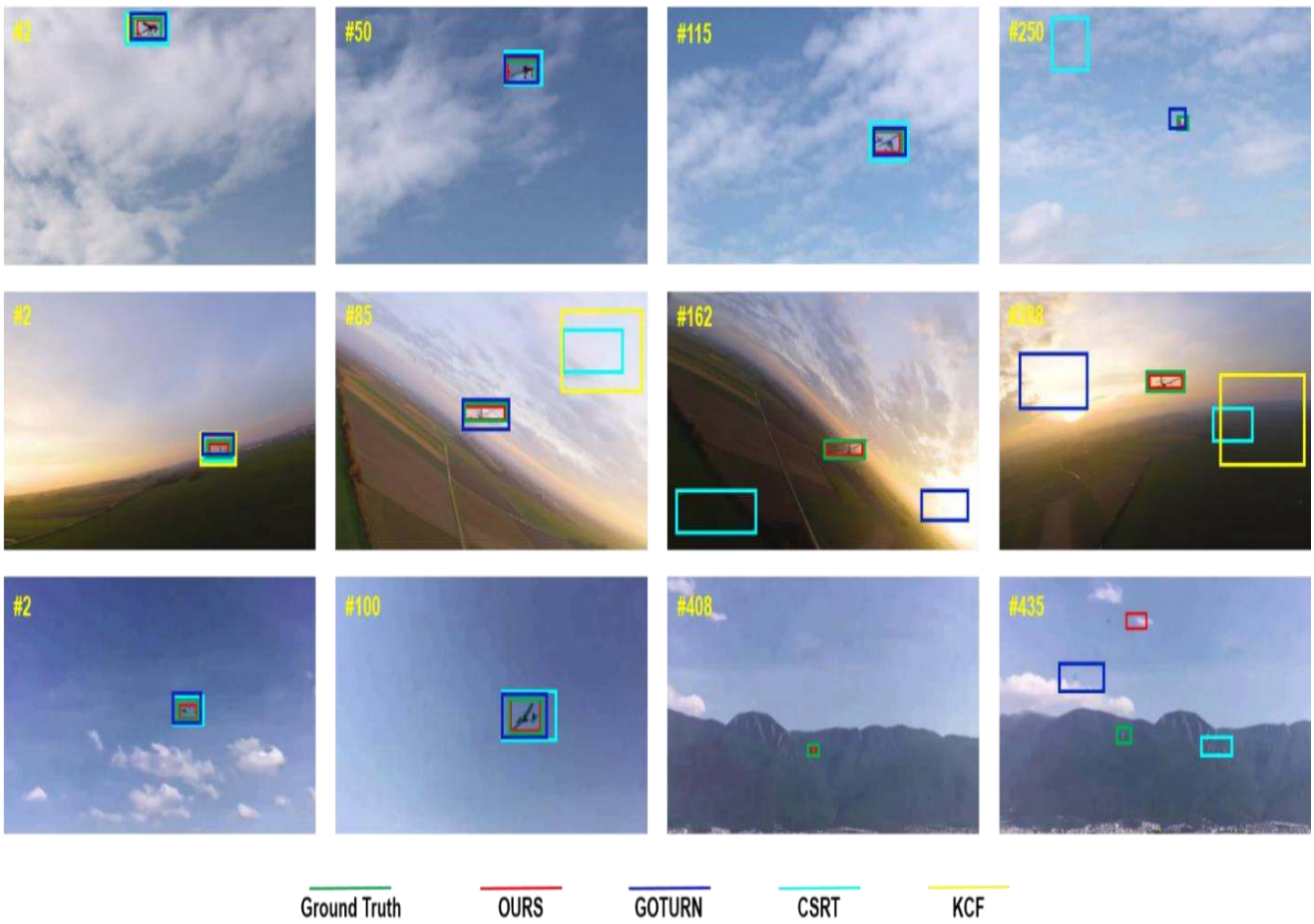

FIGURE 6. Performance evaluation of OURS and other trackers on ATAUAV dataset

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.3033481, IEEE Access

TABLE II
THE PERFORMANCE OF THE METHODS ON ATAUAV DATA SET AND HYBRID TRACKER DATA SET [35] (UAVHDB)

