DroTrack: High-speed Drone-based Object Tracking Under Uncertainty

Ali Hamdi
RMIT University
Melbourne, Australia.
ali.ali@rmit.edu.au

Flora Salim
RMIT University
Melbourne, Australia.
flora.salim@rmit.edu.au

Du Yong Kim
RMIT University
Melbourne, Australia.
duyong.kim@rmit.edu.au

***Abstract*—We present DroTrack, a high-speed visual single-object tracking framework for drone-captured video sequences. Most of the existing object tracking methods are designed to tackle well-known challenges, such as occlusion and cluttered backgrounds. The complex motion of drones, i.e., multiple degrees of freedom in three-dimensional space, causes high uncertainty. The uncertainty problem leads to inaccurate location predictions and fuzziness in scale estimations. DroTrack solves such issues by discovering the dependency between object representation and motion geometry. We implement an effective object segmentation based on Fuzzy C Means (FCM). We incorporate the spatial information into the membership function to cluster the most discriminative segments. We then enhance the object segmentation by using a pre-trained Convolution Neural Network (CNN) model. DroTrack also leverages the geometrical angular motion to estimate a reliable object scale. We discuss the experimental results and performance evaluation using two datasets of 51,462 drone-captured frames. The combination of the FCM segmentation and the angular scaling increased DroTrack precision by up to 9% and decreased the centre location error by 162 pixels on average. DroTrack outperforms all the high-speed trackers and achieves comparable results in comparison to deep learning trackers. DroTrack offers high frame rates up to 1000 frame per second (*fps*) with the best location precision, more than a set of state-of-the-art real-time trackers.**

***Index Terms*—Drone-uncertainty, Real-time, single object tracking**

I. Introduction

Drone-related research has been widely pursued over the past several years. Drones are aerial platforms with advanced equipment, e.g., high-resolution cameras. They offer low-cost, safe operations to monitor locations inaccessible to humans. Classical imagery devices, such as satellite and street-level cameras, suffer from various limitations such as low resolution and low detail, respectively. Drones fly at low altitudes and offer a wide field of view to capture high-resolution images and greater detail. Visual drone applications include topographic mapping, surveillance, and search and rescue. However, drone-based video quality is affected by different uncertainties. Drones' motion is dependant on multiple situation inputs such as the weather conditions and structures of tracking locations.

Visual object tracking is a key component of the aforementioned drone applications. It has many challenges, such as noise, occlusion, cluttered backgrounds, and object varying features [1]. The intrinsic variability in object's colour or shape causes poor tracking predictions. This challenging issue is caused by the uncertainty of tracking environments' aspects. For example, occlusion can occur due to the shadows of trees and buildings. In addition, drones move with multiple degrees of freedom in three-dimensional space. This leads to unexpected changes in the drone and object locations pose multiple uncertainty and fuzziness issues in the object rotation and scale [2]. The point tracking algorithms depend on one or more features, i.e., key-points or corners. An incremental shift may occur in the tracking location due to the spatial distances between the correct and predicted tracking points. Thus, the performance of the existing tracking algorithms is degraded in different drone-based tracking situations. Moreover, drone-based object tracking requires real-time tracking. However, most of the recent trackers utilise deep learning to achieve high-accuracy tracking regardless of the tracking speed [3]. In this paper, we propose a robust object tracking algorithm that discovers the relationship between the object's visual and geometrical representations to overcome such challenges in real-time.

The main contribution of this paper is to solve the uncertainty problem in drone-based single object tracking. DroTrack tackles the impacts of object representation and geometrical drone motion uncertainties on the tracking location and scale. Fig. 1 illustrates the different components of our methodology. The proposed DroTrack makes the following contributions:

- Adaptive feature extraction and optical flow methods that produce real-time single object tracking.
- A spatial segmentation method that incorporates a Fuzzy C Means clustering algorithm with a pre-trained CNN transfer learning model.
- A heuristic geometrical method to estimate accurate object scales.
- Comprehensive evaluation and benchmark with the baseline and state-of-the-art trackers using two drone-captured datasets with 51,462 frames.

The structure of this paper is as follows. We discuss the related works and the problem scenario in Sections II and III, respectively. The research methodology underlying the DroTrack components is explained in Section IV. Section V presents the experimental results and evaluations, and Section VI concludes the work.

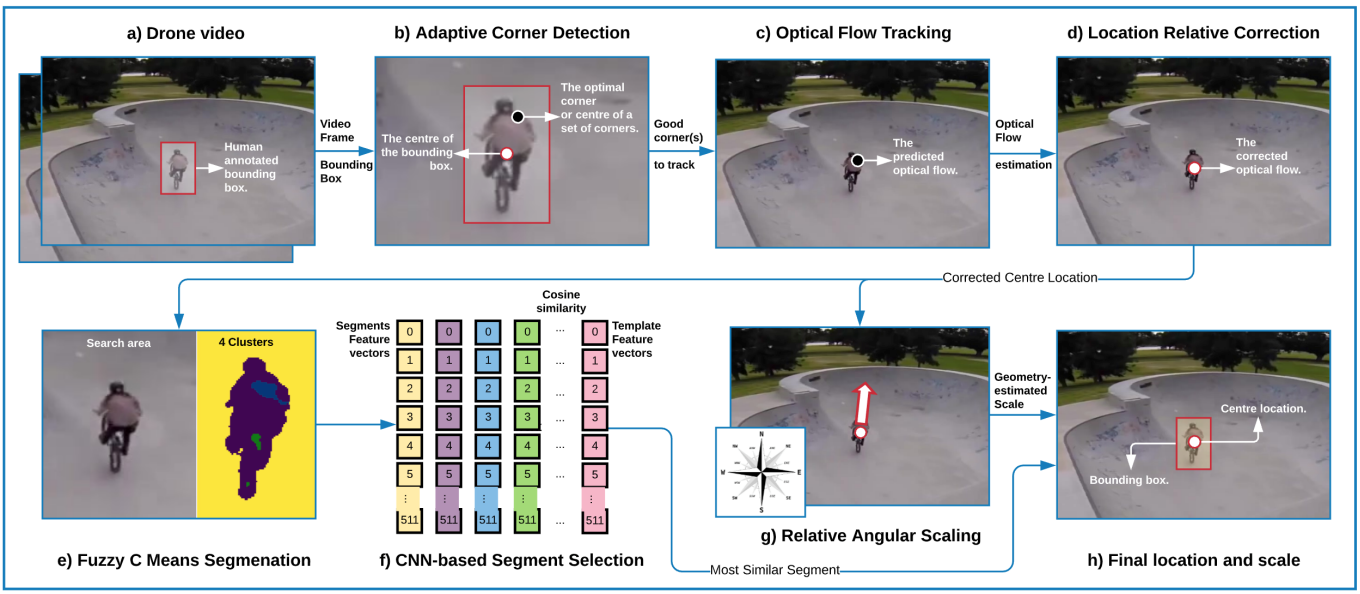

Fig. 1. DroTrack includes multiple components for single object tracking, as follows: a) reading the drone-captured video and the bounding box of the object at the first frame are given; b) detecting the optimal corner(s) to be tracked; c) estimating the optical flow; d) correcting the optical flow performance; e) segmenting using fuzzy c means clustering; f) using a VGG16 pre-trained model for extracting convolutional features and comparing the similarity between the feature vectors of the original reference template and the different segments to select the best one; g) calculating the relative angular scaling; and h) incorporating the outcome of (f) and (g) to produce the final tracking location centre and scale.

