IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 28 August 2023, accepted 21 September 2023, date of publication 25 September 2023,
date of current version 28 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3318745

RESEARCH ARTICLE

Computationally Efficient Ground-to-Air Missile Seeker Based on Camera Images

JONGHOEK KIM, (Member, IEEE)
System Engineering Department, Sejong University, Seoul 05006, South Korea
e-mail: jonghoek@gmail.com

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (Grant Number: 2022R1A2C1091682). This research was supported by the faculty research fund of Sejong university in 2023.

ABSTRACT Recently, advanced ground-to-air missiles are developed to defend against a targeted aircraft. A conventional ground-to-air missile has infrared (IR) seekers or radar sensors. However, an anti-aircraft missile based on infrared images is easily distracted by flares, thus an aircraft deploys flares when an incoming missile is detected. In this paper, we propose a missile seeker that complements the weakness of the existing IR sensors, by switching to camera sensors in the case where IR sensors are disturbed. As we switch to camera sensors, we generate the initial bounding box by applying simple line detection algorithms on the target image. Once a bounding box is generated at the target aircraft, the box is tracked continuously using object trackers, which are based on camera images. The missile then uses the bounding box for tracking the target continuously. The object tracker is based on the camera image and is not disturbed by flares or chaffs. To the best of our knowledge, our paper is unique in proposing a missile seeker that complements the weakness of the existing IR sensors, by switching to camera sensors in the case where IR sensors are disturbed. The proposed tracking algorithm is simple and computationally efficient, thus is suitable for real time applications. As far as we know, our tracking algorithm is novel in using simple line detection algorithms for tracking a target aircraft. Using experiments, we show that the proposed tracking method outperforms the-state-of-the-art on object detector considering both time efficiency and tracking accuracy.

INDEX TERMS Military aircraft, infrared image, flare, channel and spatial reliability tracking filter, camera image, chaffs, computationally efficient target tracking, ground-to-air missile.

I. INTRODUCTION

Nowadays, advanced ground-to-air missiles have been invented to defend against an enemy aircraft. A conventional ground-to-air missile has infrared (IR) seekers. The missile chases a targeted aircraft based on infrared images. Infrared homing uses an infrared sensor to detect infrared rays, i.e., heat, emitted from the targeted aircraft, and track the target.

A targeted aircraft, such as fighter jets, deploys flares, in order to defend against a missile approaching the aircraft. Flares are used to distract the on-board IR sensors of the missile. The flare emits high-temperature heat and emits infrared light similar to that of an aircraft. Furthermore, flares blur the infrared image, in order to distract the IR sensor measurements. The aircraft drops flares when it confirms that a missile has been launched using the missile approach warning (MAW). Fig. 1 shows a missile that hits the flare, instead of the target aircraft.

IR sensors have been widely used for tracking objects, such as an aircraft [1]. However, in contrast to visual images, infrared images generally have low spatial resolution, poor signal-to-noise ratios (SNR), and lack of textural information [1], [2]. In [2], an aircraft-tracking algorithm based on regional distribution in the case of interference was presented. For handing frequent occlusion caused by decoys, [2] addressed a mechanism of occlusion detection for ensuring the accuracy of the target model. The authors of [3] proposed an automatic target recognition architecture suitable for anti-ship missiles with IR capabilities. Reference [4] used convolutional neural network (CNN) to extract IR features

The associate editor coordinating the review of this manuscript and approving it for publication was Tariq Umer.

FIGURE 1. A missile hits the flare, instead of the target aircraft.

of an aircraft and achieve target detection under interference environment. Recently, avionics devices such as the Directed infrared countermeasures system (DIRCM) is used, and a smoke-type flare is launched for disabling the IR sensor on the missile [5]. Thus, relying on IR sensor only can disable the missile's ability for tracking an aircraft.

There are many papers on object detection using camera measurements. Due to the advance in artificial intelligence, object detect-classification algorithms based on CNN can be utilized for processing camera image measurements. Fast R-CNN [6] classified object proposals using deep CNN. Fast R-CNN employed several innovations to improve training and testing speed, while increasing detection accuracy. You only look once (YOLO) [7], [8] outperforms Fast R-CNN, by framing object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. YOLOv4 [9] and YOLOv4-tiny [10] were recently developed as variants of YOLO models. However, computational load of CNN-based algorithms is rather heavy, thus CNN-based algorithms may not be suitable for tracking a fast moving missile, which needs to process the image measurements in real time.