Method	Data Set	Hardware	Success	Precision	Avg. Speed (fps)
Ours	**ATAUAV**	**CPU**	**0.827**	0.827	29.6
YOLOv3-Tiny 416x416 [27]	**ATAUAV**	**CPU**	0.694	**0.910**	13.4
KCF [9]	**ATAUAV**	**CPU**	0.117	0.190	**76.4**
GOTURN [26]	**ATAUAV**	**CPU**	0.513	0.701	20
CSRT [18]	**ATAUAV**	**CPU**	0.244	0.482	29.3
TLD [19]	**ATAUAV**	**CPU**	0.196	0.253	16.2
MIL [20]	**ATAUAV**	**CPU**	0.207	0.364	9.3
BACF [21]	**ATAUAV**	**CPU**	0.203	0.321	21.6
Saribas et al. [35] UAVH	UAVHDB	GPU	0.561	0.773	53.5
Saribas et al. [35] UAVH-Tiny	UAVHDB	GPU	0.524	0.737	69.2
Saribas et al. [35] YOLOv3-Tiny	UAVHDB	GPU	0.461	0.630	47.1

The precision and success scores of the methods are shown in Table 2. Also, the performance of the methods tested on Raspberry Pi 4, 4 GB are shown in Table 3. It is observed from the table that the method we recommend combines the precision of YOLO with the high fps speed of KCF and so, it provides the highest success scores and can track targets on low-cost hardware.

The proposed method runs 16.2 fps faster than YOLOv3-Tiny running at 13.4 fps, and its overall success rate is the highest with 82.7% as seen in Table 2. Besides, it is understood that although the precision rate decreased by 8.3%, it has the highest rate compared to other studies. As can be seen in the tables, the method running on the CPU and Raspberry Pi in real-time has a high precision value and 29.6 and 19.7 fps speed respectively.

Fig. 7. shows an indicator screen from UAV and Fig. 8. depicts an example of visualization on our CPU-based ground control station of the proposed real-time method. The station allows tracking the position and attitude of the aircraft on a map displayed directly on the computer. It also allows the display of the main variables of the UAV, sent via a radio link. Besides, the system provides a user-friendly interface to instantly monitor some UAV parameters such as speed, altitude, battery status, and aerodynamic information.

TABLE III
THE PERFORMANCE OF THE METHODS ON ATAUAV DATA SET WITH Raspberry Pi 4, 4GB

Method	Data Set	Hardware	Avg. Speed (fps)
Ours	ATAUAV	Raspberry Pi 4	19.7
YOLOv3-Tiny 416x416 [27]	ATAUAV	Raspberry Pi 4	1
KCF [9]	ATAUAV	Raspberry Pi 4	46.1
GOTURN [26]	ATAUAV	Raspberry Pi 4	2
CSRT [18]	ATAUAV	Raspberry Pi 4	5.2
TLD [19]	ATAUAV	Raspberry Pi 4	3.4
MIL [20]	ATAUAV	Raspberry Pi 4	3.8
BACF [21]	ATAUAV	Raspberry Pi 4	4.3