II. Related Work

The main goal of visual object tracking is to discriminate an object's area from the background in a sequence of frames. Tracking concerns the estimation of the location and scale of the object in each frame. In the literature, there are two main streams for object tracking methods, based on appearance and motion. Recently, object tracking has been studied in multiple studies, such as [4]–[6]. Much research is available on fixed camera scenarios, whereas only limited research can be found on moving camera situations [7].

There are a few studies dedicated to drone-based object tracking. The work by [8] utilised fast feature pyramids and a median flow tracker for pedestrian detection and tracking. Multiple hypothesis tracking (MHT) was used for multi-object tracking [9]. The MHT-based framework is based on a multidimensional assignment formulation using a time-slide window approach. The work by [10] developed a composite correlation filter that is adaptively tuned to recognise the object of interest. The authors in [11] implemented a multi-object Bayesian filter based on probability hypothesis density approximation. The implemented multi-object filter revises the weights of close tracking targets and reduces the disturbance of clutter. An online drone-based object tracking controller is developed in [12]. The proposed system tracks an object of predefined colour without external localisation sensors or GPS. This system corrected the predicted motion using a Kalman filter.

On the other hand, deep learning networks have been widely utilised in various computer vision applications including moving object tracking [3], [13]–[16]. The major problem is the limited knowledge that available to train the deep networks online to track objects that was only seen previously in one example. One possible solution is to train deep CNN for object tracking. However, the lack of annotated data hinders the training of the deep CNN. Training an offline CNN using a large set of videos with tracking ground-truths can solve this issue by transferring the learned features hierarchies to online tracking [17], [18]. Moreover, Graph Convolutional Networks are also proposed for object tracking [19]. The main issue of these methods is to perform Stochastic Gradient Descent online to adapt the weights of the network. This requires high computational speed which makes the tracking unreliable, especially in the drone scenario.

The existing trackers often fail for high-speed objects and unmodeled drone motion. For example, the directions of such high-speed objects can easily be changed by 360 degrees. The use of traditional trackers is also affected by their low-speed tracking, which is not preferred in drone scenarios. Most existing research in drone-based tracking depends on either appearance [12] or feature point detection and tracking [20]. We design our solution to consider new uncertainties in the drone-based object tracking. To produce an accurate drone-based object tracker, we propose to enhance the appearance-based tracking and use the motion characteristics of the object to calculate its relative scale.

III. Problem Formulation and Scenario

We formulate a drone-based tracking problem as follows. A drone d is tracking a moving o in real time using a camera as illustrated in Fig. 2. Unlike conventional object tracking using

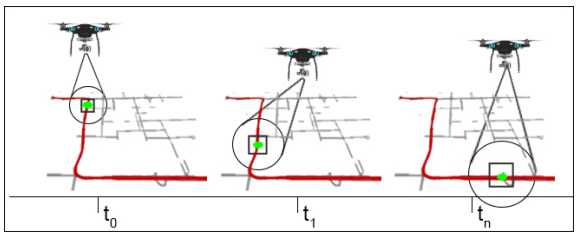

Fig. 2. Object locations and scales at two different time-stamps.

fixed cameras, a camera mounted on *d* is moving according to the motion of *d*. When *d* or *o* moves the distance between them is altered. This leads to changes in the location and scale of *o* in the video frame. We propose to solve these uncertainties issues using visual and geometrical reasoning. Fig. 2 shows three different tracking positions of a drone in different time-stamps. The drone monitors a moving object indicated in light green. As illustrated, the scale of the moving object is inversely related to the size of the drone's field of view. When the drone flies high and has a wide field of view, the object becomes smaller. Conversely, the object scale is enlarged if the drone becomes close. Fig. 3 shows the planar projection issue on two objects in the same scene. The bounding boxes *bb1* and *bb2* contain two men located at two different depths, i.e., distances from the drone camera. Their different depths affect their projections, which have different pixel representations in the image frame. Moreover, the boxes *b1* and *b2* represent two equal blocks on the grid, although their visual representations are varied. As will be discussed in Section V, the evaluation of DroTrack is done using two datasets that involve video segments captured in different environments with different scene structures and object depths.

We propose to track the object's location and infer its scale based on the variation of its visual representation and motion features. Using a Fuzzy based segmentation methodology helps to locate the object accurately. The accurate computing of geometrical relationships between the moving object and drone allows the tracking framework to ensure correct long-term tracking.

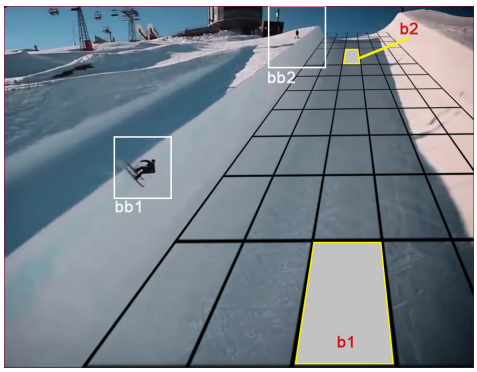

Fig. 3. An example of a planar scene.

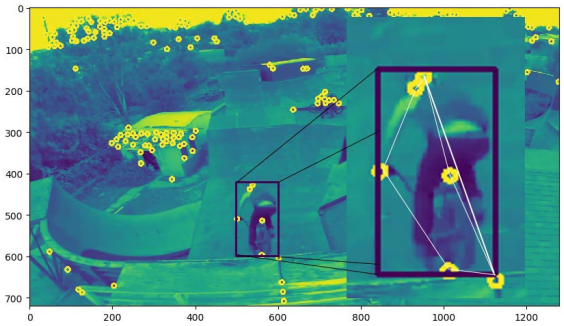

Fig. 4. An example of the the convex hull of multiple corners.

IV. DroTrack: Drone-based Object Tracking

DroTrack, the proposed drone-based object tracking method, has five main components, as showed in Fig. 1, as follows: 1) adaptive corner detection, 2) fast single-point optical flow tracking, 3) optical flow relative correction, 4) Fuzzy C Means based segmentation. 4) angular relative scaling. The feature-based tracking and intelligent scale estimation offer very high-speed tracking without compromising its accuracy. In the following sub-sections, we explain each component.

