sensors

MDPI

Article

Onboard Robust Visual Tracking for UAVs Using a Reliable Global-Local Object Model

Changhong Fu [1,2], Ran Duan [1,2], Dogan Kircali [1,2] and Erdal Kayacan [1,*]

[1] School of Mechanical and Aerospace Engineering, Nanyang Technological University (NTU), 50 Nanyang Avenue, Singapore 639798, Singapore; changhongfu@ntu.edu.sg (C.F.); duanran@ntu.edu.sg (R.D.); dkircali@ntu.edu.sg (D.K.)

[2] ST Engineering-NTU Corporate Laboratory, Nanyang Technological University, 50 Nanyang Avenue, Singapore 639798, Singapore

* Correspondence: erdal@ntu.edu.sg; Tel.: +65-9728-8774

Academic Editor: Felipe Gonzalez Toro
Received: 13 July 2016; Accepted: 25 August 2016; Published: 31 August 2016

Abstract: In this paper, we present a novel onboard robust visual algorithm for long-term arbitrary 2D and 3D object tracking using a reliable global-local object model for unmanned aerial vehicle (UAV) applications, e.g., autonomous tracking and chasing a moving target. The first main approach in this novel algorithm is the use of a global matching and local tracking approach. In other words, the algorithm initially finds feature correspondences in a way that an improved binary descriptor is developed for global feature matching and an iterative Lucas–Kanade optical flow algorithm is employed for local feature tracking. The second main module is the use of an efficient local geometric filter (LGF), which handles outlier feature correspondences based on a new forward-backward pairwise dissimilarity measure, thereby maintaining pairwise geometric consistency. In the proposed LGF module, a hierarchical agglomerative clustering, i.e., bottom-up aggregation, is applied using an effective single-link method. The third proposed module is a heuristic local outlier factor (to the best of our knowledge, it is utilized for the first time to deal with outlier features in a visual tracking application), which further maximizes the representation of the target object in which we formulate outlier feature detection as a binary classification problem with the output features of the LGF module. Extensive UAV flight experiments show that the proposed visual tracker achieves real-time frame rates of more than thirty-five frames per second on an i7 processor with 640×512 image resolution and outperforms the most popular state-of-the-art trackers favorably in terms of robustness, efficiency and accuracy.

Keywords: unmanned aerial vehicle; visual object tracking; reliable global-local model; local geometric filter; local outlier factor; robust real-time performance

1. Introduction

Visual tracking, as one of the most active vision-based research topics, can assist unmanned aerial vehicles (UAVs) to achieve autonomous flights in different types of civilian applications, e.g., infrastructure inspection [1], person following [2] and aircraft avoidance [3]. Although numerous visual tracking algorithms have recently been proposed in the computer vision community [4–9], onboard visual tracking of freewill arbitrary 2D or 3D objects for UAVs remains as a challenging task due to object appearance changes caused by a number of situations, inter alia, shape deformation, occlusion, various surrounding illumination, in-plane or out-of-plane rotation, large pose variation, onboard mechanical vibration, wind disturbance and aggressive UAV flight.

To track an arbitrary object with a UAV, the following four basic requirements should be considered to implement an onboard visual tracking algorithm: (1) real-time: the tracking algorithm

must process onboard captured live image frames at high speed; (2) accuracy: the tracking algorithm must track the object accurately even with the existence of the aforementioned challenging factors; (3) adaptation: the tracking algorithm must adapt real object appearance online; (4) recovery: the tracking algorithm must be capable of re-detecting the object when the target object becomes visible in the field of view (FOV) of the camera again after the object is lost.

In this paper, the following three main modules are proposed to have a reliable visual object model under the aforementioned challenging situations and to achieve those basic requirements in different UAV tracking applications:

- A global matching and local tracking (GMLT) approach has been developed to initially find the FAST [10] feature correspondences, i.e., an improved version of the BRIEF descriptor [11] is developed for global feature matching, and an iterative Lucas–Kanade optical flow algorithm [12] is employed for local feature tracking between two onboard captured consecutive image frames based on a forward-backward consistency evaluation method [13].
- An efficient local geometric filter (LGF) module has been designed for the proposed visual feature-based tracker to detect outliers from global and local feature correspondences, i.e., a novel forward-backward pairwise dissimilarity measure has been developed and utilized in a hierarchical agglomerative clustering (HAC) approach [14] to exclude outliers using an effective single-link approach.
- A heuristic local outlier factor (LOF) [15] module has been implemented for the first time to further remove outliers, thereby representing the target object in vision-based UAV tracking applications reliably. The LOF module can efficiently solve the chaining phenomenon generated from the LGF module, i.e., a chain of features is stretched out with long distances regardless of the overall shape of the object, and the matching confusion problem caused by the multiple moving parts of objects.

Extensive UAV flight experiments show that the proposed visual tracker achieves real-time frame rates of more than thirty-five frames per second on an i7 processor with 640×512 image resolution and outperforms the most popular state-of-the-art trackers favorably in terms of robustness, efficiency and accuracy.

The outline of the paper is organized as follows: Section 2 presents the recent works related to the visual object tracking for UAVs. Section 3 introduces the proposed novel visual object tracking algorithm. The performance evaluations in various UAV flight tests and its comparisons with the most popular state-of-the-art visual trackers are discussed in Section 4. Finally, the concluding remarks are given in Section 5.

2. Related Works

2.1. Color Information-Based Method

Color information on the image frame has played an important role in visual tracking applications. A color-based visual tracker is proposed in [16] for UAVs to autonomously chase a moving red car. A visual tracking approach based on the color information is developed in [17] for UAVs to follow a red 3D flying object. Similarly, a color-based detection approach is employed in [18] for UAVs to track a red hemispherical airbag and to achieve autonomous landing. Although all of these color-based visual tracking approaches are very efficient and various kinds of color spaces can be adopted, this type of visual tracker is very sensitive to illumination changes and noise on the image, and it is preferably applicable for target tracking with a monotone and distinctive color.

2.2. Direct or Feature-Based Approach

A static or moving object tracked by a UAV has often been represented by a rectangle bounding box. Template matching (TM) has usually been applied to visual object tracking tasks for UAVs. It searches a region of interest (ROI) in the current image frame that is similar to the template defined in the first image frame. The TM approach for UAVs can be categorized into two groups:

the direct method [19] and featured-based approach [20–23]. The direct method uses the intensity information of pixels directly to represent the tracked object and to track the target object, whereas the feature-based approach adopts a visual feature, e.g., Harris corner [24], SIFT [25], SURF [26] or ORB [27] feature, to track the target object. However, these existing visual trackers are not robust to the aforementioned challenging situations since the object appearance is only defined or fixed in the first image frame, i.e., they cannot update the object appearance during the UAV operation. What is more, these trackers are not suitable for tracking 3D or deformable objects.

2.3. Machine Learning-Based Method

Machine learning methods have also been applied to vision-based UAV tracking applications. In general, these approaches can be divided into two categories based on the learning methods: offline and online learning approaches.

2.3.1. Offline Machine Learning-Based Approach

An offline-trained face detector is utilized in [28] to detect the face of a poacher from a UAV in order to protect the wildlife in Africa. An offline-learned visual algorithm is applied in [29] to detect a 2D planar target object for a UAV to realize autonomous landing. A supervised learning approach is presented in [30] to detect and classify different types of electric towers for UAVs. However, a large number of image training datasets, i.e., positive and negative image patches with all aforementioned challenging conditions, should be cropped and collected to train these trackers, thereby guaranteeing their high detection accuracies. Moreover, labeling the image training datasets requires much experience, and it is a tedious, costly and time-consuming task. In addition, an offline-trained visual tracker is only capable of tracking specifically-trained target objects instead of freewill arbitrary objects.

2.3.2. Online Machine Learning-Based Method

Recently, online learning visual trackers have been developed as the most promising tracking approaches to track an arbitrary 2D or 3D object. Online learning visual tracking algorithms are generally divided into two categories: generative and discriminative methods.

Generative approaches learn only a 2D or 3D tracking object online without considering the background information around the tracking object and then apply the online-learned model for searching the ROI on the current image frame with minimum reconstruction error. Visual tracking algorithms based on incremental subspace learning [31] and hierarchical methods are developed in [28,32] to track a 2D or 3D object for UAVs. Although these works obtained promising tracking results, a number of object samples from consecutive image frames should be collected, and they have assumed that the appearance of the target object does not change significantly during the image collection period.

Discriminative methods treat the tracking problem as a binary classification task to separate the object from its background using an online updating classifier with positive and negative (i.e., background) information. A tracking-learning-detection (TLD) [8] approach is utilized in [33] for a UAV to track different objects. A real-time adaptive visual tracking algorithm, which is based on a vision-based compressive tracking approach [6], is developed in [1] for UAVs to track arbitrary 2D or 3D objects. A structured output tracking with kernels (STRUCK) algorithm is adopted in [2] for a UAV to follow a walking person. However, updating on consecutive image frames is prone to include noises, and the drift problem is likely to occur, thereby resulting in tracking failure. Although an online multiple-instance learning (MIL) [4] approach is developed in [3] to improve tracking performance for UAVs, the update step may still not effectively eliminate noises. Additionally, most of the discriminative methods cannot estimate the scale changes of the target object.

3. Proposed Method

The proposed method in this paper mainly includes three modules: (1) global matching and local tracking (GMLT); (2) local geometric filter (LGF); and (3) local outlier factor (LOF).

3.1. Global Matching and Local Tracking Module

Let b_1 be a bounding box around an online selected 2D or 3D target object, e.g., a moving person on the first image frame I_1 shown in Figure 1. The FAST features on each RGB image frame are detected using a bucketing approach [34], i.e., the captured image frame is separated into pre-defined non-overlapped rectangle regions, i.e., buckets. The bucketing approach keeps the FAST features evenly distributed in the image frame and guarantees the real-time visual tracking performance. A group of the FAST features detected in b_1 is denoted as $\{x_1^1, x_1^2, ..., x_1^n\}$, where $x_1^i \in \mathbb{R}^2, i = 1, 2, ..., n$; they compose the global model $\mathcal{M}_g$ of the target object. The model $\mathcal{M}_g$ is utilized to globally match the candidate FAST features detected on each image frame with the improved version of the BRIEF descriptor. It is to be noted that the global matching is able to achieve the re-detection of the object when the target object becomes visible in the camera FOV again after the object is lost. For the i-th FAST feature on the k-th image frame x_k^i, its improved BRIEF descriptor, i.e., $\mathcal{B}(x_k^i) = \{B_1(x_k^i), B_2(x_k^i), ..., B_{N_b}(x_k^i)\}$, is defined as follows:

$$B_j(x_k^i) = \begin{cases} 1 & \text{if } I_k(x_k^i + p_j) < I_k(x_k^i + q_j) \\ 0 & \text{otherwise} \end{cases}, \forall j \in [1, ..., N_b] \quad (1)$$

where $B_j(x_k^i)$ is the j-th bit of the binary vector in $\mathcal{B}(x_k^i)$, $I_k(*)$ is the intensity of the pixel on the k-th image frame and (p_j, q_j) is sampled in a local neighbor region $S_r \times S_r$ based on the location of the i-th FAST feature. $p_j = \mathcal{N}(0, (\frac{1}{5}S_r)^2)$ and $q_j = \mathcal{N}(p_j, (\frac{2}{25}S_r)^2)$. If the intensity of the pixel on the location $x_k^i + p_j$ is smaller than the one on the location $x_k^i + q_j$, then the $B_j(x_k^i)$ is one, otherwise, it is zero. The parameter N_b is the length of the binary vector $\mathcal{B}(x_k^i)$, i.e., the number of comparisons to perform. It is to be noted that the distance d of two binary vectors is computed by counting the number of different bits between these two vectors, i.e., the Hamming distance, which has less computation cost compared to the Euclidean distance. For searching the FAST feature correspondences on the k-th ($k \geq 2$) image frame using the model $\mathcal{M}_g$, the FAST feature that has the lowest Hamming distance d_1 is the best feature candidate; the FAST feature that has the second-lowest Hamming distance d_2 is the second-best feature candidate; when the ratio between the best and second-best match d_1/d_2 is less than a threshold ρ, the best FAST feature candidate is accepted as a matched FAST feature.