Our tracking method is used to support IR sensors, only in the case where flairs are deployed. In the case where flairs are not deployed, IR sensors are used until hitting the target. IR sensors are used for tracking the target before the target deploys flairs.

Once flairs are deployed, IR sensors detect light sources that are dispersed widely. At this moment, the target image can be clearly measured by cameras, since flairs provide light sources even in dark environments. Hence, our camera-based tracking approach can be used even in dark environments. To further enhance low light images in dark environments, we can apply neural network methods, such as [11], [12], and [13].

At the moment when flairs are deployed, we switch to camera sensors. Then, we generate the initial bounding box by applying simple line detection algorithms on the image of the target aircraft. Once a bounding box is generated at the target aircraft, the box is tracked continuously using object trackers, such as the channel and spatial reliability tracking (CSRT) filters [14], [15], which are based on camera images. We acknowledge that other tracking filters [16], [17], [18], [19] can be used for tracking a bounding box. We use the CSRT filter, since it runs fast while showing good tracking performance in our experiments. This CSRT filter uses spatial reliability maps for adjusting the filter support to the part of the selected region from the frame for tracking, which gives an ability to increase the search area and track arbitrary objects.

The missile then uses the bounding box of the CSRT filter for tracking the target continuously. Note that the CSRT filter is based on the camera image and is not disturbed by flares or chaffs.

To the best of our knowledge, our article is unique in proposing a missile seeker that complements the weakness of the existing IR sensors, by switching to camera sensors in the case where IR sensors are disturbed. Since a missile moves fast, time efficiency for target tracking is an important factor. The proposed tracking algorithm is simple and computationally efficient, thus is suitable for real time applications. As far as we know, our tracking algorithm is novel in using simple line detection algorithms, in order to track a target aircraft in a time-efficient manner.

The performance of the proposed tracking method is verified by comparing it with the-state-of-the-art object detectors (YOLOv4 [9] and YOLOv4-tiny [10]) using experiments. Using experiments, we show that the proposed tracking method outperforms the-state-of-the-art object detectors considering both time efficiency and tracking accuracy.

The remainder of this paper is organized as follows. Section II presents the background information of this paper. Section III presents the proposed aircraft tracker using camera images. Section IV addresses the experiment results of this paper. Section V presents the conclusions of this paper.

II. BACKGROUND INFORMATION

The proposed camera seeker complements the existing IR tracker. How to determine the location of the camera is not within the scope of our paper. The camera can be installed at any place on the missile, as long as the camera can detect a target in front of the missile. The proposed camera image processing applies to every image frame in the video. Hence, the rolling of the missile does not disturb the image processing.

In the case the aircraft has not deployed a flair, existing IR tracker can track the target aircraft. At the moment when the target deploys a flair, we switch to camera sensors. Then one generates the initial bounding box by applying simple edge detectors on the image of the target aircraft.

We address how to generate the initial bounding box based on the camera image. We assume that the target aircraft is far from the missile initially. Also, assume that the target image exists in the sky, not in mountains or buildings. These assumptions are feasible considering ground-to-air missiles.

An aircraft image has a straight line segment, such as an aircraft's wing. We thus use line segments of the target aircraft for generating the initial bounding box. We apply the canny edge detector [20] to generate the edge image from the image of the camera seeker. Then, the probabilistic hough line transform [21] detects straight line segments in the edge image. This transform detects long straight line segments in the image. Detected line segments are used to set a target point, and we generate a bounding box centered at the point. Once a bounding box is generated at the target, the box is tracked robustly using the CSRT filter [14], [15].

In practice, there may be a case where the image of the camera seeker cannot detect a line segment. For instance, a stealthy aircraft may exist behind the cloud, so that it is not detected by neither the camera seeker nor the IR sensor. In this case, we skip the frame which does not contain a line segment. We iteratively skip frames until we find a frame which contains a line segment. Once we detect a frame with a line segment, detected line segments are used to set a target point, and we generate a bounding box centered at the point. Once a bounding box is generated at the target aircraft, the box is tracked robustly using the CSRT filter [14], [15].