A. Adaptive Corner Detection Algorithm

A *corner* is a point-of-interest represented as an image pixel where any detected edge changes its direction significantly in two dimensions. Corners offer a better choice for object tracking. They enable tracking changes in two dimensions that cannot be detected with other features, such as edges, where the changes are only in one dimension [21]. DroTrack detects the strong corners on the first frame (FF) using the Shi Tomasi method [22]. The best corners are selected based on their Shi Tomasi parameters (STPs), such as quality level, minimum distance, a derivative covariation matrix block size and a maximum number of corners. The Shi Tomasi method is an extension of the Harris Corner Detector (HCD). The HCD enables invariance to rotation, scale, illumination variation and noise. It utilises a local auto-correlation function that employs small shifts to measure local changes in different directions. Here, DroTrack focuses on the closest corner(s) to the centre of the reference template (RT), i.e., the image segment that falls in the human-annotated bounding box at the first frame. There maybe one or more closest corners detected inside the RT. Fig. 4 shows an example of detected multiple corners (coloured in yellow) and the connecting convex hull (coloured on white). For simplicity, in the rest of this paper, closest corner (s) **CC** is used to represent one or multiple selected corners. In order to achieve this, we propose an adaptive algorithm that works recursively to find the CC. The algorithm begins the corners detection with STPs of high quality, small corner number, distance threshold and block size. The algorithm decides either to continue tracking these corner or to tune the STPs and redo the previous process until the best CC is found based on two evaluation scores: similarity and distance.

We define a test template to be the area positioned around the detected corner, with the same width and height of the RT. In case of multiple corners, the centre of their convex hull is used as the template centre, see Fig. 4. Then we extract the histograms, i.e., graphical representations of the tonal distribution of the image pixel intensity values, of both the RT and the test template. The similarity ratio (*simi.*) is computed as a correlation score between the histograms of the RT and the test template, both in colours. We use the correlation measured according to Eq. 1 to compute the correlation ratio between the two histograms. The score is between 0 and 1 for the lowest and highest similarities, respectively. If the *simi.* is below a certain threshold (α), the algorithm will adapt the feature extraction parameters to achieve a more fine-grained search.

$$simi.(H_1, H_2) = \frac{\sum_I (H_1(I) - \bar{H}_1)(H_2(I) - \bar{H}_2)}{\sqrt{\sum_I (H_1(I) - \bar{H}_1)^2 \sum_I (H_2(I) - \bar{H}_2)^2}} \tag{1}$$

Where H_1 and H_2 are the histograms of the RT and test template, *I* iterates for each histogram bin, and the $\bar{H}$ refers to the histogram mean. The distance between the RT centre point and the CC coordinates is computed. DroTrack relates the distance threshold (β) to the object scale. It also requires the CC to be less distant than the distance-scaled threshold (β). Using the distance formulas in Eq. 1 and 2 based on the histogram correlation and Pythagorean theorem, respectively, the function f in Eq. 3 compares the *simi.* and *dist.* for the given corner. Eq. 3 returns 1 if the corner is selected or 0 to rerun the algorithm. Here, α and β represent the similarity and distance thresholds and *C* is the given corner. The threshold of the *simi.* (α) is defined as 0.5 of the histogram similarity and the *dist.* (β) is defined as 0.5 of the sum of boundary box's height and width.

$$dist.((x_1, y_1), (x_2, y_2)) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{2}$$

where, x_1, y_1 and x_2, y_2 represent the coordinates of the RT centre and the given corner, respectively.

$$f(C) = \begin{cases} 1, & \text{if } simi. \geq \alpha \ \ \& \ \ dist. \leq \beta \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

Specifically, the adaptive corner detector uses a set of minimum and maximum thresholds that are used to match the *simi.* and *dist.* So, the algorithm keeps iterating on the given threshold range until it meets the *simi.* and *dist.*. The Open CV library implements the Shi Tomasi method with a region of interest parameter, whereas the proposed adaptive method is well designed to enhance the performance of the algorithm by starting at higher quality and with fewer corners. In addition, the proposed algorithm overcomes the limitation of the dependency of corner-based tracking on the environmental factors.

B. Optical Flow Tracking

The selected CC is used to develop the optical flow tracking method. DroTrack employs the implementation of a sparse iterative Lucas-Kanade optical flow in pyramids [23]. The optical flow method takes two consecutive frames, and a set of corner coordinates belong to the previous frame (PF). In our case, DroTrack only passes the CC coordinates to the optical flow method and obtains the new coordinates.

C. Optical Flow Relative Correction

We consider the distance between the CC (the closest corner or the convex hull centre) and the FF centre point as a correction margin; see Δ_x and Δ_y in Eq. 4 and Fig. 1 at (b). The new coordinates are calculated based on the relative scale of the new RT; see RT_{scale_F} in Eq. 5.

$$\Delta_x = FF_x - F_x, \Delta_y = FF_y - F_y, \tag{4}$$

$$RT_{scale_{FF}} = \frac{h_{RT_{FF}}}{h_{FF}}, RT_{scale_F} = \frac{h_{RT_F}}{h_F} \tag{5}$$

where h refers to the height.

Thus, the x and y coordinates of corrected point (CP) are computed as in Eq. 6 and Eq. 7, respectively.

$$CP_x = F_x + \Delta_x * \frac{RT_{scale_F}}{RT_{scale_{FF}}} \tag{6}$$

$$CP_y = F_y + \Delta y * \frac{RT_{scale_F}}{RT_{scale_{FF}}} \tag{7}$$

The relative correction is useful due to the fact that whenever the object moves away from the camera its size changes and has different location centre. The output corrected location centre is used later by the Fuzzy-based segmentation and geometrical algorithms.

D. Object Segmentation with Fuzzy C-means

In some cases, DroTrack loses tracking of the object due to the uncertainty or fuzziness of the bounding area around the object. Therefore, we propose to apply object segmentation based on a Fuzzy C Means (FCM) methodology. The conventional FCM is sensitive to the image noise. Basic FCM algorithm expects the data to have separate clusters in order to produce accurate membership values. However, this dependency on the cluster similarity is not suitable for image data. This is because the neighbour clusters in an image are highly correlated. Multiple research works propose to overcome this problem, such as [24]. They harness the spatial information to overcome the sensitivity issue of FCM. They simply compute the likelihood that a neighbourhood pixel belongs to a certain cluster. Then, the spatial likeliness score is injected into the membership function. We employ the methodology presented in [25]. They propose to incorporate the hesitation degree and spatial likeliness in the membership function calculated as (msh) in Eq. 8.

$$msh_{ij} = \frac{u_{ij}^p h_{ij}^q}{\sum_{k=1}^{c} u_{kj}^p h_{kj}^q} \tag{8}$$

where msh_{ij} represents the membership values of the given neighbourhood pixel, i and j represent the pixel coordinates, u is the membership function computed with hesitation score, h is the spatial function, and p and q control the weights of the initial membership and spatial functions, respectively. We then apply morphological transformation methods, including erosion and dilation, to clean the noise after segmentation. We decide to segment the search area into n clusters, e.g., 5.