An iterative Lucas–Kanade optical flow algorithm with a three-level pyramid has been utilized to track each FAST feature between the (k-1)-th image frame and the k-th image frame based on the forward-backward consistency evaluation approach within a $S_h \times S_h$ local search window. These tracked FAST features constitute the local model $\mathcal{M}_l$ of the target object. It is to be noted that in the local tracking stage, the model $\mathcal{M}_l$ is updated frame-by-frame, i.e., the model $\mathcal{M}_l$ is adaptive.

Let $\mathcal{F}_1$ be the matched and tracked FAST features on the k-th image frame, denoted as $\mathcal{F}_1 = \{x_k^1, x_k^2, ..., x_k^m\}$, where $x_k^i \in \mathbb{R}^2$, $i = 1, 2, ..., m$. In this work, the scale s_k of the target object is estimated based on [13], i.e., for each pair of FAST features, a ratio between the FAST feature distance on the current image frame and the corresponding FAST feature distance on the first image frame is calculated:

$$s_k^{ij} = \frac{||x_k^i - x_k^j||}{||x_1^i - x_1^j||}, i \neq j \quad (2)$$

and then, the median of $\{s_k^{ij}\}$ is the estimated scale s_k of the target object, since it is more robust with respect to outliers.

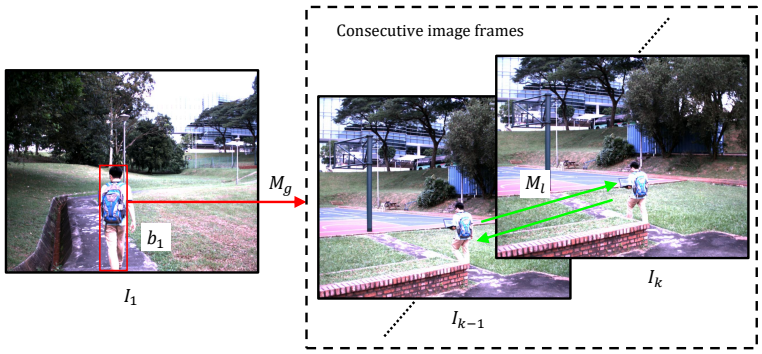


Figure 1. Illustration of the global matching and local tracking (GMLT) module. The bounding box b_1 is shown with the red rectangle. The FAST features detected in the b_1 compose the global object model $\mathcal{M}_g$, which is employed to globally find the feature correspondences on onboard captured consecutive image frames with the improved BRIEF descriptor. The green arrow is the Lucas–Kanade optical flow algorithm-based tracking between the (k-1) -th frame and the k-th frame. $\mathcal{M}_l$ is the local object model, which is updated frame-by-frame.

In our extensive visual tracking tests, we find that the $\mathcal{F}_1$ includes certain outlier FAST features, as one example is shown in the left image in Figure 2, i.e., I_k^{GMLT}. Let a matching with the model $\mathcal{M}_g$ be a global FAST feature correspondence, a tracking with model $\mathcal{M}_l$ be a local FAST feature correspondence and the combination of the global matching and local tracking correspondences be $\mathcal{C}_k^{\cup}$; mt_k^i is the i-th FAST feature correspondence in $\mathcal{C}_k^{\cup}$. The next subsection introduces the second main module to detect these outliers.

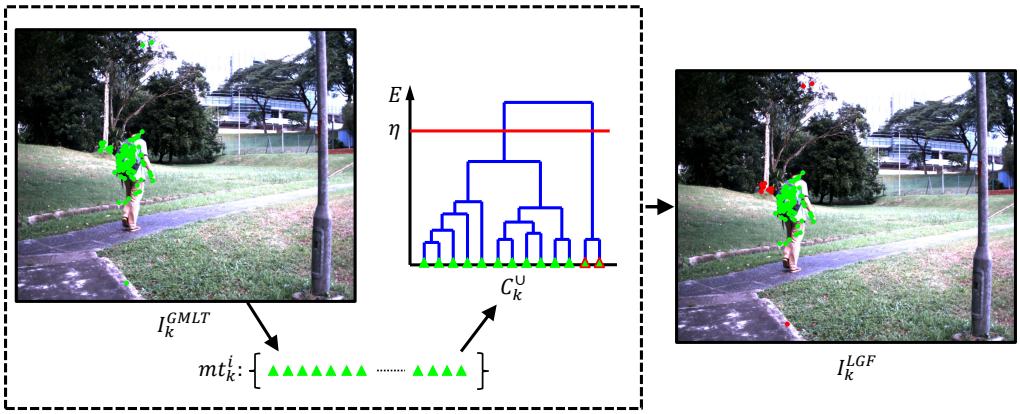


Figure 2. Illustration of the local geometric filter (LGF) module. The green points on the I_k^{GMLT} are the matched and tracked FAST features from the GMLT module, i.e., $\mathcal{F}_1$. The green triangles are the FAST feature correspondences, i.e., $\mathcal{C}_k^{\cup}$. The LGF module is utilized to filter outlier correspondences, as the green triangles with red edges shown in the dendrogram. The red points on the I_k^{LGF} are the outliers filtered by the LGF module.

3.2. Local Geometric Filter Module

The second main module in the proposed method is a novel efficient local geometric filter (LGF), which utilizes a new forward-backward pairwise dissimilarity measure E_{LGF} between correspondences mt_k^i and mt_k^j based on pairwise geometric consistency, as illustrated in Figure 3.

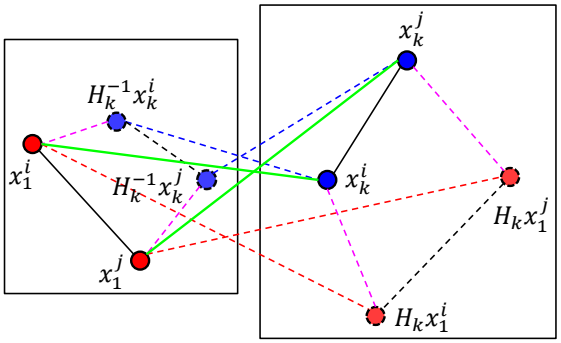


Figure 3. Forward-backward pairwise dissimilarity measure E_{LGF} between correspondence mt_k^i and mt_k^j (green solid lines). The dashed red and blue points are transformed by the homography H_k and the inversion of homography H_k^{-1}.

The E_{LGF} for every pair of correspondences is defined as below:

$$E_{\mathrm{LGF}}(mt_k^i, mt_k^j) = \frac{1}{2}\left[E_{\mathrm{LGF}}(mt_k^i, mt_k^j|H_k) + E_{\mathrm{LGF}}(mt_k^i, mt_k^j|H_k^{-1})\right], i \neq j \tag{3}$$

where:

$$E_{\mathrm{LGF}}(mt_k^i, mt_k^j|H_k) = \left\|(x_k^i - x_k^j) - H_k(x_1^i - x_1^j)\right\|, i \neq j$$

$$E_{\mathrm{LGF}}(mt_k^i, mt_k^j|H_k^{-1}) = \left\|(x_1^i - x_1^j) - H_k^{-1}(x_k^i - x_k^j)\right\|, i \neq j$$

$\|*\|$ is the Euclidean distance; H_k is a homography transformation [35] estimated by the $\mathcal{C}_k^{\cup}$; and H_k^{-1} is the inversion of this homography transformation.

To reduce the ambiguity correspondences and filter the erroneous correspondences, a hierarchical agglomerative clustering (HAC) approach [14] is utilized to separate outlier correspondences from inliers based on an effective single-link approach with the forward-backward pairwise dissimilarity measure E_{LGF}. Let $S(G, G')$ be a cluster dissimilarity for all pairs of clusters, $G, G' \subset \mathcal{C}_k^{\cup}$; the single-link HAC algorithm is defined:

$$S(G, G') = \min_{mt_k^i \in G, mt_k^j \in G'} E_{\mathrm{LGF}}(mt_k^i, mt_k^j) \tag{4}$$

It defines the cluster dissimilarity S as the minimum among all of the forward-backward pairwise dissimilarities between the two correspondences of the two clusters. A dendrogram generated from the single-link HAC approach is shown in the middle of Figure 2; the $\mathcal{C}_k^{\cup}$ is divided into some subgroups based on a cut-off threshold η, and the biggest subgroup is considered as the correspondences for the target object. The green points shown on the right side of Figure 2, i.e., I_k^{LGF}, are the FAST features output from the LGF module, denoted as $\mathcal{F}_2 = \{\bar{x}_k^1, \bar{x}_k^2, ..., \bar{x}_k^w\}$, where $\bar{x}_k^i \in \mathbb{R}^2$, $i = 1, 2, ..., w$, while the red points are the outliers.

The bottom-up aggregation in the single link-based clustering method is strictly local. The single-link HAC approach is easy to generate the chaining phenomenon, i.e., a chain of correspondences is stretched out for long distances without considering the real shape of the target object, especially in cluttered environments, leading to inefficient exclusion of the outliers. Additionally, multiple parts of objects, e.g., moving hands, are prone to confuse the FAST feature matching in the next new image frame. In this work, the local outlier factor (LOF) is developed to handle the chaining and confusion problems efficiently. The following subsection introduces the third main module.

3.3. Local Outlier Factor Module

The third main module of this work is a heuristic local outlier factor (LOF) [15], which is developed for the first time in a visual tracking application to further remove outliers, thereby maximizing target object representation and solving the matching confusion problem. The LOF is based on local density, i.e., the outlier is considered when its surrounding space contains relatively few FAST features.

As shown in Figure 4, the local density of the FAST feature $\bar{x}_k^i$ is compared to the densities of its neighborhood FAST features. In this case, if the FAST feature $\bar{x}_k^i$ has much lower density than its neighbors, then it is an outlier.