III. THE MISSILE TRACKER USING CAMERA IMAGES

A. GENERATE A BOUNDING BOX AT THE TARGET

For detecting the target aircraft's body in the camera image, we first apply the canny edge detector to the image. Then, the probabilistic hough line transform is applied to find straight line segments in the edge image.

The purpose of the probabilistic hough line transform is to find straight line segments in the edge image by a voting procedure. The efficiency of the probabilistic hough line transform is dependent on the quality of the input data. It is desirable to apply a denoising stage before we apply the probabilistic hough line transform on a noisy image. In the case where the image is corrupted by speckle, as is the case in radar images, the Radon transform [22] is sometimes preferred to detect lines, because it attenuates the noise through summation.

Using the probabilistic hough line transform, we find a set of edges in the image. Suppose that we find N edges in the image. Let $e_i = \{p_i, q_i\}$ where $i \in \{1, 2, \ldots, N\}$ denote the i-th edge detected by the probabilistic hough line transform. In the image plane, $p_i \in \mathcal{R}^2$ and $q_i \in \mathcal{R}^2$ denote the coordinates of two end points of e_i. The coordinates of the center of an edge e_i is $\frac{p_i+q_i}{2} \in \mathcal{R}^2$.

The center of end points of all line segments is set as

$$t = \sum_{i=1}^{N} \frac{p_i + q_i}{2N} \tag{1}$$

For instance, Fig. 2 shows a target aircraft. This figure is an image from Video 1 in experiments. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. See that multiple line segments are found, since the aircraft is close to the camera.

FIGURE 2. Considering Video 1 in experiments, this figure shows a target aircraft. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. A bounding box (blue box) is generated at the aircraft.

In practice, outlier line segments can be generated on clutter, such as cloud, but not on the aircraft. To reduce the effect of outlier line segments, we detect line segments which are too far from $\mathbf{t}$ in (1).

K-means clustering [23] can be used to detect outlier line segments which are too far from $\mathbf{t}$. We set $K = 2$ in K-means clustering. This implies that the distance from $\mathbf{t}$ is used to distinguish outlier line segments from other line segments. Suppose that we have $M \leq N$ remaining line segments, other than the outlier line segments. This implies that $N - M$ line segments and M line segments form two clusters, as a result of applying K-means clustering.

Let $E_i = \{P_i, Q_i\}$ where $i \in \{1, 2, \ldots, M\}$ denote the i-th edge in M remaining line segments. The center of end points of all M line segments is set as the *target point*, which is derived as

$$T = \sum_{i=1}^{M} \frac{P_i + Q_i}{2M} \tag{2}$$

We then generate a bounding box centered at the point $T \in \mathcal{R}^2$.

The bounding box has the following size:

$$\begin{aligned} width &= max_{i\in\{1,2,\ldots,M\}}(max(P_i[1], Q_i[1])) \\ &\quad - min_{i\in\{1,2,\ldots,M\}}(min(P_i[1], Q_i[1])), \\ height &= max_{i\in\{1,2,\ldots,M\}}(max(P_i[2], Q_i[2])) \\ &\quad - min_{i\in\{1,2,\ldots,M\}}(min(P_i[2], Q_i[2])). \end{aligned} \tag{3}$$

Here, considering an arbitrary set s, $\max(s)$ denotes the maximum value in the set s. Also, $\min(s)$ denotes the minimum value in the set s. For an arbitrary vector $\mathbf{v}$, $v[j]$ denote the j-th element in $\mathbf{v}$.

Let δ denote the minimum side length (in pixels) of the bounding box. Here, $\delta > 0$ is a tuning parameter. In our experiments, we use $\delta = 150$ pixels. In the case where *width*

FIGURE 3. Considering Video 2 in experiments, a bounding box is generated at a target aircraft which is far from the camera. The straight line segments detected are depicted with green line segments.

in (3) is less than δ, we set $width = \delta$. In the case where $height$ in (3) is less than δ, we set $height = \delta$.

B. TRACKING THE BOUNDING BOX USING OBJECT TRACKERS

Once a bounding box is formed at the aircraft, the box is tracked using object trackers, such as the CSRT filter [14], [15]. For instance, Fig. 2 plots a bounding box (blue box), which is generated at the aircraft.

As another example, Fig. 3 shows the case where a bounding box is generated at a target aircraft. This figure is an image from Video 2 in experiments. The blue rectangle presents the generated bounding box. The straight line segments detected using both the canny edge detector and the probabilistic hough line transform are depicted with green line segments.