We use a pre-trained VGG16 [26] network, trained on the ImageNet dataset, to extract the convolutional features of the clustered segments. This process offers to transfer the learning of that large dataset to produce discriminative features vectors. Then, we calculate the cosine distance between the feature vectors of the original reference template and each segments as in Eq. 9.

$$\cos simi.(\mathbf{T},\mathbf{S}) = \frac{\mathbf{TS}}{\|\mathbf{T}\|\|\mathbf{S}\|} = \frac{\sum_{i=1}^{n}\mathbf{T}_i\mathbf{S}_i}{\sqrt{\sum_{i=1}^{n}(\mathbf{T}_i)^2}\sqrt{\sum_{i=1}^{n}(\mathbf{S}_i)^2}} \tag{9}$$

where $\mathbf{T}$ and $\mathbf{S}$ represent the template and segment feature vectors, respectively. Fig. 5 shows three different examples of the FCM based segmentation process. The first row shows a two-clusters FCM segmentation. This example shows better bounding box estimation that the one in the second row with three clusters. The last row comes with five clusters to extract the sheep from the surrounding grass area. We incorporate the performances for the relative angular scaling with the FCM segmentation for the best tracking results.

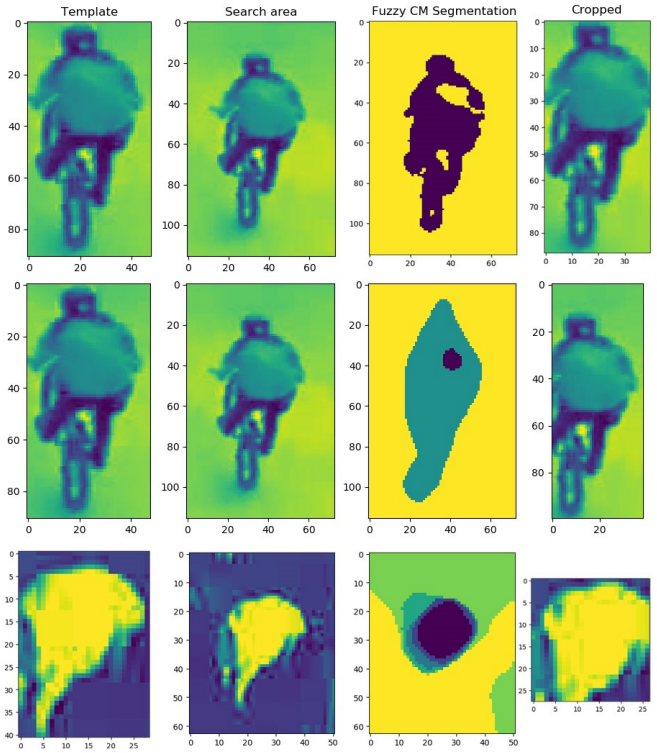

Fig. 5. Fuzzy C Means based segmentation examples with different cluster numbers. 1st and 2nd cluster 2 and 3 segments for the same object, and 3rd clusters 5 segments.

E. Relative Angular Scaling

DroTrack updates the object scale based on its recent motion features. Here, scale refers to the size of the current tracking template area in relation to the previous template.

The new scale is dependent on its distance from the drone camera. Since the utilised datasets do not have camera parameters, DroTrack computes the new scale in two-dimensional projection. The motion model; i.e., speed and direction, are calculated between the coordinates of the centre locations in the previous and current frames. When the object moves up in the image; i.e., has a negative change of the coordinate y, the scale is relatively reduced. Moving closer to the drone camera; i.e., having a high y value, the scale is enlarged. In order to make the scaling algorithm more accurate, we relate the scale ratio to the motion angle. The more vertical the object's direction is, the higher the scale ratio it has. For example, if an object is moving vertically at $\frac{\pi}{2}$ or $\frac{-\pi}{2}$ from the drone camera, its scale is at the highest possible ratio. However, if the moving angle becomes low, the scale ratio will be small.

Fig. 6 shows the concept of the proposed relative angular scaling algorithm. We feed the algorithm with the current and previous templates as well as the motion angle (θ). The angle between the coordinates x,y in the two templates is calculated using the two-argument *atan2*. The *atan2* computes the angle between the positive x-axis of a plane and the coordinates x, y on it according to Eq. 10.

$$\theta((x_1,y_1),(x_2,y_2)) = atan2(y_2-y_1, x_2-x_1) \in (-\pi,\pi) \tag{10}$$

The frame F is divided into four zones that are sliced from the coordinates of the current location centre. In the case that Δy, i.e., the difference in the coordinate y between the two frames is negative, two zones are defined as $\frac{-\pi}{2} < angle < 0$ and $-\pi < angle < \frac{-\pi}{2}$. For the positive case, Δy, $0 < angle < \frac{\pi}{2}$ and $\frac{\pi}{2} < angle < \pi$. The red arrows in Fig. 6 point up and down to the negative and positive scaling in each zone, respectively. The template scale is computed as its height over the height of the current frame F. The relative ratio is calculated as the ratio between the current frame y coordinate and the previous one. Based on the fact that the scale is relatively dependent on the motion angle, the algorithm computes the new relative scale under one of seven conditions, as in Eq. 11. In the initial case, the algorithm returns the previous RT (PRT) scale when there has been no change in the previous motion model. Two cases are directly scaled-down and -up for angle $\frac{-\pi}{2}$ and $\frac{\pi}{2}$, respectively. The other four cases include two cases when the $\Delta x > 0$ are normalised with their angle over $\frac{\pi}{2}$, and two when $\Delta x < 0$. The latter two cases have inverse directions. Therefore, they are first subtracted from 180 and normalised over $\frac{\pi}{2}$.