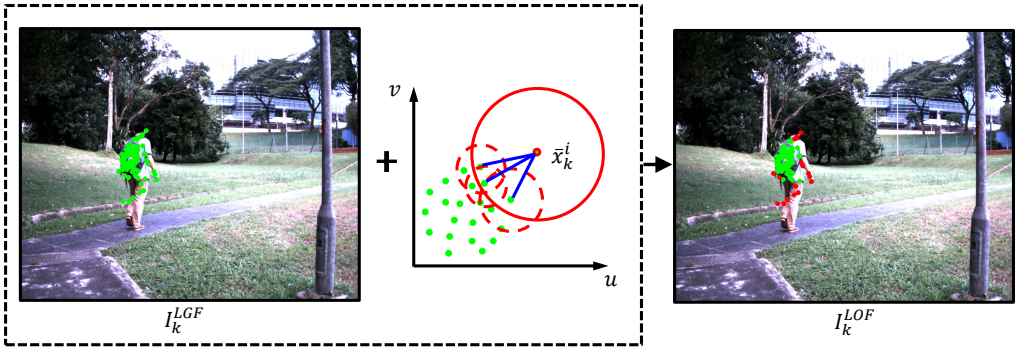


Figure 4. Illustration of local outlier factor (LOF) module. The green points on the I_k^{LGF} are the FAST features from the LGF module, i.e., $\mathcal{F}_2$. The LOF module is developed to further remove the outliers, as the red points shown on the I_k^{LOF}. The green points on the I_k^{LOF} are final reliable output FAST features in our visual tracking application.

In this work, we formulate outlier feature detection as a binary classification problem. The binary classifier is defined as follows:

$$f(\bar{x}_k^i) = \begin{cases} \text{target object}, & E_{\text{LOF}}(\bar{x}_k^i) \leq \mu \\ \text{outlier}, & E_{\text{LOF}}(\bar{x}_k^i) > \mu \end{cases} \tag{5}$$

where $E_{\text{LOF}}(\bar{x}_k^i)$ is a density dissimilarity measure of FAST feature $\bar{x}_k^i$, $\bar{x}_k^i \in \mathcal{F}_2$, and μ is a cut-off threshold to classify that a FAST feature belongs to the target object or an outlier. If the value of the $E_{\text{LOF}}(\bar{x}_k^i)$ is larger than μ, then $\bar{x}_k^i$ is the outlier, otherwise, $\bar{x}_k^i$ belongs to the target object.

The LOF module includes three steps to calculate the $E_{\text{LOF}}(\bar{x}_k^i)$:

- Construction of the nearest neighbors: the nearest neighbors of the FAST feature $\bar{x}_k^i$ are defined as follows:
$$\mathcal{NN}(\bar{x}_k^i) = \{\bar{x}_k^j \in \mathcal{F}_2 \setminus \{\bar{x}_k^i\} | D(\bar{x}_k^i, \bar{x}_k^j) \leq R_t(\bar{x}_k^i)\} \tag{6}$$
where $D(\bar{x}_k^i, \bar{x}_k^j)$ is the Euclidean distance between the FAST features $\bar{x}_k^i$ and $\bar{x}_k^j$. $R_t(\bar{x}_k^i)$ is the Euclidean distance from $\bar{x}_k^i$ to the t-th nearest FAST feature neighbor.
- Estimation of neighborhood density: the neighborhood density δ of the FAST feature $\bar{x}_k^i$ is defined as:
$$\delta(\bar{x}_k^i) = \frac{|\mathcal{NN}(\bar{x}_k^i)|}{\sum_{\bar{x}_k^j \in \mathcal{NN}(\bar{x}_k^i)} \max\{R_t(\bar{x}_k^j),\ D(\bar{x}_k^i, \bar{x}_k^j)\}} \tag{7}$$
where $|\mathcal{NN}(\bar{x}_k^i)|$ is the nearest neighbor number of $\bar{x}_k^i$.

- Comparison of neighborhood densities: the comparison of neighborhood densities results in the density dissimilarity measure $E_{\mathrm{LOF}}(\bar{x}_k^i)$, which is defined below:

$$E_{\mathrm{LOF}}(\bar{x}_k^i) = \frac{\sum_{\bar{x}_k^j \in \mathcal{NN}(\bar{x}_k^i)} \frac{\delta(\bar{x}_k^j)}{\delta(\bar{x}_k^i)}}{|\mathcal{NN}(\bar{x}_k^i)|} \tag{8}$$

Figure 4 shows the illustration of the LOF module; the green points on the I_k^{LOF} are final reliable output FAST features for our visual tracking application, denoted as $\mathcal{F}_3 = \{\hat{x}_k^1, \hat{x}_k^2, ..., \hat{x}_k^o\}$, where $\hat{x}_k^i \in \mathbb{R}^2$, $i = 1, 2, ..., o$. The FAST features in $\mathcal{F}_3$ and their corresponding features in b_1 compose final FAST feature correspondences $\hat{\mathcal{C}}_k^{\cup}$. Then, the center c_k of the target object is calculated as follows:

$$c_k = \frac{\sum_{mt_k^i \in \hat{\mathcal{C}}_k^{\cup}} (\hat{x}_k^i - Hx_1^i)}{|\hat{\mathcal{C}}_k^{\cup}|} \tag{9}$$

4. Real Flight Tests and Comparisons

In the UAV flight experiments, a Y6 coaxial tricopter UAV equipped with a Pixhawk autopilot from 3D Robotics is employed; the onboard computer is an Intel NUC Kit NUC5i7RYH Mini PC, which has a Core i7-5557U processor with dual-core, 16 GB RAM and a 250-GB SATA SSD drive. Both forward- and downward-looking cameras are USB 3.0 RGB cameras from Point Grey, i.e., Flea3 FL3-U3-13E4C-C, which capture the image frames with a resolution of 640 × 512 at 30 Hz. The whole UAV system is shown in Figure 5.

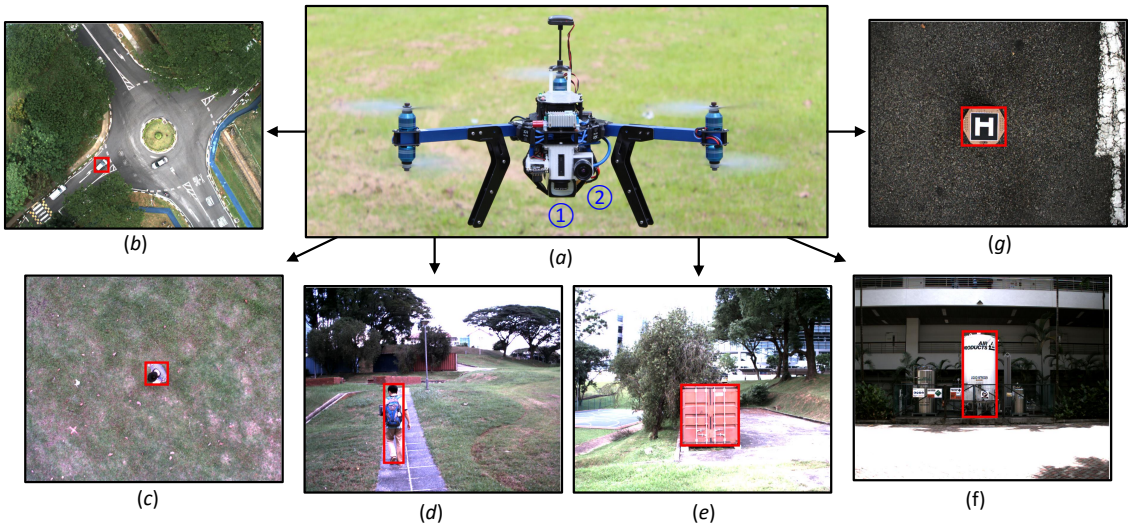


Figure 5. Robust real-time accurate long-term visual object tracking onboard our Y6 coaxial tricopter UAV. No. 1 and 2 in (**a**) show downward- and forward-looking monocular RGB cameras. Some 2D or 3D objects with their tracking results are shown in (**b**–**g**).

To practically test and evaluate the robustness, efficiency and accuracy of the proposed onboard visual tracker, we have developed our visual tracker in C++ and conducted more than fifty UAV flights in various types of environments of Nanyang Technological University, including challenging situations. As shown in Figure 5, target objects include a moving car (b), walking people (c and d), a container (e), a gas tank (f) and a moving unmanned ground vehicle (UGV) with a landing pad (g). In this paper, six recorded image sequences are randomly selected which contain 11,646 image frames, and manually labeled for the ground truth. The challenging factors of the each image sequence are listed in Table 1.

Table 1. Challenging factors of each image sequence. MV: mechanical vibration; AF: aggressive flight; IV: illumination variation; OC: partial or full occlusion; SV: scale variation; DE: deformation, i.e., non-rigid object deformation; IR: in-plane rotation; OR: out-of-plane rotation; OV: out-of-view; CB: cluttered background. The total number of evaluated image frames in this paper is 11,646.

Sequence	Number	MV	AF	IV	OC	SV	DE	IR	OR	OV	CB
Container	2874	√				√			√	√	√
Gas tank	3869	√	√	√		√		√	√	√	√
Moving car	582	√						√		√	
UGV_{lp}	1325	√			√			√		√	
$People_{bw}$	934	√					√	√	√		
$People_{fw}$	2062	√		√	√	√	√		√		√

To compare our proposed visual tracker, we have employed the most popular state-of-the-art visual trackers, e.g., MIL [4], STRUCK [5], CT [6], Frag [7], TLD [8] and KCF [9], which have adaptive capabilities for appearance changes of the target objects and have been utilized to achieve the real UAV tracking applications. For all of these state-of-the-art trackers, we have utilized the source or binary programs provided by the authors with default parameters. In our proposed visual tracker, the main parameters are defined in Table 2 below. In addition, all visual trackers are initialized with the same parameters, e.g., initial object location.

Table 2. Main parameters in our presented visual tracker.

Parameter Name	Value	Parameter Name	Value
Bucketing configuration	10×8	FAST threshold	20
Sampling patch size (S_r)	48	BRIEF descriptor length (N_b)	256
Ratio threshold (ρ)	0.85	Local search window (S_h)	30
LGF cut-off threshold (η)	18	LOF cut-off threshold (μ)	1.5

In this work, the center location error (CLE) of the tracked target object has been utilized to evaluate all visual trackers. It has been defined as the Euclidean distance between the estimated target object center and the manually-labeled ground truth center on each image frame, i.e.:

$$\text{CLE} = \left\| O_k^E - O_k^{GT} \right\| \tag{10}$$

where O_k^E and O_k^{GT} are the estimated and ground truth centers of the target object. Figures 6–11 show the CLE evolutions of all visual trackers in different image sequences. Specifically, we note that the TLD tracker easily loses the target completely for certain image frames when the target object is still in the FOV of the onboard camera; since it is able to re-detect the target object, we show the CLE error for the image sequence that the TLD tracker can track more than 96% of frames as a reference. Table 3 shows the CLE errors of all visual trackers. To visualize the tracking precisions of all visual trackers, Figures 12a, 13a, 14a, 15a, 16a and 17a show the precision plots of all image sequences.