While the CSRT filter runs, its bounding box may lose tracking the target. For handling the case where the filter loses track of the target, we re-generate a bounding box at the target in the case where the box side length is bigger than a certain threshold, say Th. We use $Th = 300$ pixels. For re-generation of a bounding box, we use the approach in Section III-A. The effectiveness of this re-generation of a bounding box is verified using experiments in Section IV.

C. THE EFFECT OF USING THE CANNY EDGE DETECTOR

For detecting an aircraft's body in the first image of the camera seeker, we first apply the canny edge detector to the first image of the camera seeker. This subsection presents the effect of using the canny edge detector.

Fig. 4 shows the edge image under the canny edge detector. Considering Video 1 in experiments, Fig. 4 shows the edge image of the aircraft in Fig. 2. We then apply the probabilistic hough line transform to the edge image. This transform detects long straight line segments in the edge image, as plotted in Fig. 2. K-means clustering in Section III-A is further used to cancel outlier line segments, which exists at the upper part of Fig. 4. Detected line segments are used to set a target point, and we generate a bounding box centered at the point. Once a bounding box is generated at the aircraft, the box is tracked robustly using the CSRT filter [14], [15].

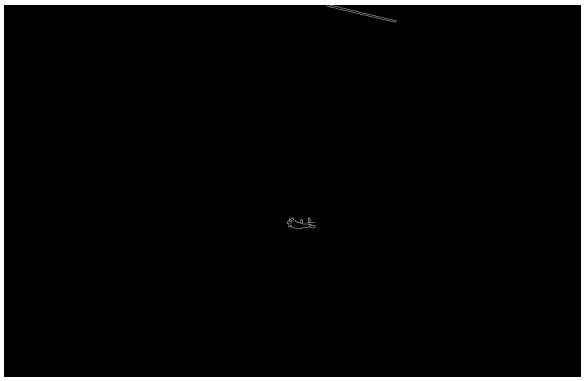


FIGURE 4. Considering Video 1 in experiments, this figure shows the edge image of the aircraft in Fig. 2. For detecting an aircraft's body in the first image of the camera seeker, we first apply the canny edge detector to the first image of the seeker. This plot shows the edge image derived under the canny edge detector. K-means clustering in Section III-A is further used to cancel outlier line segments, which exists at the upper part of this figure.

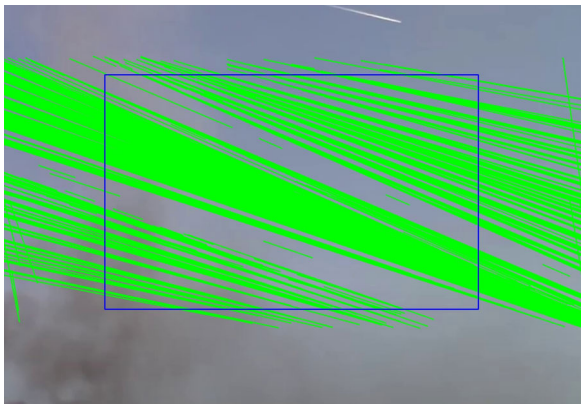


FIGURE 5. Considering Video 1 in experiments, this figure shows the case where we apply the probabilistic hough line transform to the first image of the camera seeker, not to the edge image. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. Compared to Fig. 2, many false line segments are generated outside the aircraft. This shows the effect of applying the canny edge detector before applying the probabilistic hough line transform.

Next, Fig. 5 presents the case where we apply the probabilistic hough line transform to the first image of the camera seeker, not to the edge image. In other words, Fig. 5 presents the case where the canny edge detector is not used. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. Compared to Fig. 2, many false line segments are generated outside the aircraft in Fig. 5. This shows the effect of applying the canny edge detector before applying the probabilistic hough line transform.

IV. EXPERIMENTS

Using both the canny edge detector and the probabilistic hough line transform, we find a set of edges in the image. In the canny edge detector, several parameters must be set. If a pixel intensity gradient is higher than the upper threshold 100, the pixel is accepted as an edge. If a pixel intensity gradient

FIGURE 6. This plot shows the examples of a bounding box generated at various target aircrafts (Section III-A). The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. See that our bounding box is generated on various kinds of aircrafts, including helicopters.