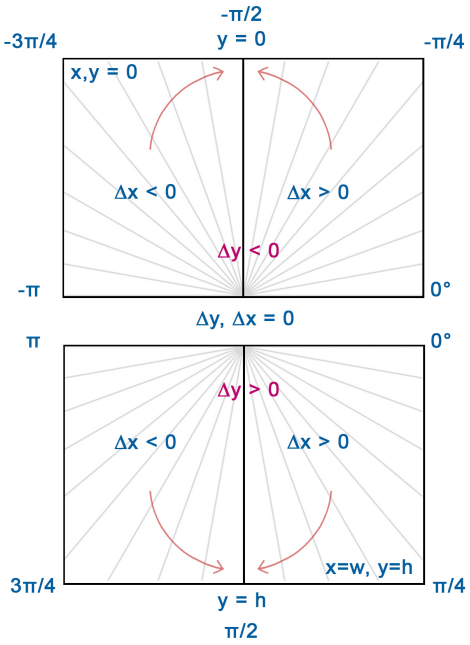

Fig. 6. The angular relative scaling zones and their constraints. The Δx and Δy refer to the differences in the coordinates between the two frames.

$$RT = \begin{cases} PRT & \rightarrow \quad \Delta x \wedge \Delta y = 0; \\ PRT * Scale & \rightarrow \quad \theta = \frac{\pi}{2} \vee \theta = \frac{-\pi}{2}; \\ PRT * Scale * \frac{\theta}{\pi/2} & \rightarrow \quad \Delta x > 0 \wedge \theta > 0; \\ PRT * Scale * \frac{\theta}{\pi/2} & \rightarrow \quad \Delta x > 0 \wedge \theta < 0; \\ PRT * Scale * \frac{\pi-\theta}{\pi/2} & \rightarrow \quad \Delta x < 0 \wedge \theta < 0; \\ PRT * Scale * \frac{\pi-\theta}{\pi/2} & \rightarrow \quad \Delta x < 0 \wedge \theta > 0; \end{cases} \tag{11}$$

The experimental results showed that the angular method is effective for scale adaptation. Since the object motion is projected in two dimensions, the algorithm can capture accurate scale changes regardless of the object direction.

V. Experimental Results and Evaluations

The proposed DroTrack algorithm is implemented in Python using standard methods for feature extraction and optical flow in the OpenCV library. Fig. 1 highlights the work-flow and the inter-relationships among the DroTrack components.

Datasets We used two publicly released datasets: DTB70 [7] and UAV123 [27]. The two datasets consist of 51,462 frames. The datasets are of high diversity and captured in multiple environments. For examples, see Fig. 3, 1, and 12. These datasets cover more difficulties and uncertainties aspects that are not found in the traditional tracking datasets such as VOT [28] and VTB50 [29]. The datasets include both translation and rotation camera motions. The results show that this dataset is challenging for conventional tracking algorithms. They also cover highly challenging cases in both short-term and long-term occlusion. The datasets contain different moving object types, such as humans, animals, cars, boats, birds and drones. This offers different levels of degree of freedom for the motion. Objects like cars and boats can only translate or rotate, whereas humans and animals, birds and drones have a higher degree of freedom. The datasets outdoor scenes are in various situations, including significantly varied backgrounds. These challenging motion characteristics cause object deformation, leading to more difficult object tracking.

Evaluation metrics To evaluate DroTrack, we computed the success overlap and centre location error. The intersection over union (IoU) is used to compute the success plots and the precision thresholds for the centre location errors. The IoU is an evaluation metric used to measure the tracking accuracy. IoU is computed for each frame using the predicted boundary box and the ground truth box. The precision score is calculated with a set of thresholds of centre location for each frame prediction. The trackers are ranked using the area under the curve (AUC) metric for the success plot and the representative precision at the threshold of (ϵ = 20 and 100) for the precision plot. All the reported results are in one-pass evaluation (OPE) settings.

Ablation study We first run two versions of DroTrack (with the angular module only), on the DTB70 dataset, with and without the algorithm for the relative correction (*rc*) of the optical flow. The experimental results show that the *rc* algorithm improved the precision ($\epsilon = 100$) score from 0.62 to 0.75 and the IoU from 0.21 to 0.25. For even faster tracking, we implemented DroTrack in two different modes using full-size (*fs*) and half-size (*hs*) frames. The results show that reducing the frame size to half decreased the tracking computational costs. However, the tracking success overlap and precision scores are slightly degraded. Table I lists the results of the ablation study of the DroTrack versions. Using the *rc* algorithm with the *fs* mode produces the best centre location precision ($\epsilon = 20$) / ($\epsilon = 100$) and the best IoU success overlap score. With an average of 206 *fps*, it enables these accurate results at real-time speeds. The DroTrack version without the *rc* in the *hs* mode results in very high speed tracking, with **1033** *fps*. Fig. 7 shows the significance of the proposed framework. It illustrates the performance precision of DroTrack using only the Fuzzy-based segmentation in comparison to the geometrical angular scaling. It also shows how DroTrack enhance the FCM based results by adding the proposed geometrical angular method. In the following experiments, we show the three different versions of DroTrack, i.e., DroTrack-FCM based, DroTrack-Angular, DroTrack (having both), in comparison to the state-of-the-art trackers.

Quantitative and Qualitative Benchmark We compare DroTrack with nine state-of-the-art and baseline trackers, including CSRT [30], ADNet [31], SiamFC [3], MIL [32], kernelised correlation filters (KCF) [33], Median Flow [34], Boosting [35], MOSSE [36] and TLD [37]. The implementations of these trackers are found in OpenCV package and at the GitHub platform. All the benchmarking experiments were done using one CPU of Intel Core i7-3635QM and 8 GB SDRAM. We compare three versions of DroTrack, including 1) the DroTrack with FCM, 2) DroTrack with Angular scaling,

TABLE I
COMPARISON RESULTS BETWEEN THE PROPOSED DROTRACK VERSIONS USING THE DTB70 DATASET.

Tracker	*P*.100	*P*.20	IoU	Time	*fps*
DroTrack *rc fs*	**0.75**	**0.41**	**0.25**	**0.0048**	**206**
DroTrack *rc hs*	**0.67**	**0.35**	0.23	0.0012	**840**
DroTrack *fs*	**0.63**	**0.36**	0.21	0.0036	275
DroTrack *hs*	**0.62**	0.34	0.20	0.0010	**1033**

Note: *rc* refers to using the relative correction algorithm. *fs* and *hs* refer to using the full- and half-sizes of the given frames.

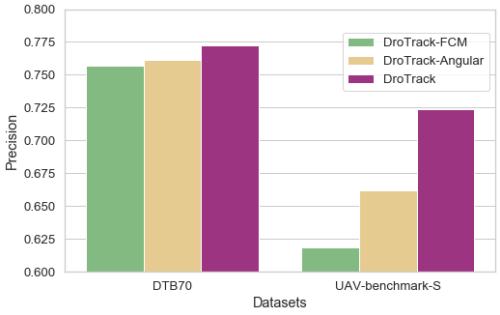

Fig. 7. DroTrack precision of using FCM, angular geometry of for the utilised two datasets.

and 3) DroTrack with both FCM and Angular scaling. Tables II and III show the mean IoU and precision, and *fps* for DroTrack and the other trackers. The scores are highlighted in these tables in different colours: **green** for 1st rank, **blue** for 2nd rank, **red** for 3rd rank, and **orange** for 4th rank. Fig. 5 shows the benchmarking results on the four datasets. The four columns compare: the IoU, Precision ($\epsilon = 20$), Precision ($\epsilon = 100$), and tracking (*fps*). Table II and Fig. 9 show the benchmarking results using the DTB70 dataset. DroTrack ranks second (out of 10) for the average distance and third for all other experiments. Using the FCM produces good Precision results ($\epsilon = 20\&100$) where it ranks fourth and fifth. DroTrack with the angular method achieves better distance average and tracking speed. The combination of the two algorithms ranks second in terms of the distance average and comes better than using each method separately. Here, DroTrack outperforms all the high-speed trackers and achieves promising results in comparison to deep learning trackers. It has 0.29, 0.73 for mean and lowest IoU; 0.43 and 0.77 for the Precision; and 206 and 65 *fps* for the tracking speed. Table III and Fig. 11 highlight the experimental benchmarking results utilising the UAV-Benchmark-S dataset. DroTrack ranks second and fourth in the average of the error distance with 82 for the combination version and 93 for the angular one, respectively. The deep learning SiamFc comes better than DroTrack with a distance error of 75. However, DroTrack still outperforms other deep learning trackers such as CSRT and ADNet. Moreover, the average tracking speed of DroTrack is high with 47, 383 (3rd rank), and 80 *fps* in the three versions. However, the IoU comes lower with only 0.24 and 0.62. However, DroTrack here is still better than all the other high-speed trackers.