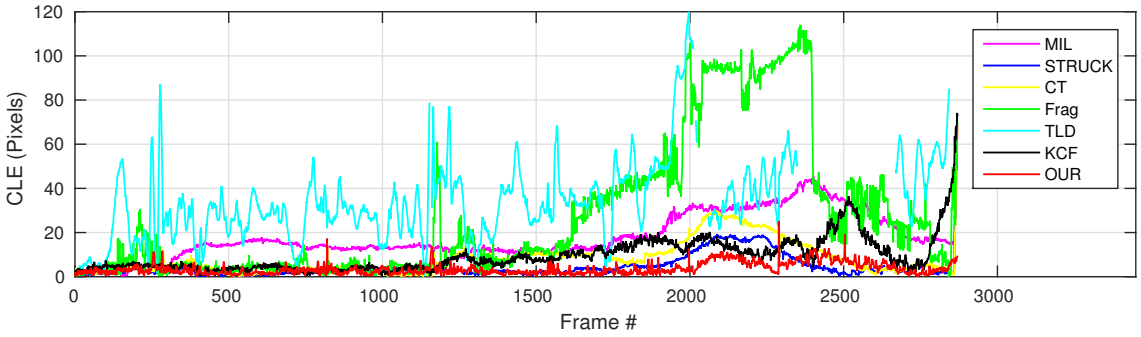


Figure 6. Center location error (CLE) error evolution plot of all visual trackers with the container image sequence.

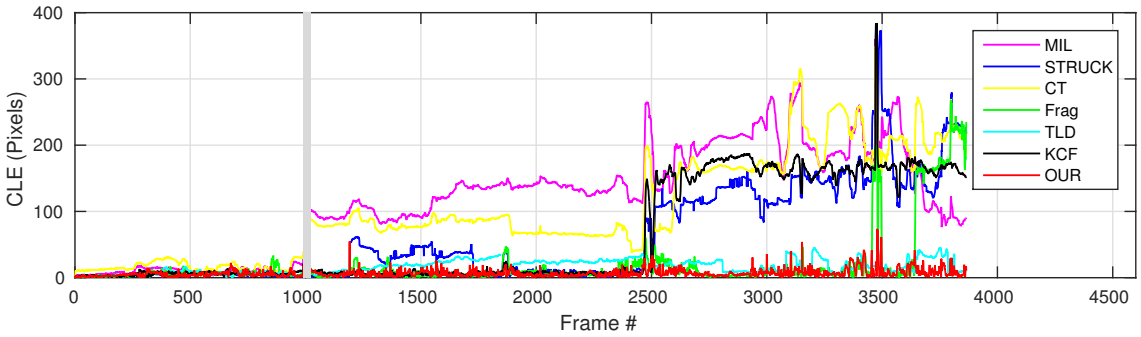


Figure 7. CLE evolution plot of all visual trackers with gas tank image sequence. The grey area represents that the whole target object is out of view.

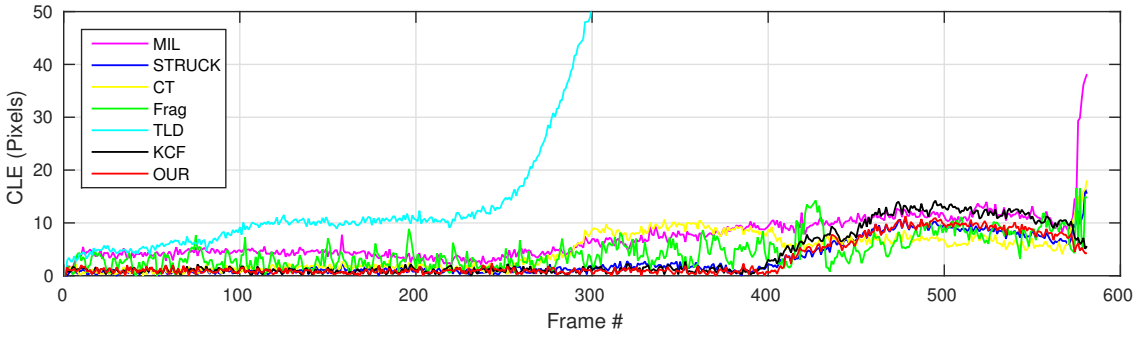


Figure 8. CLE error evolution plot of all trackers with the moving car image sequence.

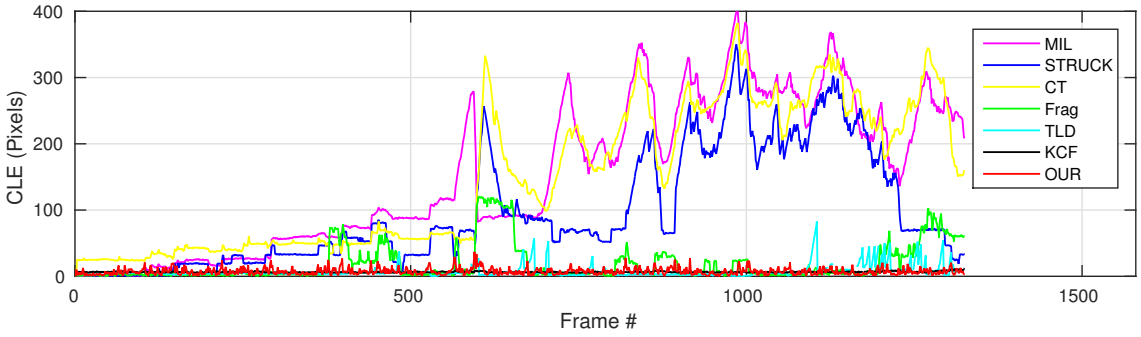


Figure 9. CLE error evolution plot of all trackers with the UGV_{lp} image sequence.

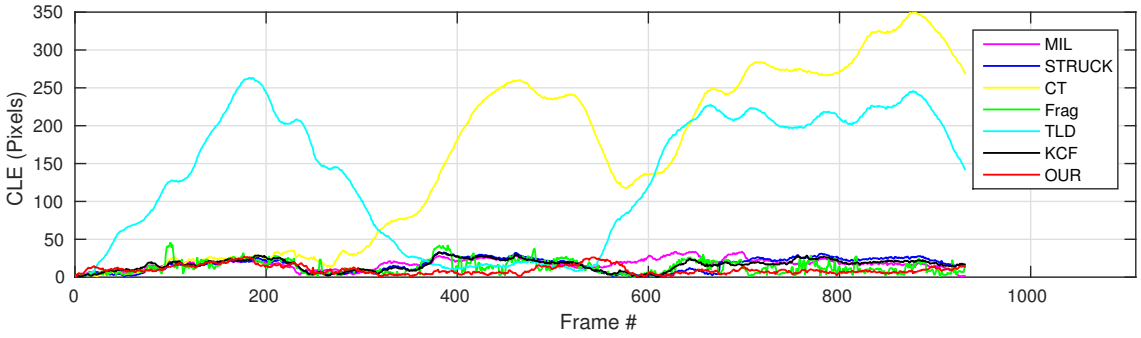


Figure 10. CLE error evolution plot of all trackers with the People$_{bw}$ image sequence.

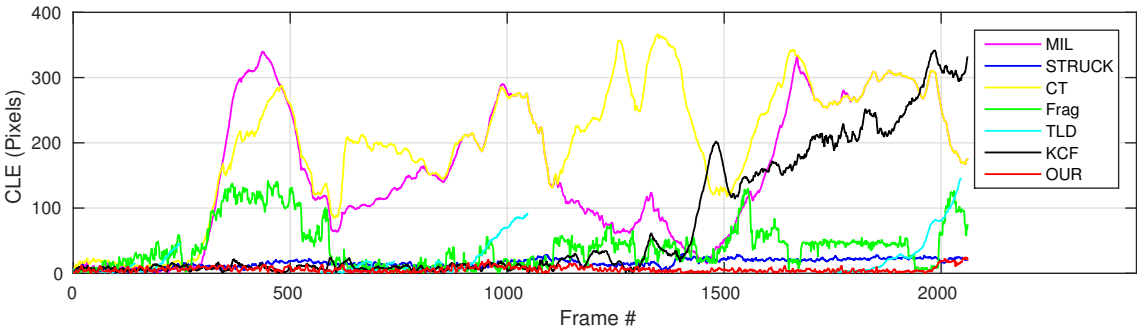
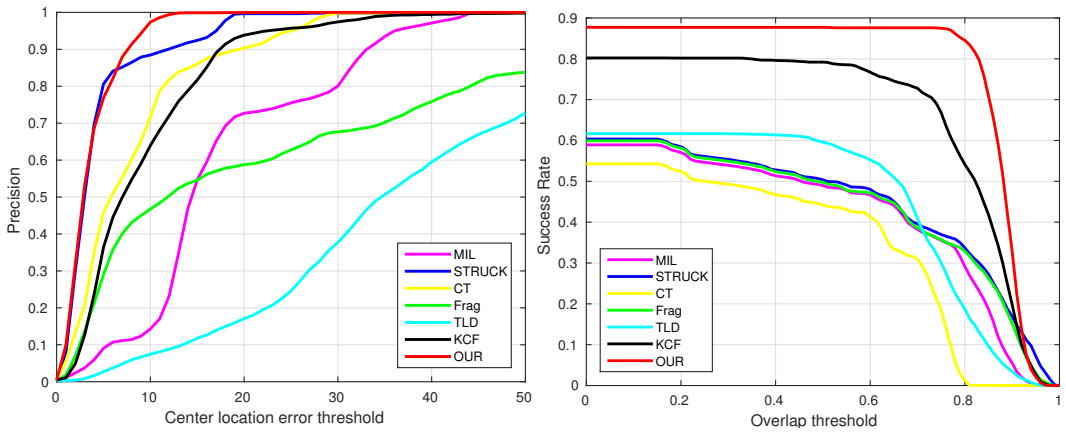


Figure 11. CLE error evolution plot of all trackers with the People$_{fw}$ image sequence.

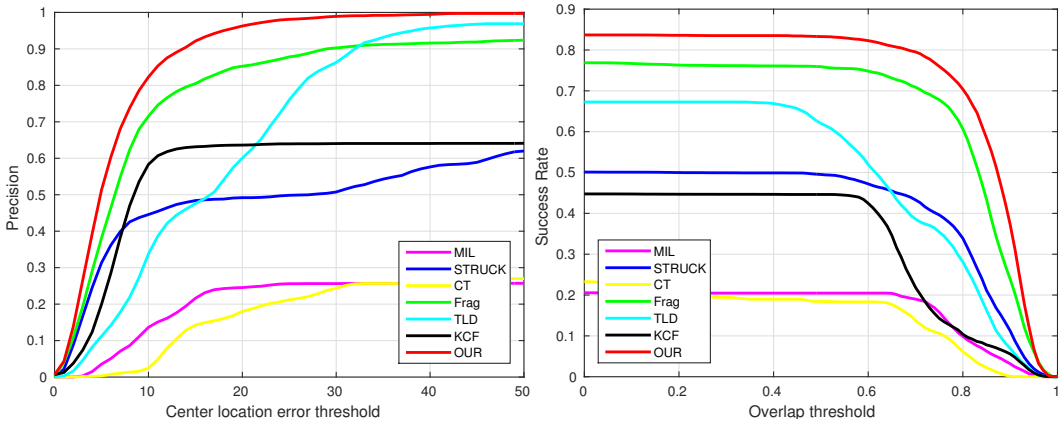
Table 3. Center location error (CLE) (in pixels) and frames per second (FPS). Red, blue and green fonts indicate the best, second best and third best performances in all visual trackers. The total number of evaluated image frames in this paper is 11,646.