FIGURE 7. Our GCS indicator screen

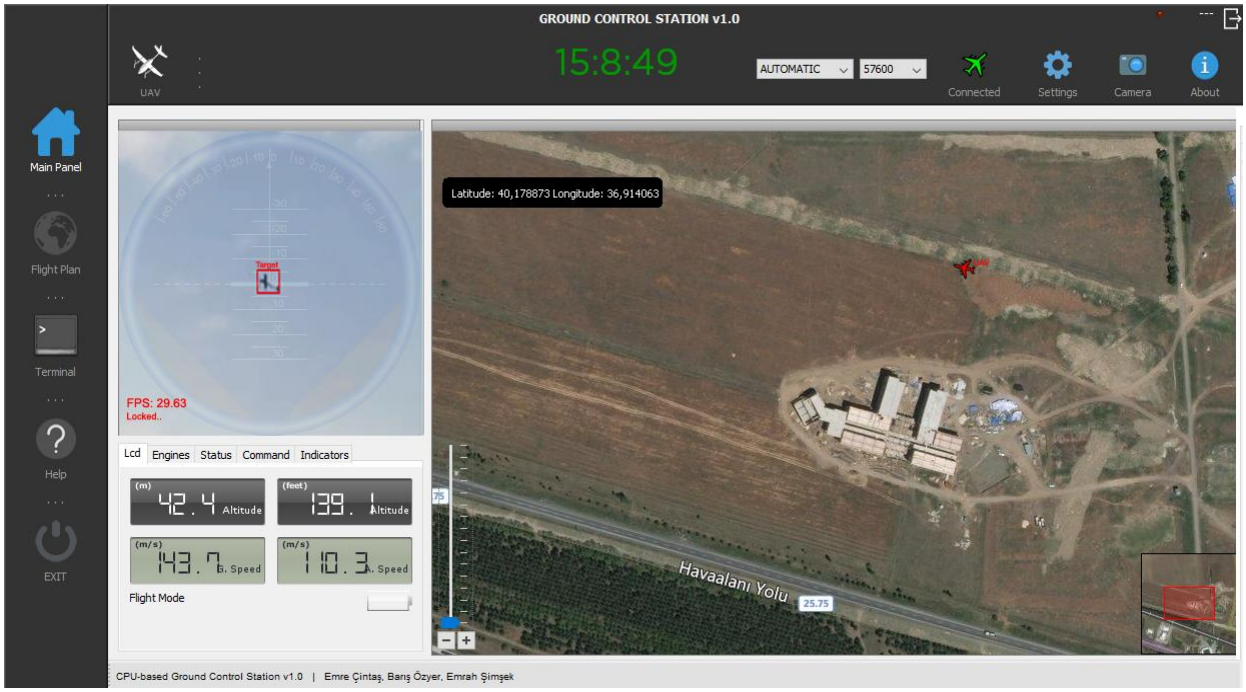

FIGURE 8. Recommended method success on our ground control station

IV. CONCLUSION

In this study, we design a low-cost tracking system without a graphic card. YOLOv3-Tiny is used to detect and recognize the UAV that improves success, and precision scores. KCF algorithm is used to track the detected object that improves the fps speed. We observed that the proposed approach provides the highest accuracy rate as 82.7% and a mean fps speed as 29.6 on CPU. A new benchmark ATAUAV dataset is created to be used in scientific studies. The ground control station is developed to control UAV and monitories the performance of the algorithms.

ACKNOWLEDGMENT

The work in this paper is supported by the Atatürk University Scientific Research Projects Coordination Unit (BAP) under the project FBA-2020-8447.

REFERENCES

[1] M. Zhu, H. Zhang, J. Zhang and L. Zhuo, "Multi-level prediction Siamese network for real-time UAV visual tracking," Image and Vision Computing, 2020, doi: 10.1016/j.imavis.2020.104002.

[2] C. Fu, R. Duan and D. Kircali, "Onboard Robust Visual Tracking for UAVs Using a Reliable Global-Local Object Model," Sensors, 2016, 16:1406, doi: 10.3390/s16091406.

[3] N. A. Khan, N. Z. Jhanjhi, S. N. Brohi, R. S. A. Usmani and A. Nayyar, "Smart traffic monitoring system using Unmanned Aerial Vehicles (UAVs)," Computer Communications, 2020, pp. 434-443, doi: 10.1016/j.comcom.2020.04.049.

[4] H. Lim, S. N. Sinha, "Monocular localization of a moving person onboard a Quadrotor MAV," Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 2015, pp. 2182–2189, doi: 10.1109/ICRA.2015.7139487.

[5] C. Fu, A. Carrio, M. A. Olivares-Mendez, R. Suarez-Fernandez and P. Campoy, "Robust real-time vision-based aircraft tracking from Unmanned Aerial Vehicles," 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, 2014, pp. 5441-5446, doi: 10.1109/ICRA.2014.6907659.