FIGURE 7. Considering Video 3 in experiments, this plot shows the bounding box formed at the target aircraft. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments.

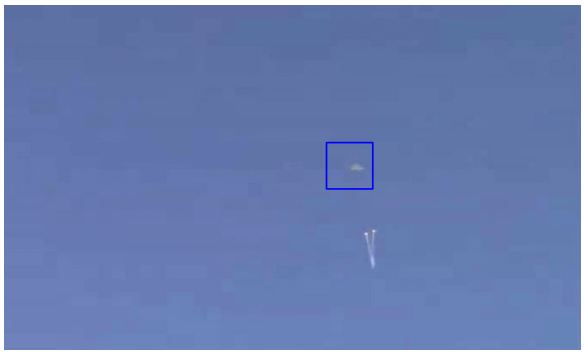


FIGURE 8. Considering Video 3 in experiments, a bounding box is maintained to track the far target aircraft which deploys flares.

value is below the lower threshold 50, then it is rejected. If the pixel intensity gradient is between the two thresholds, then it will be accepted only if it is connected to a pixel that is above the upper threshold.

In the probabilistic hough line transform [21], several parameters must be set. In the probabilistic hough line transform, ρ presents the resolution of the parameter ρ in pixels. We use 1 pixel. In addition, θ presents the resolution of the parameter θ in radians. We use 1 degree. Let *threshold* present the minimum number of intersections to detect a line. We use 10 for *threshold*. Let *minLineLength* denote the minimum number of points that can form a line. Lines with less than this number of points are disregarded. We use 20 for *minLineLength*. Let *maxLineGap* denote the maximum gap between two points to be considered in the same line. We use 10 for *maxLineGap*.

We use various aircrafts for testing the generation of a bounding box in Section III-A. Fig. 6 shows the examples of a bounding box generated at various target aircraft. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. See that our bounding box is generated on various kinds of aircrafts, including helicopters.

For verification of our tracking system, the hardware specifications are limited to low-power CPUs for laptops without external GPUs. Our system was tested with various video files. Each video shows a scene of a flying aircraft in $1052 * 502$ resolution. Our videos are available upon the request of readers.

Video 1 considers a target aircraft as plotted in Fig. 2. In Video 1, the aircraft deploys flares while it flies. Video 2 considers a far target aircraft as plotted in Fig. 3. In Video 2, the aircraft does not deploy flares while it flies. Video 5 is a blurred video of Video 2, by applying blurring to every image in Video 2. Considering Video 3 in experiments, Fig. 7 shows the bounding box generated at the target aircraft. The straight line segments detected using the probabilistic hough line transform are depicted with green line segments. See Section III-A for generation of a bounding box at the target.

Considering Video 3, Fig. 8 depicts the video image where a bounding box is maintained to track the far target aircraft which deploys flares. See that the bounding box is not disturbed by flares. We use the tracking method in Section III-B, for tracking the target while not losing the target.

A. COMPARISON WITH THE-STATE-OF-THE-ART OBJECT DETECTION METHODS

We evaluate the proposed tracker by the following metrics, (i) success rate (*SuccessRate*), which shows the percentage of frames where the tracking succeeds (generated bounding box contains the tracked aircraft); (ii) FPS (*FPS*), which shows how many frames are processed within a second. Thus, a tracker with large FPS runs fast and is suitable for real-time applications.

We compare the proposed detection system with the-state-of-the-art object detectors (YOLOv4 [9] and YOLOv4 tiny algorithms [10]). These Yolo-based algorithms use CNN transfer learning in [10]. The algorithm's weighted model is pre-trained with COCO dataset [24]. Note that the proposed tracker based on line detection does not require the learning process, which is required for CNN-based

TABLE 1. Comparison results.