Sequence	MIL	STRUCK	CT	Frag	TLD	KCF	Our
Container	17.7	**4.5**	**8.2**	27.4	-	9.3	**3.6**
Gas tank	118.1	62.4	103.4	**22.7**	**16.6**	63.1	**6.8**
Moving Car	7.1	**3.1**	4.5	4.5	105.1	**3.7**	**3.3**
UGV_{lp}	152.2	97.1	150.1	20.6	**7.6**	**4.8**	**7.3**
$People_{bw}$	17.8	16.7	157.9	**13.8**	130.8	**16.6**	**10.2**
$People_{fw}$	153.3	**15.7**	197.1	**41.7**	-	73.0	**6.1**
CLE_{Ave}	96.4	41.6	107.1	**28.4**	-	**18.9**	**6.5**
FPS_{Ave}	24.8	16.2	**28.7**	13.1	23.9	**149.8**	**38.9**



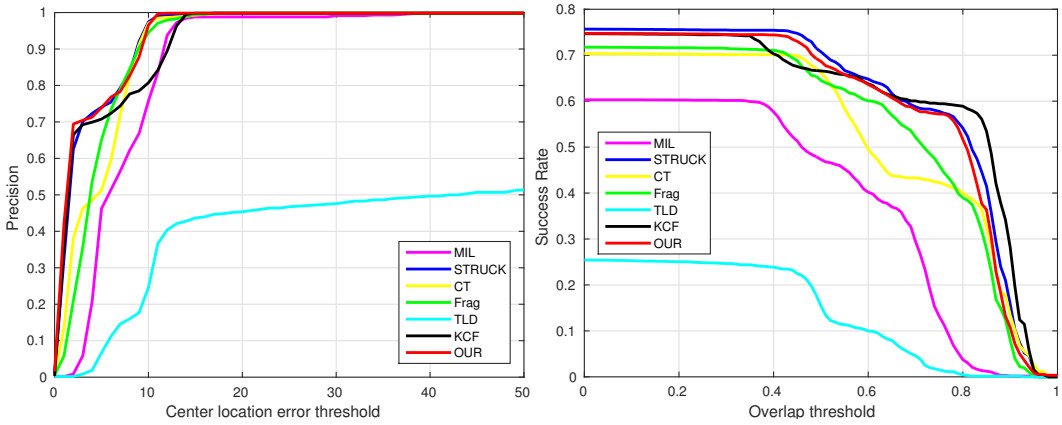
(a) Precision plot. **(b)** Success plot.

Figure 12. Precision and success plots of all visual trackers with the container image sequence.



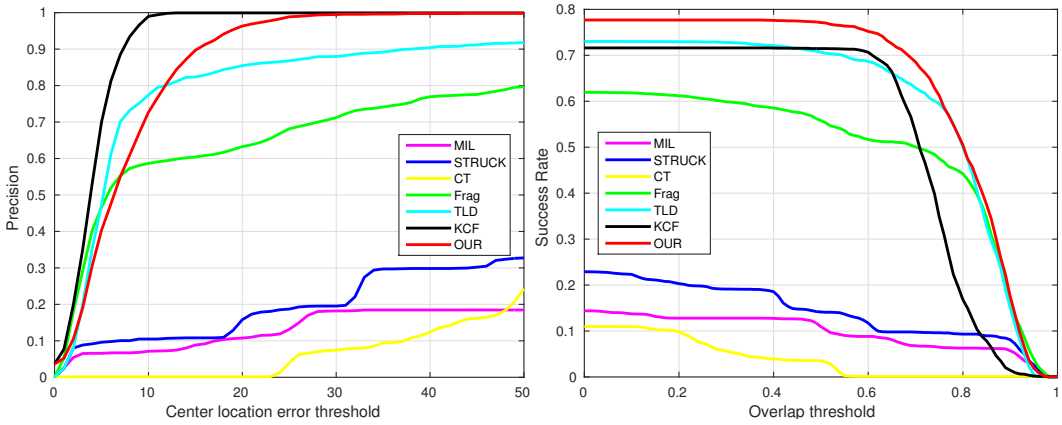
(a) Precision plot. **(b)** Success plot.

Figure 13. Precision and success plots of all visual trackers with the gas tank image sequence.



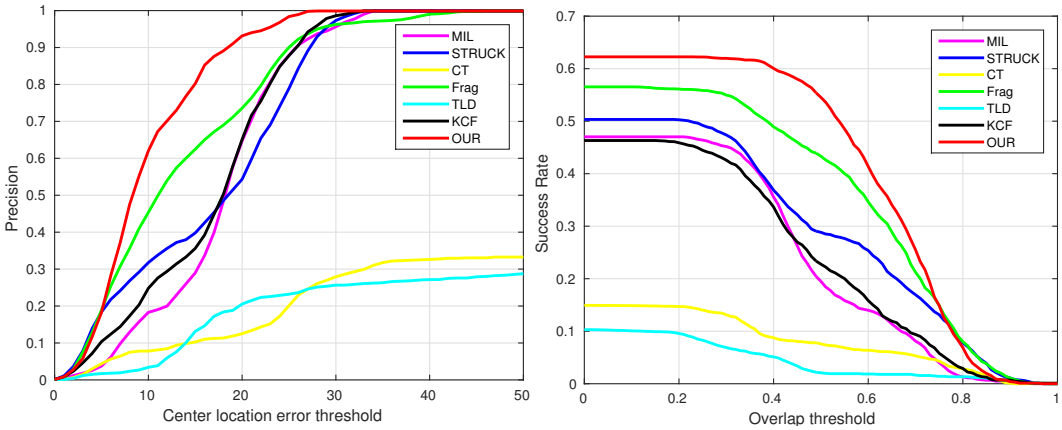
(a) Precision plot. **(b)** Success plot.

Figure 14. Precision and success plots of all trackers with the moving car image sequence.



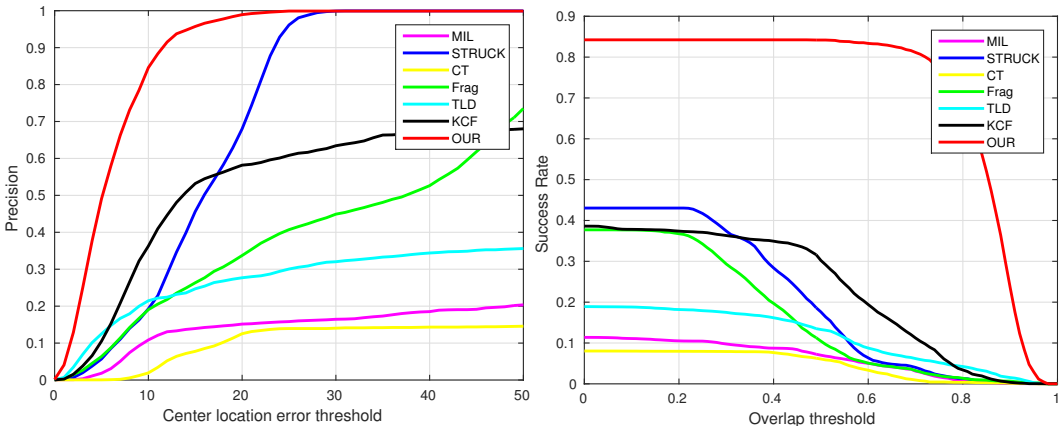
(a) Precision plot. **(b)** Success plot.

Figure 15. Precision and success plots of all trackers with the UGV_{lp} image sequence.



(a) Precision plot. **(b)** Success plot.

Figure 16. Precision and success plots of all trackers with the People$_{bw}$ image sequence.



(a) Precision plot. **(b)** Success plot.

Figure 17. Precision and success plots of all trackers with the People$_{fw}$ image sequence.

In addition, the success score (SS) has also been employed to evaluate the performances of all visual trackers in this paper, as it can evaluate the scales of the target object. The SS has been defined as below:

$$\mathrm{SS} = \frac{|ROI_E \cap ROI_{GT}|}{|ROI_E \cup ROI_{GT}|} \tag{11}$$

where ROI_E and ROI_{GT} are the estimated and ground truth sizes of the target object. $\cap$ and $\cup$ are the intersection and union operators. $|*|$ represents the number of pixels in a region. If the SS is larger than ξ in an image frame, the tracking result is considered as a success. Table 4 shows the tracking results ($\xi = 0.5$) of all visual trackers in terms of success rate, which is defined as the ratio between the number of success frames and the total number of image frames. Moreover, we have shown the area under area (AUC) of each success plot, which is defined as the average of the success rates based on the overlap thresholds, as the results shown in Figures 12b, 13b, 14b, 15b, 16b and 17b.

Table 4. Success rate (SR) (%) (ξ = 0.5). Red, blue and green fonts indicate the best, second best and third best performances in all visual trackers. The total number of evaluated image frames in this paper is 11,646.

Sequence	MIL	STRUCK	CT	Frag	TLD	KCF	Our
Container	62.9	62.7	62.7	62.5	81.2	96.6	99.8
Gas tank	25.7	61.8	24.5	90.8	85.6	62.7	97.3
Moving car	68.0	88.1	89.9	82.5	24.7	78.4	85.7
UGV_{lp}	15.1	18.7	6.8	70.3	89.9	99.6	98.6
$People_{bw}$	30.0	40.5	10.5	63.2	2.67	34.7	81.7
$People_{fw}$	10.6	29.5	9.9	16.9	19.6	46.8	99.4
SR_{Ave}	32.9	50.1	30.8	65.3	51.8	71.0	95.9

4.1. Test 1: Visual Tracking of The Container

In this test, a static container is selected as the target object for our Y6 coaxial tricopter UAV to carry out the visual tracking application, and the onboard forward-looking camera is employed to track the locations of the target object. As the challenging factors concluded in Table 1, the container image sequence includes mechanical vibration (all image frames), scale variation (e.g., Figure 18, Frames 1827 and 2000), out-of-plane rotation (e.g., Figure 18, Frames 175 and 2418), out-of-view (e.g., Figure 18, Frame 2874) and cluttered background (all image frames).