[6] X. Xue, Y. Li and Q. Shen, "Unmanned Aerial Vehicle Object Tracking by Correlation Filter with Adaptive Appearance Model," Sensors (Basel), 2018, 18(9):2751, doi: 10.3390%2Fs18092751.

[7] B. Bai, B. Zhong, G. Ouyang, P. Wang, P. Liu, Z. Chen, and C. Wang, "Kernel correlation filters for visual tracking with adaptive fusion of heterogeneous cues," Neurocomputing, vol. 286, 2018, pp. 109–120, doi: 10.1016/j.neucom.2018.01.068.

[8] M. Danelljan, A. Robinson, F.S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," European Conference on Computer Vision. Springer, 2016, pp. 472–488, doi: 10.1007/978-3-319-46454-1_29.

[9] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 37, no. 3, 2015, pp. 583–596, doi: 10.1109/TPAMI.2014.2345390.

[10] P. J. Davis, "Circulant matrices," American Mathematical Soc, 2012.

[11] N. Dalal, and B. Triggs, "Histograms of oriented gradients for human detection," IEEE Conference on Computer Vision and Pattern Recognition, 2006, doi: 10.1109/CVPR.2005.177.

[12] M. Awais et al., "Real-Time Surveillance Through Face Recognition Using HOG and Feedforward Neural Networks," in IEEE Access, vol. 7, pp. 121236-121244, 2019, doi: 10.1109/ACCESS.2019.2937810.

[13] T. Ahonen, A. Hadid, and M. Pietikainen, "Face description with local binary patterns," IEEE Tran. On PAMI, 28, 2006, pp. 2037-2041, doi: 10.1109/TPAMI.2006.244.

[14] A. S. Bozkir, and M. Aydos, "Local Image Descriptor Based Phishing Web Page Recognition as an Open-Set Problem," European Journal of Science and Technology, 2019, pp. 444-451, doi: 10.31590/ejosat.638404.

[15] Z. Huang, C. Fu, Y. Li, F. Lin and P. Lu, "Learning Aberrance Repressed Correlation Filters for Real-Time UAV Tracking," 2019 IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 2891-2900, doi: 10.1109/ICCV.2019.00298.

[16] Y. Li, C. Fu, F. Ding, Z. Huang and G. Lu, "Autotrack: Towards high-performance visual tracking for UAV with automatic spatio-temporal regularization," CoRR, 2020.

[17] J. Yan, J. Du, Y. Young, C. R. Chatwin and P. Birch, "Real-time unmanned aerial vehicle tracking of fast moving small target on

ground," Journal of Electronic Imaging, 2018, doi: 10.1117/1.JEI.27.5.053010.

[18] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," In Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6309–6318, doi: 10.1007/s11263-017-1061-3.

[19] Z. Kalal, K. Mikolajczyk, J. Matas, et al, "Tracking-learning detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2012, 34(7):1409, doi: 10.1109/TPAMI.2011.239.

[20] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," In Proc. IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 983–990, doi: 10.1109/CVPR.2009.5206737.

[21] H. Kiani Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," In Proc. IEEE International Conference on Computer Vision, 2017, pp. 1135–1143.

[22] Y. Zhang, Y. Shen and Q. Zhao, "Multi-Person tracking algorithm based on data association," Optik, 2019, doi: 10.1016/j.ijleo.2019.163124.

[23] A. Rohan, M. Rabah and S-H. Kim, "Convolutional Neural Network-Based Real-Time Object Detection and Tracking for Parrot AR Drone 2", Access IEEE, vol. 7, pp. 69575-69584, 2019, doi: 10.1109/ACCESS.2019.2919332.

[24] Y. Li, H. Dong, H. Li, X. Zhang, B. Zhang, and Z. Xiao, "Multi-block SSD based on small object detection for UAV railway scene surveillance," Chinese Journal of Aeronautics, 2020, doi: 10.1016/j.cja.2020.02.024.