Video (total frame number)	*Alg.*	*FPS*	*SuccessRate*
Video 1 (482)	$[Pro]$	55	100
Video 1 (482)	$[Pro_c]$	48	100
Video 1 (482)	$[Y]$	4	65
Video 1 (482)	$[T]$	50	27
Video 2 (1200)	$[Pro]$	47	99
Video 2 (1200)	$[Pro_c]$	37	90
Video 2 (1200)	$[Y]$	5	98
Video 2 (1200)	$[T]$	44	50
Video 3 (413)	$[Pro]$	62	100
Video 3 (413)	$[Pro_c]$	64	100
Video 3 (413)	$[Y]$	8	34
Video 3 (413)	$[T]$	50	20
Video 4 (292)	$[Pro]$	45	99
Video 4 (292)	$[Pro_c]$	49	100
Video 4 (292)	$[Y]$	5	11
Video 4 (292)	$[T]$	50	1
Video 5 (1200)	$[Pro]$	51	99
Video 5 (1200)	$[Pro_c]$	43	85
Video 5 (1200)	$[Y]$	3	95
Video 5 (1200)	$[T]$	31	40

detection algorithms, such as YOLOv4 [9] and YOLOv4 tiny algorithms [10].

Using Yolo-based algorithms, the aircraft was misclassified as birds or kites occasionally. We consider this misclassification as success in *SuccessRate*, since generated bounding box still contains the tracked aircraft. In other words, classification error of the aircraft was not considered, as long as the bounding box contains the tracked aircraft.

In Table 1, we use four video clips for comparison among filters. In Table 1, $[Y]$ denotes the YOLOv4 [9], and $[T]$ denotes the YOLOv4 tiny algorithm [10]. In addition, $[Pro]$ denotes the proposed tracker.

While the CSRT filter runs, its bounding box may lose tracking the target. For handling the case where the filter loses track of the target, we re-generate a bounding box at the target in the case where the box side length is bigger than Th. We use $Th = 300$ pixels. Let $[Pro_c]$ denote the proposed tracker, in the case where the proposed bounding box re-generation in Section III-B is not used.

See that the proposed tracker outperforms other algorithms considering both *FPS* and *SuccessRate*. Table 1 shows that flares disturb the image detection efficiency *SuccessRate* in other state-of-the-art tracking methods ($[Y]$ and $[T]$). However, flares do not disturb *SuccessRate* in the proposed tracker $[Pro]$.

Among all video clips, only Video 2 considers the case where the target does not generate flares. In Video 2, $[Y]$ provides the success rate of 98 percents. However, its computational load is much heavier than the proposed tracking filter. On the other hand, $[T]$ runs fast, while its accuracy is not satisfactory.

Table 1 shows that $[Pro]$ outperforms $[Pro_c]$ considering the success rate. This verifies that the re-generation of a bounding box (Section III-B) is effective for tracking the target without losing the target.

Video 5 is a blurred video of Video 2. Thus, $[Pro_c]$ of Video 5 shows worse *SuccessRate* compared to $[Pro_c]$ of Video 2. However, $[Pro]$ of Video 5 shows equal *SuccessRate* compared to $[Pro]$ of Video 2. This shows the effectiveness of using the proposed bounding box re-generation approach in Section III-B.

V. CONCLUSION

A conventional ground-to-air missile chases an enemy aircraft based on infrared images. This article addresses a missile seeker that complements the defects of existing IR sensors by switching to camera sensors if necessary. If IR sensors are disturbed by flairs, then one generates the initial bounding box by applying simple edge detectors on the image of the target aircraft. Once a bounding box is generated at the aircraft, the box is tracked continuously using the CSRT filter, which is based on camera images. For handling the case where the filter loses track of the target, we re-generate a bounding box at the target in the case where the box side length is bigger than a certain threshold.

The proposed tracking algorithm is computationally efficient, thus is applicable for real time tracking scenarios. To the best of our knowledge, our tracking algorithm is unique in applying simple line detection algorithms for tracking a target aircraft in a time-efficient manner. The performance of the proposed tracking method is demonstrated by comparing with the-state-of-the-art tracking algorithms using experiments. Using experiments, we show that the proposed tracking method outperforms the-state-of-the-art object detectors considering both time efficiency and tracking accuracy.

REFERENCES

[1] E. Gundogdu, H. Ozkan, H. S. Demir, H. Ergezer, E. Akagündüz, and S. K. Pakin, "Comparison of infrared and visible imagery for object tracking: Toward trackers with superior IR performance," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2015, pp. 1–9.

[2] S. Wu, K. Zhang, S. Niu, and J. Yan, "Anti-interference aircraft-tracking method in infrared imagery," *Sensors*, vol. 19, no. 6, p. 1289, Mar. 2019.