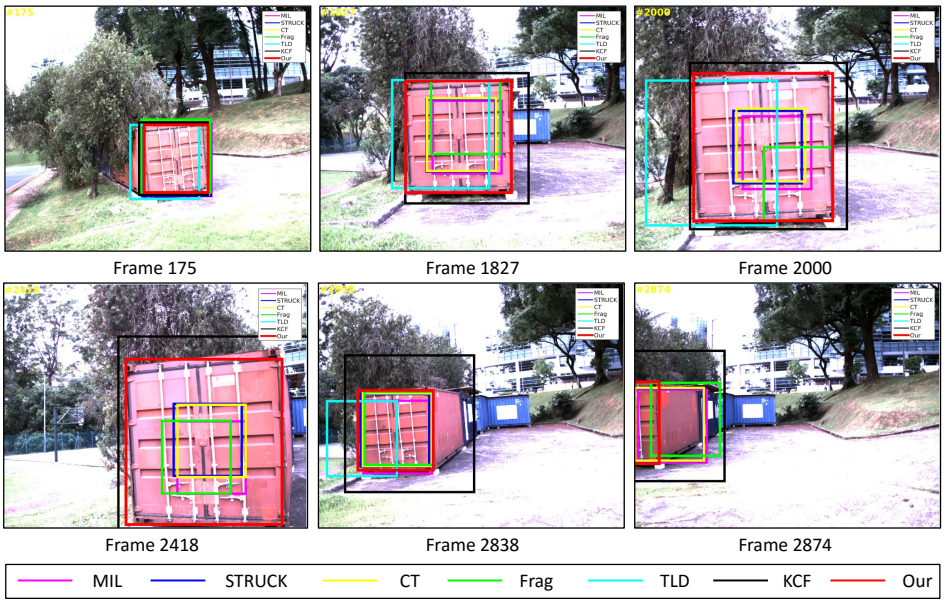


Figure 18. Some tracking results in the container image sequence. The total number of frames: 2874.

As can be seen in Figure 6, we can find that the CLE errors of our presented visual tracker (red line) and STRUCK (blue line) are always less than 20 pixels, i.e., the precisions of these two visual trackers can achieve almost one when the CLE threshold is 20 pixels, as shown in Figure 12a. Moreover, the tracking performance of the CT tracker (yellow line) is ranking as No. 3 in this image sequence; its CLEs are changing extensively when the target object is out-of-plane; the maximum CLE error of the CT tracker is 29.8 pixels. The KCF tracker is ranking No. 4; its performance is decreasing when the flying UAV is approaching the target object. Additionally, the MIL tracker (magenta line) outperforms the TLD (cyan line) and Frag (green line) trackers, and the tracking performance of Frag is better than that of the TLD tracker; it is noticed that the TLD tracker completely loses track of the target object when some portion of the target object is out of view, i.e., some parts of the target object are not shown in the FOV of the onboard forward-looking camera, as Frame 2874 shown in Figure 18. In addition, the

2874th frame also shows that our presented visual tracker is able to locate the target object accurately even under the out-of-view situation.

Figure 12b shows the average success rates of all visual trackers. Since the MIL, STRUCK, CT and Frag trackers cannot estimate the scales of the target object, their average success rates are relatively low. Conversely, the TLD, KCF and our presented visual tracker can estimate the target object scales. However, the accuracies of the TLD and KCF trackers for estimating the center locations are lower. Therefore, the average success rates of the TLD and KCF trackers are also lower than ours.

4.2. Test 2: Visual Tracking of the Gas Tank

A static gas tank is employed as the tracking object of our UAV in this test. As shown in Figure 19, this target object does not contain much texture information. Additionally, the aggressive flight (e.g., Figure 19, Frames 3343 and 3490), out-of-view (e.g., Figure 19, Frames 999, 1012 and 3490), scale variation (e.g., Figure 19, Frames 500 and 2587) and cluttered background (all of the frames) are the main challenging factors.

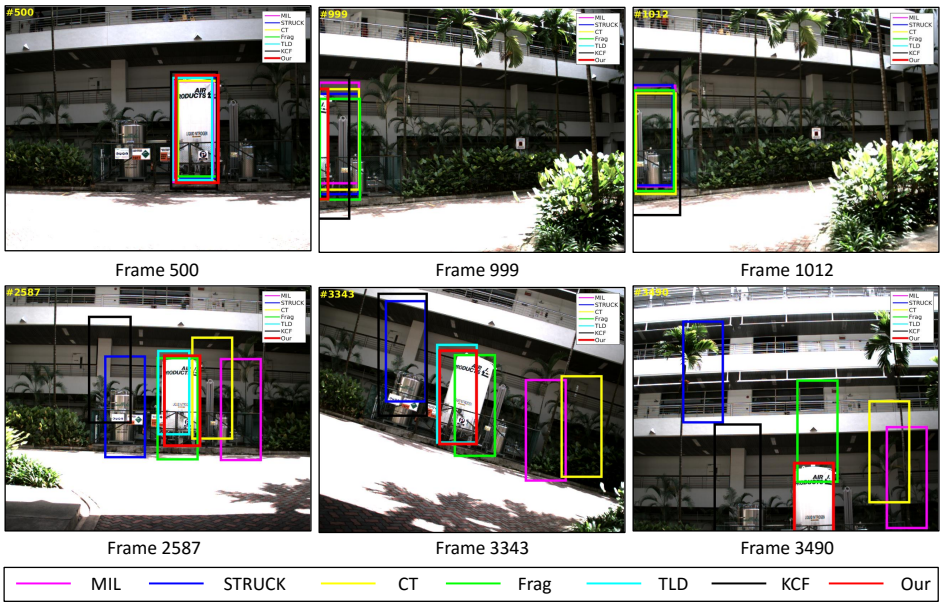


Figure 19. Some tracking results in the gas tank image sequence. The total number of frames: 3869.

As shown in Figure 7, the CLE errors of the MIL and CT trackers are relatively high after the whole target object has been out-of-view (the period is shown as the gray area). In this case, the MIL and CT trackers have completely learned new appearances for their target objects, resulting in losing the target object, which they should track. Moreover, the STRUCK tracker has some drifts from Frame 1191 because of the sudden large displacement.

From Frame 2452, our UAV has carried out the first aggressive flight. The CLE errors of the STRUCK and KCF trackers started to increase due to larger displacement, since the STRUCK and KCF trackers have also adapted to the new appearances of the target object during the first aggressive flight, leading to losing their target objects until the end of the UAV tracking task.

The second aggressive flight has started from Frame 2903. Although the movements are even larger than the ones in the first aggressive flight, the Frag and our presented trackers can locate the target object well; their CLE errors are 7.8 and 7.5 pixels, respectively. The strongest aggressive flight in this test is from Frame 3318, as Frames 3343 and 3490 shown in Figure 19; its maximum flight speed has reached 3.8 m/s. Figure 13a shows the precision plot of all visual trackers. It can be seen that our presented visual tracker has achieved 90% precision when the CLE threshold is 14 pixels. In addition, Figure 13b also shows that our presented visual tracker is ranked as No.1.

4.3. Test 3: Visual Tracking of the Moving Car

In Tests 3–6, moving target objects are selected for our UAV to conduct the vision-based tracking applications. In this test, our UAV is utilized to track one moving car from an 80 m height over a traffic intersection. The main challenging factors are mechanical vibration (all image frames), in-plane rotation (e.g., Figure 20, Frame 410), out-of-view (e.g., Figure 20, Frame 582) and similar appearances of other moving cars (all image frames).

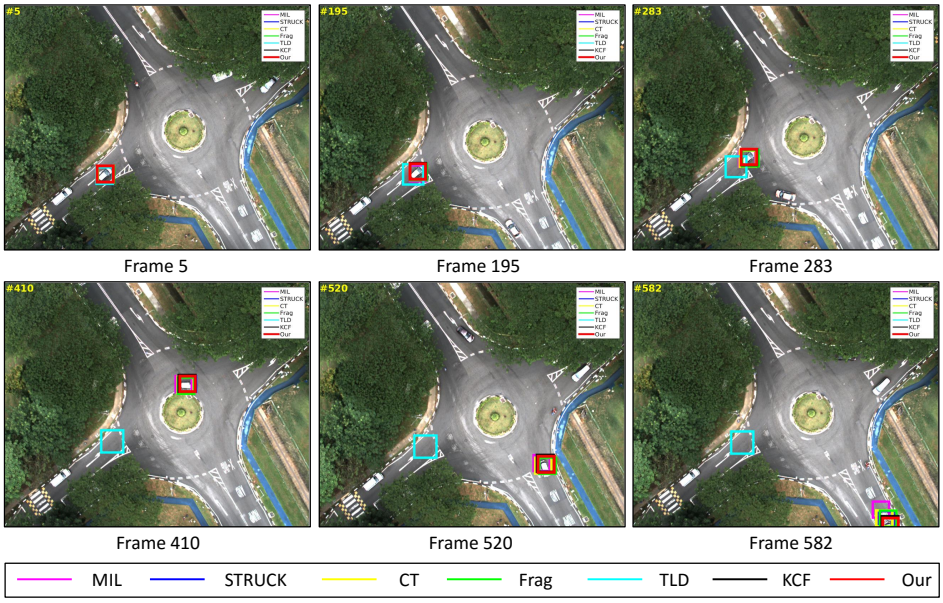


Figure 20. Some tracking results in the moving car image sequence. The total number of frames: 582.

Figure 8 shows the CLE error evolutions of all visual trackers. We can find that all of the trackers can track the target object well in all image frames, except for the TLD tracker, as also shown in Figure 14a; the CT, STRUCK, Frag and our trackers have achieved 95% precision when the CLE threshold is 10.2 pixels.

From Frame 301, the TLD tracker has lost its target object completely because of its adaptation to a new target appearance, which is similar to the background information around the moving car, as shown in Figure 20, Frames 5 and 410. Additionally, the MIL and Frag trackers have generated slightly higher drifts compared to the STRUCK, CT, KCF and our tracker at the beginning of the UAV tracking application. When the moving car is conducting the in-plane rotation movement, these six trackers have started to generate larger drifts, as shown in Figure 20, Frame 520; they are not able to locate the head of the moving car. Before the moving car is out-of-view, the MIL also lost track of the moving car; it located the "HUMP" logo on the road, as shown in Figure 20, Frame 582. When some portion of the moving car is out-of-view, only KCF and our tracker can continue to locate the moving car well, achieving better tracking performances. As can be seen from Figure 14b, we can also find that the MIL, STRUCK, CT, Frag, KCF and our presented tracker have outperformed the TLD tracker.

4.4. Test 4: Visual Tracking of the UGV with the Landing Pad (UGV_{lp})

In this test, a moving UGV with a landing pad is chosen. During the UAV tracking process, the direction of the UGV is manually controlled by an operator, as Frames 543, 770 and 1090 shown in Figure 21. Thus, some portion of target object is occluded by the operator's hand.