Fig. 8 and 10 show polar radar charts comparing the three version of DroTrack with the benchmarking trackers. The charts represent the results in terms of precision, IoU, time and tracking speed. The figures show how real-time trackers produce low accuracy and high speeds. In contrast, deep learning-based trackers have high accuracy and low speeds. Here, DroTrack balances the performance between accuracy and speed. In terms of accuracy, DroTrack outperforms all the real-time trackers and compete with the deep learning ones. For the tracking speed, DroTrack outperforms all the deep learning trackers.

TABLE II
BENCHMARKING ON THE DTB70 DATASET.

Tracker	*P*.100	*P*.20	Dist.	IoU	*fps*
CSRT [30]	**0.80**	**0.53**	**95**	**0.35**	31
SiamFC [3]	**0.86**	**0.72**	**69**	**0.51**	3.6
MOSSE [36]	0.22	0.16	579	0.10	**2692**
ADNet [31]	0.25	0.12	356	0.09	0.2
Boosting [35]	0.55	0.34	184	0.22	25.8
TLD [37]	0.44	0.25	254	0.16	5.2
KCF [33]	0.14	0.11	656	0.08	**800**
MIL [32]	0.68	0.43	131	**0.27**	21
Median Flow [34]	0.47	0.33	255	0.23	**187**
DroTrack-FCM (Ours)	**0.74**	**0.44**	**124**	**0.25**	**38.6**
DroTrack-Angular (Ours)	**0.75**	**0.41**	**94**	**0.27**	**206**
DroTrack (Ours)	**0.77**	**0.43**	**86**	**0.29**	**65**

Colours note: **1st, 2nd, 3rd, and 4th ranks.**

TABLE III
BENCHMARKING ON THE UAV-BENCHMARK-S DATASET.

Tracker	*P*.100	*P*.20	Dist.	IoU	*fps*
CSRT [30]	**0.78**	**0.65**	**89**	**0.35**	36.3
SiamFC [3]	**0.86**	**0.77**	**75**	**0.48**	3.6
MOSSE [36]	0.29	0.19	463	0.10	**2750**
ADNet [31]	0.49	0.40	236	0.24	0.4
Boosting [35]	**0.73**	**0.55**	112	**0.29**	41
TLD [37]	0.27	0.15	298	0.09	8
KCF [33]	0.25	0.24	482	0.15	**1089**
MIL [32]	0.66	0.45	131	0.21	21.9
Median Flow [34]	0.53	0.44	250	**0.23**	**271**
DroTrack-FCM (Ours)	**0.60**	**0.44**	**188**	**0.15**	**47.0**
DroTrack-Angular (Ours)	**0.64**	**0.43**	**93**	**0.22**	**382.8**
DroTrack (Ours)	**0.72**	**0.48**	**82**	**0.24**	**80**

Colours note: **1st, 2nd, 3rd, and 4th ranks.**

Fig. 12 shows a sample of 12 frames with the predicted template rectangle for each tracker. The results show the accurate position and scale of DroTrack predictions. In many cases, such as in frames b, d, h, i, and j, the other trackers produce erroneous scales larger than the actual ones. The drone motion and scene illumination distract the trackers. However, the proposed Fuzzy segmentation and angular relative scale algorithms enable accurate DroTrack results. DroTrack's performances support this interpretation, by having low centre location errors (e.g. Precision scores) from the ground truth and consistent IoU and precision results. This low error and consistency prove the superiority of DroTrack. In the scene j, however, DroTrack's scale prediction is not accurate. This inaccurate prediction is due to the high similarity between the bounding box and the surrounding area. Specifically, is this

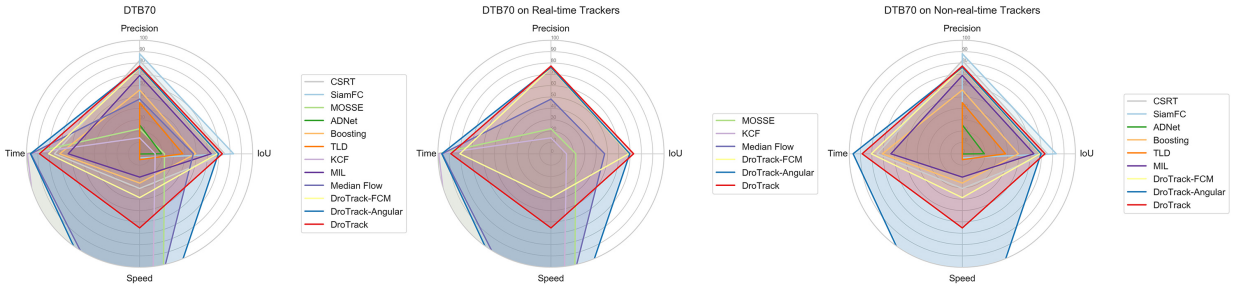

Fig. 8. Radar charts benchmarking on the DTB70.

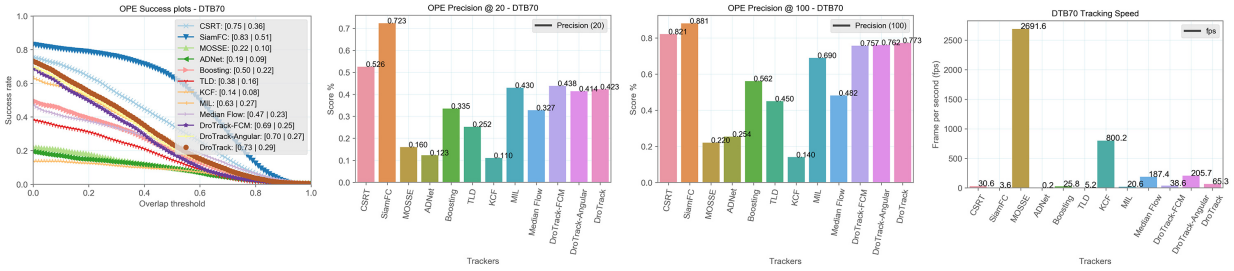

Fig. 9. Benchmarking results of the IoU, Precision ($\epsilon = 20$) and ($\epsilon = 100$), and tracking speed *fps*, for the DTB70 dataset.