[3] O. Kechagias-Stamatis, N. Aouf, and D. Nam, "Multi-modal automatic target recognition for anti-ship missiles with imaging infrared capabilities," in *Proc. Sensor Signal Process. Defence Conf. (SSPD)*, Dec. 2017, pp. 1–5.

[4] W. Qiu, K. Wang, S. Li, and K. Zhang, "YOLO-based detection technology for aerial infrared targets," in *Proc. IEEE 9th Annu. Int. Conf. CYBER Technol. Autom., Control, Intell. Syst. (CYBER)*, Jul. 2019, pp. 1115–1119.

[5] M. Petersson, "Real-time DIRCM system modeling," *Proc. SPIE*, vol. 5615, pp. 149–160, Dec. 2004.

[6] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 1440–1448.

[7] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

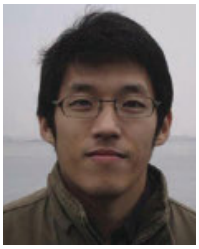
[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[9] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[10] Z. Jiang, L. Zhao, S. Li, and Y. Jia, "Real-time object detection method based on improved YOLOv4-tiny," 2020, *arXiv:2011.04244*.

[11] W. Wang, X. Chen, C. Yang, X. Li, X. Hu, and T. Yue, "Enhancing low light videos by exploring high sensitivity camera noise," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4110–4118.

[12] W. Chen, Y. Huang, M. Wang, X. Wu, and X. Zeng, "TSDN: Two-stage raw denoising in the dark," *IEEE Trans. Image Process.*, vol. 32, pp. 3679–3689, 2023.
[13] B. Peng, X. Zhang, J. Lei, Z. Zhang, N. Ling, and Q. Huang, "LVE-S2D: Low-light video enhancement from static to dynamic," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 12, pp. 8342–8352, Dec. 2022.
[14] X. Farhodov, O.-H. Kwon, K. W. Kang, S.-H. Lee, and K.-R. Kwon, "Faster RCNN detection based OpenCV CSRT tracker using drone data," in *Proc. Int. Conf. Inf. Sci. Commun. Technol. (ICISCT)*, Nov. 2019, pp. 1–3.
[15] M. Kristan et al., "The sixth visual object tracking VOT2018 challenge results," in *Computer Vision—ECCV 2018 Workshops*. Cham, Switzerland: Springer, 2019, pp. 3–53.
[16] A. Lukežič, T. Vojíř, L. Č. Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter tracker with channel and spatial reliability," *Int. J. Comput. Vis.*, vol. 126, no. 7, pp. 671–688, Jul. 2018.
[17] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with Kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.
[18] R. Liu, J. Cheng, and H. Lu, "A robust boosting tracker with minimum error bound in a co-training framework," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep. 2009, pp. 1459–1466.
[19] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 5388–5396.
[20] B. Wang and S. Fan, "An improved Canny edge detection algorithm," in *Proc. 2nd Int. Workshop Comput. Sci. Eng.*, vol. 1, 2009, pp. 497–500.
[21] N. Kiryati, Y. Eldar, and A. M. Bruckstein, "A probabilistic Hough transform," *Pattern Recognit.*, vol. 24, no. 4, pp. 303–316, Jan. 1991.
[22] G. Bal and P. Moireau, "Fast numerical inversion of the attenuated radon transform with full and partial measurements," *Inverse Problems*, vol. 20, no. 4, pp. 1137–1164, Aug. 2004.
[23] (2022). *K-Means Clustering*. [Online]. Available: https://machinelearningmastery.com/clustering-algorithms-with-python/
[24] COCO. (2022) *COCO Dataset*. [Online]. Available: https://cocodataset.org/#home



JONGHOEK KIM (Member, IEEE) received the B.S. degree in electrical and computer engineering from Yonsei University, Republic of Korea, in 2006, the M.S. degree in electrical and computer engineering from the Georgia Institute of Technology, USA, in 2008, and the Ph.D. degree from the Georgia Institute of Technology, in 2011, co-advised by Dr. Fumin Zhang and Dr. Magnus Egerstedt. He is with the System Engineering Department, Sejong University, Seoul, South Korea. He was a Professor with Hongik University, Republic of Korea, from 2018 to 2022. He was a Senior Researcher with the Agency for Defense Development, Republic of Korea, from 2011 to 2018. His research is on target tracking, control theory, robotics, multi-agent systems, and optimal estimation.