[25] A. Rivas, P. Chamoso, A. González-Briones and J. M. Corchado, "Detection of cattle using drones and convolutional neural networks," Sensors, vol. 18, no. 7, p. 2048, 2018, doi: 10.3390/s18072048.

[26] D. Held, S. Thrun and S. Savarese, "Learning to Track at 100 FPS with Deep Regression Networks," European Conference on Computer Vision, 2016, doi: 10.1007/978-3-319-46448-0_45.

[27] J. Redmon, and A. Farhadi, "Yolov3: An incremental improvement," 2018, arXiv preprint arXiv:1804.02767.

[28] E. Çintaş, B. Özyer and Y. S. Hanay, "Ontology-based instantaneous route suggestion of enemy warplanes with unknown mission profile," Sakarya University Journal of Science, pp. 803-818, 2020, doi: 10.16984/saufenbilder.711109.

[29] E. Şimşek, B. Özyer, and G. T. Özyer "Foto-Kapan Görüntülerinde Derin Öğrenme Tabanlı İnsan Tespiti," Bayburt Üniversitesi Fen Bilimleri Dergisi, 3(1), pp. 1-8.

[30] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, and T. Chen, "Recent advances in convolutional neural networks," Pattern Recognition, 77, 2018, pp. 354-377, doi: 10.1016/j.patcog.2017.10.013.

[31] D. Perez, I. Maza, F. Caballero, et al. "A Ground Control Station for a Multi-UAV Surveillance System," J Intell Robot Syst 69, 2013, pp. 119–130, doi: 10.1007/s10846-012-9759-5.

[32] J. Li, Y. Zhou and L. Lamont, "Communication Architectures and Protocols for Networking Unmanned Aerial Vehicles," Globecom Workshop - Wireless Networking and Control for Unmanned Autonomous Vehicles, 2013.

[33] N. Prapulla, S. Veena, and G. Srinivasalu, "Development of Algorithms for MAV Security," IEEE International Conference on Recent Trends in Electronics Information Communication Technology, India, 2016, pp. 799-802, doi: 10.1109/RTEICT.2016.7807936.

[34] S. Atoev, K. Kwon, S. Lee and K. Moon, "Data analysis of the MAVLink communication protocol," 2017 International Conference on Information Science and Communications Technologies (ICISCT), Tashkent, 2017, pp. 1-3, doi: 10.1109/ICISCT.2017.8188563.

[35] H. Saribas, B. Uzun, B. Benligiray, O. Eker and H. Cevikalp, "A Hybrid Method for Tracking of Objects by UAVs," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Long Beach, CA, USA, 2019, pp. 563-572, doi: 10.1109/CVPRW.2019.00082.

EMRE ÇİNTAŞ received his B.S and M.S degree all in Computer Engineering Department from İstanbul Aydın University and Atatürk University, Turkey respectively. His research interests include artificial neural networks, computer vision, object tracking, unmanned aerial vehicles, aircraft engineering, software engineering, semantic knowledge-based systems, ontology and machine learning algorithms. He is currently pursuing the Ph.D. degree in Computer Engineering with Atatürk University and working as a lecturer at Erzurum Technical University, Erzurum, Turkey.

BARIŞ ÖZYER is an Assistant Professor at the Department of Computer Engineering at Atatürk University, Erzurum, Turkey. He received B.Sc. degree from Erciyes University in 2002 and integrated Ph.D. degree from Middle East Technical University in 2012, all in Electrical and Electronics Engineering Department. He worked at CNS Computational Neuroscience Laboratories, ATR, Japan as an intern researcher between 2007 and 2008. He is interested in robotics, artificial neural networks, machine learning applications, computer vision and semantic knowledge-based systems.

EMRAH ŞİMŞEK received his B.S and M.S degree all in Computer Engineering Department from Atatürk University, Erzurum, Turkey. His research interests include machine learning algorithms, artificial neural networks, computer vision and object tracking. He is currently pursuing the Ph.D. degree in Computer Engineering with Atatürk University and working as a lecturer at Erzurum Technical University, Erzurum, Turkey.