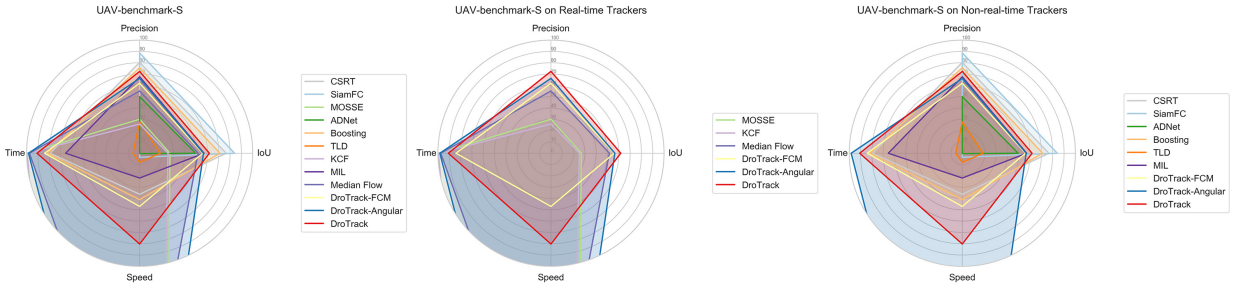

Fig. 10. Radar charts benchmarking on the UAV-benchmark-S.

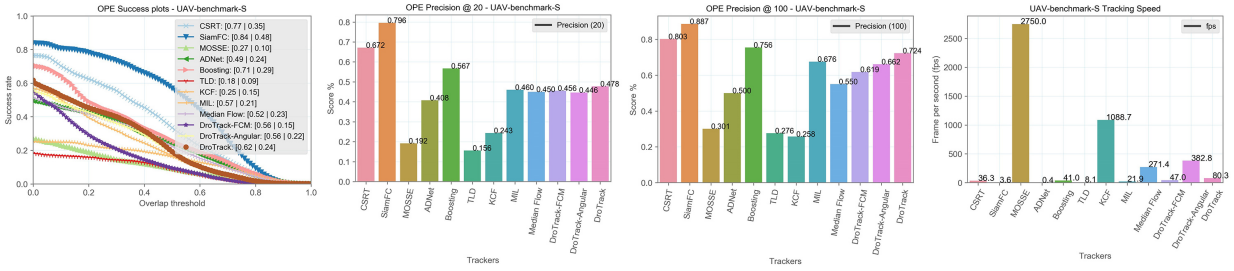

Fig. 11. Benchmarking results of the IoU, Precision ($\epsilon = 20$) and ($\epsilon = 100$), and tracking speed *fps*, for the UAV-benchmark-S dataset.

scene (j), the tracked sheep is surrounded by multiple sheep. Therefore, the FCM based segmentation was not successful. However, the motion correction algorithms lead DroTrack to better locate the centre tracking than the other trackers in J. In most scenes, DroTrack seems to estimate smaller scales; however, in scene k, DroTrack has a relatively large-scale. The vertical drone motion on the moving object seems to vary the object depth significantly. In this case, if the drone moves high up from the object, the object scale will be decreased. However, the stationary nature of the object projection in two dimensions will prevent DroTrack from generating a better scale estimation. Therefore, it worth as future work to investigate DroTrack's sensitivity to the object's motion and try to overcome the challenge of lacking the scene depth.

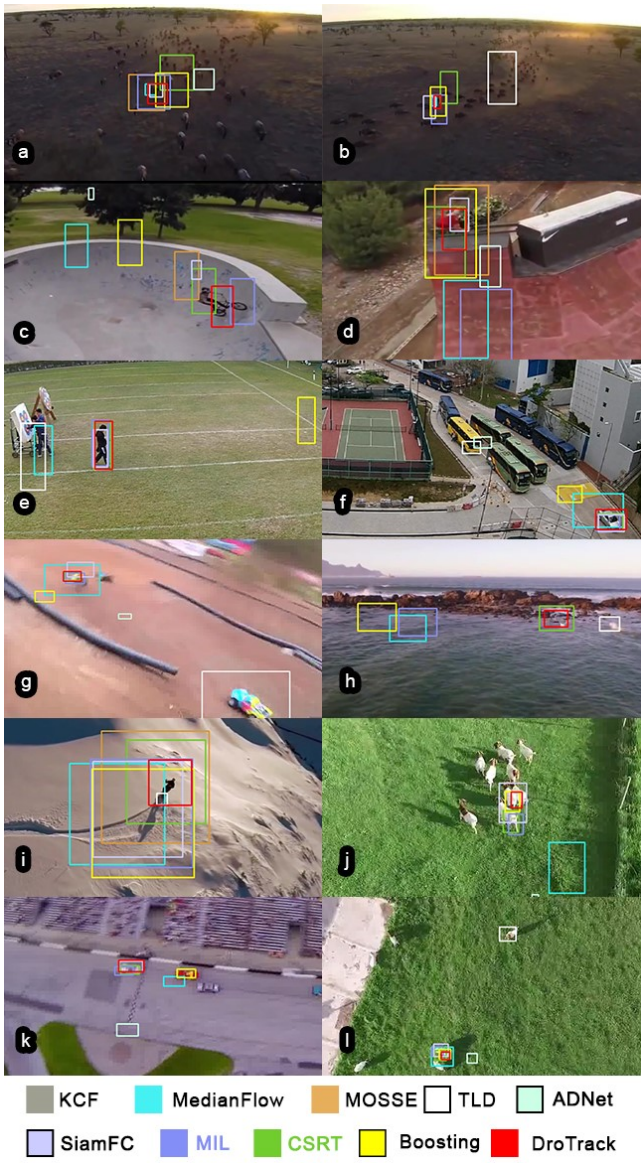

Fig. 12. Sample frames for tracking results.

Computational performance To compare computation time, the tracking speed columns in Fig. 9 and 11 and Tables II and III summarise the averages of the execution time periods and the FPS (*fps*) for each tracker. The MOSSE tracker has the best average time followed by the KCF and half-size version of DroTrack. DroTrack-Angular is always faster than the other two variations. In the combined version, sometimes, the FCM segmentation process is skipped due to the poor outcome. Therefore in some cases, the speed of the combined version comes faster than the FCM based one. Although the deep learning-based trackers, such as SiamFC achieves better IoU than DroTrack, they cannot be implemented in real-time drone scenarios. SiamFC only processed 3.5 *fps* and the ADNet takes more than three seconds to process one frame. In addition to having the promising success overlap and location centre precision, DroTrack offers high frame rates of more than 1000 *fps*. This high computational speed is due to the adaptive components of DroTrack; e.g., its adaptive corner detection. DroTrack starts with a high level of quality and a low number of corners to decrease the tracking time. Tracking one optimal corner contributes to the real-time performance of DroTrack. This shows the superiority of using DroTrack for high-speed real-time tracking.