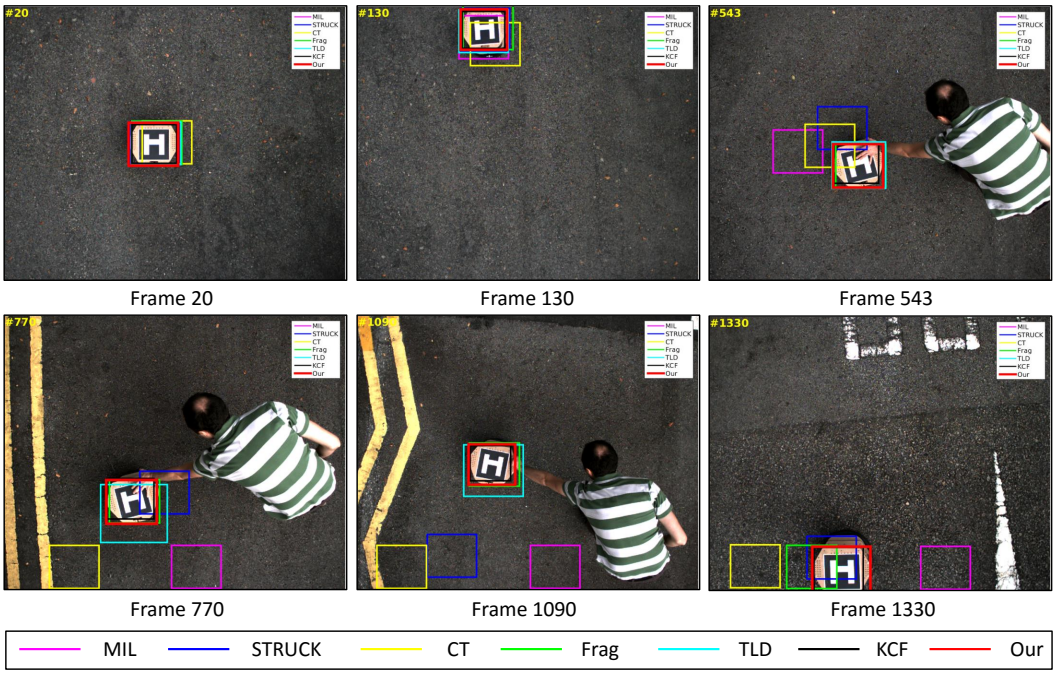


Figure 21. Some tracking results in the UGV_{lp} image sequence. The total number of frames: 1325.

As can be seen from Figures 9 and 15, the KCF, Frag, TLD and our presented tracker have outperformed the CT, MIL and STRUCK trackers. Especially, the CT and our presented tracker have generated the drifts on Frame 3 because of the sudden roll rotation of our UAV. However, our presented tracker has tracked the target object back on Frame 4, while the CT tracker has learned a new appearance on Frame 3, i.e., the background information around the target object has been included as positive samples to train and update its model. Therefore, the CT tracker cannot locate the target object well from Frame 4. For the MIL and STRUCK trackers, their performances are similar to that of the CT tracker in a way that both of them have also adapted to the new appearances of target objects when our UAV is quickly moving forward. Although the CLE error of the KCF tracker is less than ours, the average success rate of our presented tracker is better than the one of the KCF tracker.

4.5. Test 5: Visual Tracking of Walking People Below ($People_{bw}$)

Recently, different commercial UAVs have been developed to follow a person. However, most of these UAVs still mainly depend on the GPS and IMU sensors to achieve the person following application. Therefore, a moving person is selected as the target object for our vision-based UAV tracking task in this paper. The task of the fifth test is that our UAV is utilized to locate a moving person from a high altitude using its onboard downward-looking camera. The main challenging factors include deformation and in-plane rotation, as shown in Figure 22.

As shown in Figure 10, we can find that all visual trackers except the TLD and CT trackers, can locate the moving person in all image frames. Their CLE errors are less than 46 pixels. As can be seen in Figure 16a, their precisions have achieved more than 90% when the CLE threshold is 27 pixels. For the TLD tracker, it has lost the target object from Frame 72, since it has adapted to a new appearance of the target object when the deformable target object is conducting in-plane rotation. After the target object moved back to the previous positions, the TLD tracker has learned the appearance of the target object back. Therefore, the TLD tracker can locate the target object well. From Frame 566, the TLD tracker has lost its deformable target object again until the end of the visual tracking application because of in-plane rotation. For the CT tracker, its tracking performance has also been influenced by Figure 16b; although the MIL, STRUCK, KCF and our tracker can track the target object well, their average success rates are relatively low because of in-plane rotation and deformation.

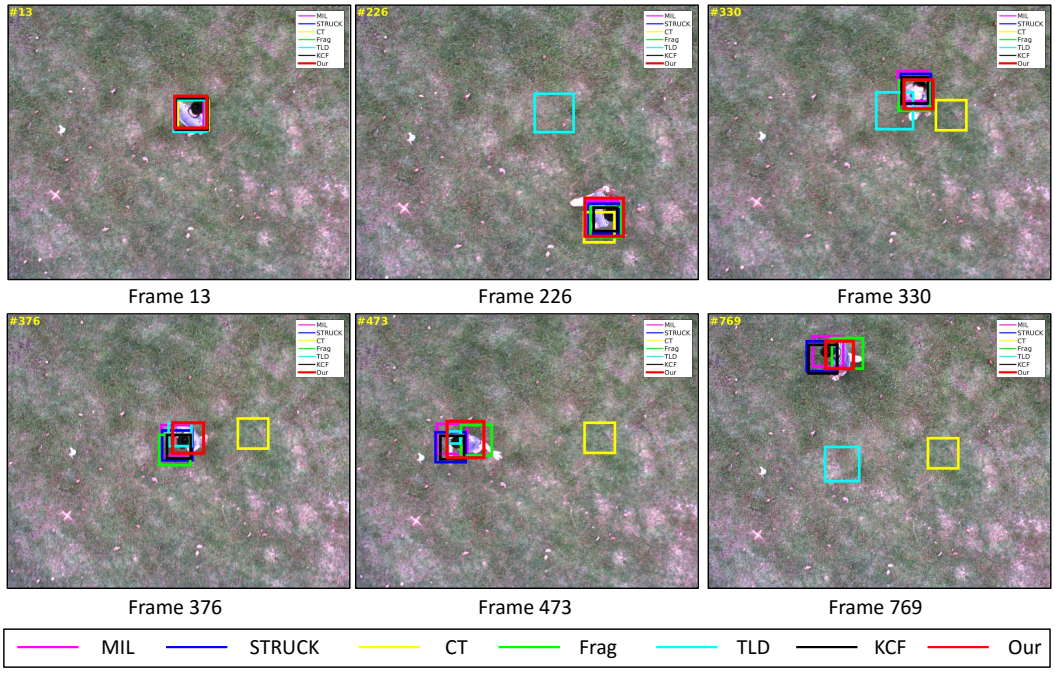


Figure 22. Some tracking results in the People$_{bw}$ image sequence. The total number of frames: 934.

4.6. Test 6: Visual Tracking of Walking People in Front (People$_{fw}$)

In this test, our UAV is employed to follow a moving person using its onboard forward-looking camera. The main challenging factors include deformation, scale variation, cluttered background and out-of-plane rotation (e.g., Figure 23, Frames 312, 1390 and 1970).

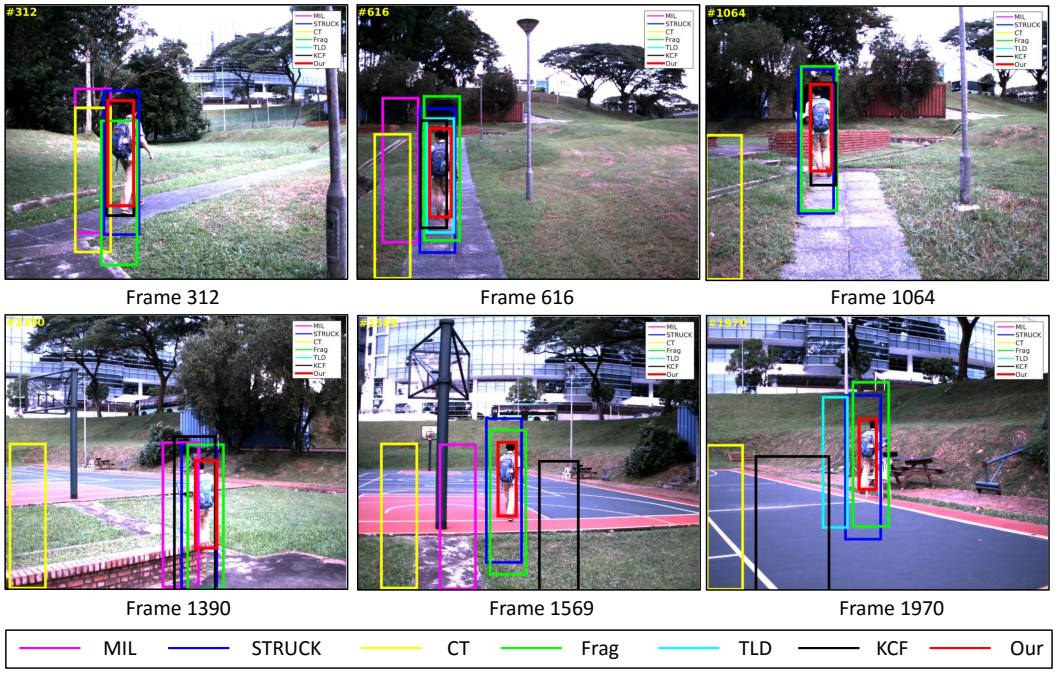


Figure 23. Some tracking results in the People$_{fw}$ image sequence. The total number of frame: 2062.

As can be seen in Figure 11 and Frame 312 shown in Figure 23, the MIL and CT trackers have lost their target object because of the similar appearance in the background, e.g., a tree. Although the

MIL tracker has relocated its target object from Frame 1151 and 1375, it has continued to lose the target object from Frame 1290 and 1544 since the appearances of the brick fence and road are similar to the ones of the target object. For the KCF tracker, it can locate the target object well at the beginning of the UAV tracking application. However, it also lost its target from Frame 1432 due to the similar appearance of the background, e.g., brick fence. For the TLD tracker, it is prone to lose the target object when the target object is conducting out-of-plane rotation. On the other hand, the other visual trackers can always locate the target object, especially the STRUCK and our presented tracker, which are able to track their target object within 30-pixel CLE errors. However, our presented tracker has outperformed the STRUCK tracker in the average success rate, as shown in Figure 17b, since the STRUCK tracker cannot estimate the scale of the target object.