VI. Conclusion

We have introduced a novel drone-based single object tracking algorithm, called DroTrack. We described the dependency between the object motion model and the visual projection model. A Fuzzy C Means based segmentation algorithm was utilised to solve the visual uncertainty issues. An angular relative scaling algorithm was also developed to manage object scale variations. The performance of DroTrack is promising in comparison to the state-of-the-art and baseline trackers. In future work, the DroTrack scale algorithm should be enhanced to overcome the problem of the missing object depth. Although DroTrack outperforms the high-speed trackers and achieves promising results in comparison to deep learning trackers, new methods for reference template update can be considered as future work for further improvement.

Acknowledgment

Ali Hamdi is supported by RMIT Research Stipend Scholarship. This research is also supported partially by the Australian Government through the Australian Research Council's Linkage Projects funding scheme (project LP150100246).

References

[1] Jiří Apeltauer, Adam Babinec, David Herman, and Tomáš Apeltauer. Automatic vehicle trajectory extraction for traffic analysis from aerial video data. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 40(3):9, 2015.

[2] Thomas Moranduzzo and Farid Melgani. Automatic car counting method for unmanned aerial vehicle images. *IEEE Transactions on Geoscience and Remote Sensing*, 52(3):1635–1647, 2014.

[3] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016.

[4] Chenglong Li, Liang Lin, Wangmeng Zuo, and Jin Tang. Learning patch-based dynamic graph for visual tracking. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[5] Tianzhu Zhang, Changsheng Xu, and Ming-Hsuan Yang. Multi-task correlation particle filter for robust object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4335–4343, 2017.

[6] Matthias Mueller, Neil Smith, and Bernard Ghanem. Context-aware correlation filter tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1396–1404, 2017.

[7] Siyi Li and Dit-Yan Yeung. Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models. In *AAAI*, pages 4140–4146, 2017.

[8] Li Zhang, Zenghui Zhang, and Huilin Xiong. Visual pedestrian tracking from a UAV platform. In *2017 2nd International Conference on Multimedia and Image Processing (ICMIP)*, pages 196–200. IEEE, 2017.

[9] Liu Jianfang, Zheng Hao, and Gao Jingli. A novel fast target tracking method for UAV aerial image. *Open Physics*, 15(1):420–426, 2017.

[10] Leopoldo N Gaxiola, Victor H Diaz-Ramirez, Juan J Tapia, and Pascuala García-Martínez. Target tracking with dynamically adaptive correlation. *Optics Communications*, 365:140–149, 2016.

[11] Huanqing Zhang, Hongwei Ge, Jinlong Yang, and Yunhao Yuan. A gm-phd algorithm for multiple target tracking based on false alarm detection with irregular window. *Signal Processing*, 120:537–552, 2016.

[12] Alex G Kendall, Nishaad N Salvapantula, and Karl A Stol. On-board object tracking control of a quadcopter with monocular vision. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 404–411. IEEE, 2014.

[13] Xin Li, Chao Ma, Baoyuan Wu, Zhenyu He, and Ming-Hsuan Yang. Target-aware deep tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1369–1378, 2019.

[14] Janghoon Choi, Junseok Kwon, and Kyoung Mu Lee. Deep meta learning for real-time target-aware visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 911–920, 2019.

[15] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Convolutional features for correlation filter based visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 58–66, 2015.

[16] Naiyan Wang, Siyi Li, Abhinav Gupta, and Dit-Yan Yeung. Transferring rich feature hierarchies for robust visual tracking. *arXiv preprint arXiv:1501.04587*, 2015.

[17] Naiyan Wang, Siyi Li, Abhinav Gupta, and Dit-Yan Yeung. Transferring rich feature hierarchies for robust visual tracking. *CoRR*, abs/1501.04587, 2015.

[18] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. *CoRR*, abs/1510.07945, 2015.

[19] Junyu Gao, Tianzhu Zhang, and Changsheng Xu. Graph convolutional tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4649–4659, 2019.

[20] Andreas Nussberger, Helmut Grabner, and Luc Van Gool. Aerial object tracking from an airborne platform. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 1284–1293. IEEE, 2014.

[21] Lubas Juranek, Jiri Stastny, and Vladislav Skorpil. Effect of low-pass filters as a shi-tomasi corner detector's window functions. In *2018 41st International Conference on Telecommunications and Signal Processing (TSP)*, pages 1–5, July 2018.

[22] Jianbo Shi and Carlo Tomasi. Good features to track. in proceedings of the ieee conference on computer vision and pattern recognition, seattle, usa, pp. 593-600. 1994.

[23] Jean-Yves Bouguet. Pyramidal implementation of the lucas kanade feature tracker. *Open Source Computer Vision Library*, 2003.

[24] Keh-Shih Chuang, Hong-Long Tzeng, Sharon Chen, Jay Wu, and Tzong-Jer Chen. Fuzzy c-means clustering with spatial information for image segmentation. *computerized medical imaging and graphics*, 30(1):9–15, 2006.

[25] BK Tripathy, Avik Basu, and Sahil Govel. Image segmentation using spatial intuitionistic fuzzy c means clustering. In *2014 IEEE International Conference on Computational Intelligence and Computing Research*, pages 1–5. IEEE, 2014.

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[27] Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for UAV tracking. In *European conference on computer vision*, pages 445–461. Springer, 2016.

[28] Matej Kristan, Roman Pflugfelder, Ales Leonardis, Jiri Matas, Luka Čehovin, Georg Nebehay, Tomas Vojir, Gustavo Fernandez, Alan Lukežič, Aleksandar Dimitriev, et al. The visual object tracking vot2014 challenge results," 2014, 2014.

[29] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2411–2418, 2013.

[30] Alan LukeźIăź, Tomáš Vojíř, Luka Čehovin Zajc, Jiří Matas, and Matej Kristan. Discriminative correlation filter tracker with channel and spatial reliability. *International Journal of Computer Vision*, 126(7):671–688, 2018.

[31] Sangdoo Yun, Jongwon Choi, Youngjoon Yoo, Kimin Yun, and Jin Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2711–2720, 2017.

[32] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 983–990. IEEE, 2009.

[33] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[34] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *Pattern recognition (ICPR), 2010 20th international conference on*, pages 2756–2759. IEEE, 2010.

[35] Helmut Grabner, Michael Grabner, and Horst Bischof. Real-time tracking via on-line boosting. In *Bmvc*, page 6, 2006.

[36] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2544–2550. IEEE, 2010.

[37] Zdenek Kalal, Krystian Mikolajczyk, Jiri Matas, et al. Tracking-learning-detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409, 2012.