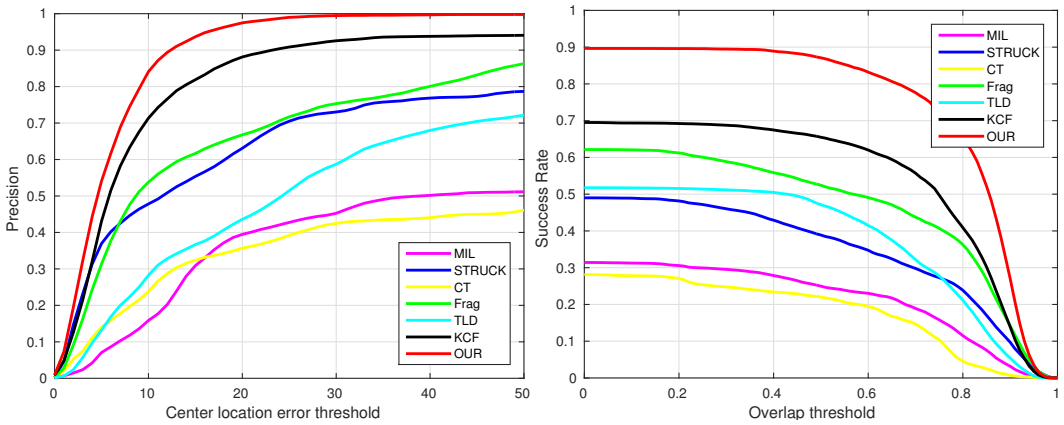
4.7. Discussion

4.7.1. Overall Performances

Tables 3 and 4 show the overall performances of all visual trackers.

For the average CLE error (i.e., CLE_{Ave}) of all image sequences, our presented tracker, the KCF and Frag trackers are ranked as No. 1, No. 2 and No. 3 in all visual trackers. For the average FPS (i.e., FPS_{Ave}), the KCF, our presented tracker and CT tracker have achieved 149.8, 38.9 and 28.7 frames per second, resulting in rankings of No. 1, No. 2 and No. 3 among all visual trackers. For the average success rate (SR) (i.e., SR_{Ave}) when the ξ is set to 0.5, our presented tracker, the KCF and Frag trackers are ranked as No.1, No. 2 and No. 3 again, especially for our presented visual tracker, which has achieved a 95.9% success rate in all image sequences.

Figure 24 shows the overall performances of all visual trackers in 11,646 image frames with precision and success plots. It can be seen that our presented tracker has obtained the best performance. In addition, the KCF and Frag trackers are ranked No. 2 and No. 3.



(a) Precision plot. **(b)** Success plot.

Figure 24. Overall performances of all visual trackers in 11,646 image frames.

The video related to the tracking results of all visual trackers can be checked at the following YouTube link: https://youtu.be/cu9cUYqJ1P8.

4.7.2. Failure Case

Our presented visual tracker cannot handle the below situations properly as it has employed the features to represent the target object: (1) strong motion blur; and (2) large out-of-plane rotation. These cases can result in cases in which the feature detector cannot detect many features, leading to imprecise estimation for the bounding box of the target object.

5. Conclusions

In this paper, a novel robust onboard visual tracker has been presented for long-term arbitrary 2D or 3D object tracking for a UAV. Specifically, three main modules, i.e., the GMLT, LGF and LOF modules, have been developed to obtain a reliable global-local feature-based visual model efficiently and effectively for our visual tracking algorithm, which has achieved the real-time frame rates of more than thirty-five FPS. The extensive UAV flight tests show that our presented visual tracker has outperformed the most promising state-of-the-art visual trackers in terms of robustness, efficiency and accuracy and overcome the object appearance change caused by the challenging situations. It is to be noted that the KCF tracker has achieved an average of 149.8 FPS, but its tracking precision and average success rate are less than ours in all image sequences. Additionally, the UAV can achieve good control performance with more than 20 FPS in real flights. In addition, our visual tracker does not require software or hardware stabilization systems, e.g., a gimbal platform, for stabilizing the onboard captured consecutive images throughout the UAV flights.

Acknowledgments: The research was partially supported by the ST Engineering-NTU Corporate Lab through the NRF corporate lab@university scheme.

Author Contributions: Changhong Fu has developed the presented visual tracking algorithm and carried out the UAV flight experiments in collaboration with Ran Duan and Dogan Kircali. Erdal Kayacan has supervised the work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fu, C.; Suarez-Fernandez, R.; Olivares-Mendez, M.; Campoy, P. Real-time adaptive multi-classifier multi-resolution visual tracking framework for unmanned aerial vehicles. In Proceedings of the 2nd Workshop on Research, Development and Education on Unmanned Aerial Systems (RED-UAS), Compiegne, France, 20–22 November 2013; pp. 99–106.
2. Lim, H.; Sinha, S.N. Monocular localization of a moving person onboard a Quadrotor MAV. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 2182–2189.
3. Fu, C.; Carrio, A.; Olivares-Mendez, M.; Suarez-Fernandez, R.; Campoy, P. Robust real-time vision-based aircraft tracking from Unmanned Aerial Vehicles. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 5441–5446.
4. Babenko, B.; Yang, M.H.; Belongie, S. Visual tracking with online Multiple Instance Learning. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Miami, FL, USA, 20–25 June 2009; pp. 983–990.
5. Hare, S.; Saffari, A.; Torr, P. Struck: Structured output tracking with kernels. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 263–270.
6. Zhang, K.; Zhang, L.; Yang, M.H. Real-time compressive tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Florence, Italy, 7–13 October 2012; pp. 864–877.
7. Adam, A.; Rivlin, E.; Shimshoni, I. Robust fragments-based tracking using the integral histogram. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), New York, NY, USA, 17–22 June 2006; pp. 798–805.
8. Kalal, Z.; Mikolajczyk, K.; Matas, J. Tracking-learning-detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2012**, *34*, 1409–1422.
9. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 583–596.
10. Rosten, E.; Drummond, T. Machine learning for high-speed corner detection. In Proceedings of the 9th European Conference on Computer Vision (ECCV), Graz, Austria, 7–13 May 2006; pp. 430–443.
11. Calonder, M.; Lepetit, V.; Strecha, C.; Fua, P. BRIEF: Binary robust independent elementary features. In Proceedings of the 11th European Conference on Computer Vision (ECCV), Heraklion, Greece, 5–11 September 2010; pp. 778–792.

12. Lucas, B.D.; Kanade, T. An iterative image registration technique with an application to stereo vision. In Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI), Vancouver, BC, Canada, 24–28 August 1981; pp. 674–679.
13. Kalal, Z.; Mikolajczyk, K.; Matas, J. Forward-backward error: automatic detection of tracking failures. In Proceedings of the 20th International Conference on Pattern Recognition (ICPR), Istanbul, Turkey, 23–26 August 2010; pp. 2756–2759.
14. Mullner, D. Fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python. *J. Stat. Softw.* **2013**, *53*, 1–18.
15. Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. *ACM SIGMOD Rec.* **2000**, *29*, 93–104.
16. Teuliere, C.; Eck, L.; Marchand, E. Chasing a moving target from a flying UAV. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), San Francisco, CA, USA, 25–30 September 2011; pp. 4929–4934.
17. Olivares-Mendez, M.A.; Mondragon, I.; Cervera, P.C.; Mejias, L.; Martinez, C. Aerial object following using visual fuzzy servoing. In Proceedings of the 1st Workshop on Research, Development and Education on Unmanned Aerial Systems (RED-UAS), Sevilla, Spain, 30 November–1 December 2011.
18. Huh, S.; Shim, D. A vision-based automatic landing method for fixed-wing UAVs. *J. Intell. Robotic Syst.* **2010**, *57*, 217–231.
19. Martínez, C.; Campoy, P.; Mondragón, I.F.; Sánchez-Lopez, J.L.; Olivares-Méndez, M.A. HMPMR strategy for real-time tracking in aerial images, using direct methods. *Mach. Vis. Appl.* **2014**, *25*, 1283–1308.
20. Mejias, L.; Campoy, P.; Saripalli, S.; Sukhatme, G. A visual servoing approach for tracking features in urban areas using an autonomous helicopter. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Orlando, FL, USA, 15–19 May 2006; pp. 2503–2508.
21. Campoy, P.; Correa, J.; Mondragon, I.; Martinez, C.; Olivares, M.; Mejias, L.; Artieda, J. Computer vision onboard UAVs for civilian tasks. *J. Intell. Robotic Syst.* **2009**, *54*, 105–135.
22. Mondragón, I.; Campoy, P.; Martinez, C.; Olivares-Mendez, M. 3D pose estimation based on planar object tracking for UAVs control. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK, USA, 3–7 May 2010; pp. 35–41.
23. Yang, S.; Scherer, S.; Schauwecker, K.; Zell, A. Autonomous landing of MAVs on an arbitrarily textured landing site using onboard monocular vision. *J. Intell. Robotic Syst.* **2014**, *74*, 27–43.
24. Harris, C.; Stephens, M. A combined corner and edge detector. In Proceedings of the Fourth Alvey Vision Conference (AVC), Manchester, UK, 31 August–2 September 1988; pp. 147–151.
25. Lowe, D. Distinctive image features from Scale-Invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110.
26. Bay, H.; Ess, A.; Tuytelaars, T.; Van Gool, L. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.* **2008**, *110*, 346–359.
27. Rublee, E.; Rabaud, V.; Konolige, K.; Bradski, G. ORB: An efficient alternative to SIFT or SURF. In Proceedings of the International Conference on Computer Vision (ICCV), Barcelona, Spain, 6–13 November 2011; pp. 2564–2571.
28. Olivares-Mendez, M.A.; Fu, C.; Ludivig, P.; Bissyandé, T.F.; Kannan, S.; Zurad, M.; Annaiyan, A.; Voos, H.; Campoy, P. Towards an autonomous vision-based unmanned aerial system against wildlife poachers. *Sensors* **2015**, *15*, 31362–31391.
29. Sanchez-Lopez, J.; Saripalli, S.; Campoy, P.; Pestana, J.; Fu, C. Toward visual autonomous ship board landing of a VTOL UAV. In Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), Atlanta, GA, USA, 28–31 May 2013; pp. 779–788.
30. Sampedro, C.; Martinez, C.; Chauhan, A.; Campoy, P. A supervised approach to electric tower detection and classification for power line inspection. In Proceedings of the International Joint Conference on Neural Networks (IJCNN), Beijing, China, 6–11 July 2014; pp. 1970–1977.
31. Ross, D.; Lim, J.; Lin, R.S.; Yang, M.H. Incremental learning for robust visual tracking. *Int. J. Comput. Vis.* **2008**, *77*, 125–141.
32. Fu, C.; Carrio, A.; Olivares-Mendez, M.A.; Campoy, P. Online learning-based robust visual tracking for autonomous landing of Unmanned Aerial Vehicles. In Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, USA, 27–30 May 2014; pp. 649–655.

33. Pestana, J.; Sanchez-Lopez, J.L.; Saripalli, S.; Campoy, P. Computer vision based general object following for GPS-denied multirotor unmanned vehicles. In Proceedings of the American Control Conference (ACC), Portland, OR, USA, 4–6 June 2014; pp. 1886–1891.
34. Kitt, B.; Geiger, A.; Lategahn, H. Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. In Proceedings of the IEEE Intelligent Vehicles Symposium (IV), San Diego, CA, USA, 21–24 June 2010; pp. 486–492.
35. Hartley, R.I.; Zisserman, A. *Multiple View Geometry in Computer Vision*, 2nd ed.; Cambridge University Press: Cambridge, England, 2004.

© 2016 by the authors; licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (http://creativecommons.org/licenses/by/4.0/).