12th IEEE International Conference on Control & Automation (ICCA)
Kathmandu, Nepal, June 1-3, 2016

Long-term Cooperative Tracking using Multiple Unmanned Aerial Vehicles

Yue Yang, Shuhang Zhai, Bharath Ramesh, Xiaoxu Zheng, Cheng Xiang, Ben M. Chen, and Tong Heng Lee

Abstract— **In this paper, we study the long-term cooperative tracking in large areas using multiple unmanned aerial vehicles (UAVs). Firstly, a hybrid tracking algorithm, which switches between a correlation filters based tracker and a cascaded detector, is developed for the robust long-term tracking using a single UAV. Then, a centralized cooperation strategy is proposed to achieve cooperative tracking using multiple UAVs. Finally, the developed algorithms are demonstrated using both a public database and a laboratory mock-up.**

I. Introduction

In recent years, there has been a growing interest in the cooperative control of inexpensive small UAVs in military [1] and civil applications [2], due to the extended capabilities that multiple UAVs offer with respect to a single UAV for the same task. Among different applications, area surveillance and higher-level vision-based tasks, have drawn most of attention.

In this paper, we study the long-term cooperative tracking in large areas using multiple hovering UAVs. Consider the video stream taken by a group of hovering UAVs with on-board cameras, the goal of long-term cooperative tracking is to automatically determine the object's state (camera ID and bounding box) or indicate that the object is not visible in real-time for a long period, given the initial state of the object. Therefore, we need to solve two problems to achieve long-term cooperative tracking using multiple UAVs. One is the long-term robust tracking for a single UAV and the other one is the cooperation among different UAVs.

The long-term tracking for a single UAV can be approached either from tracking or from detection perspectives. Trackers only require initialization, are fast, and produce smooth trajectories. However, they accumulate error during runtime and typically fail if the object disappears from the camera view. On the other hand, detectors do not drift and do not fail if the object disappears from the camera view. However, they require an offline training stage and are time-consuming. Therefore, neither tracking nor detection can solve the long-time tracking task for a single UAV independently, and we have to use them both and design an appropriate strategy to benefit one from another. In particular, a tracker can provide weakly labeled training data for a detector and thus improve it during runtime. A detector can reinitialize a tracker and thus reduce the tracking failures.

Y. Yang, S. H. Zhai, B. Ramesh, X. X. Zheng, C. Xiang, B. M. Chen, and T. H. Lee are with the Department of Electrical and Computer Engineering, National University of Singapore. (Email: eleyyue@nus.edu.sg, zhaishuhang@u.nus.edu, elerame@nus.edu.sg, zheng_xiaoxu17@u.nus.edu, elexc@nus.edu.sg, elecbm@nus.edu.sg, eleleeth@nus.edu.sg).

Object tracking is to estimate the states of the target in the subsequent frames, given the initial state (e.g., position and extent) of a target object in the first frame. In general, tracking algorithms can be categorized into two classes based on their representation schemes: generative [3], [4], [5], [6], [7] and discriminative models [8], [9], [10], [11], [12]. Generative algorithms typically learn an appearance model and use it to search for image regions with minimal reconstruction errors as tracking results. However, generative models do not take surrounding visual context into account and discard useful information that can be exploited to better separate target object from the background. On the other hand, discriminative models pose object tracking as a detection problem in which a classifier is learnt to separate the target object from its surrounding background within a local region. However, most of the approaches use a sparse sampling strategy to choose negative samples, which inhibits the tracking performance. By observing the fact that the process of taking subwindows of an image induces circulant structure, correlation filters based trackers [13], [14], [15], [16] establish links to Fourier analysis that allows the use of the Fast Fourier Transform (FFT) to quickly incorporate information from all subwindows. In particular, the Discriminative Scale Space Tracker (DSST) [16] can deal with scale change and achieve real-time processing simultaneously.

Object detection is the task of localization of objects in every frame independently. Object detection methods are typically based on the local image features [17] or a sliding window [18]. The main limitations of the feature-based detection are the detection of image features and the requirement of knowing the geometry of the object in advance. The sliding window-based approaches [18] scan the input image by a window of various sizes and decide whether the underlying patch contains the object of interest or not for each window. To reduce computational load, sliding window-based detectors adopt the cascaded architecture [18].

In the literature, there are already several approaches which combine tracking and detection in some sense. In [19], an offline trained detector is used to validate the trajectory output by a tracker and, if the trajectory is not validated, an exhaustive image search is performed to find the target. Other approaches integrate the detector within a particle filtering framework [20]. However, these methods rely on an offline trained detector that does not change its properties during runtime. Recently, a tracking-learning-detection (TLD) algorithm, which treats tracking and detection as two independent processes and exchange information using learning is proposed [21]. By keeping the tracking and detection separated, this approach does not have to compromise either on tracking or detection capabilities of its components. However, there are two major drawbacks for TLD. First, the tracker it uses is just a variant of optical flow, whose performance is not as good as those sophisticated trackers in the literature. Second, TLD proceeds tracking and detection simultaneously for every frame, which is too time-consuming for real-time applications. Considering these two issues, this paper proposes a novel hybrid tracking algorithm which can achieve good performance and real-time processing simultaneously.

With a long-term robust tracking algorithm for a single UAV, we proceed to handle the cooperation among different UAVs. Multiple-UAV cooperation can be categorized into centralized and distributed approaches. In the centralized approach, a central control agent (one of the UAVs or a control center) allocates the missions to each UAV based on the global information available from all UAVs. In the distributed approach, each UAV chooses its own mission based only on its local information or that communicated by neighboring UAVs. In our setting [22], the ground control station (GCC) plans the hovering positions for all the UAVs to achieve complete coverage of the target area and each UAV sends its own image to the GCC for surveillance and object tracking at any time instant. In this case, a centralized strategy will be adopted to achieve cooperative tracking using multiple UAVs. In particular, we aim to develop a novel centralized strategy to achieve cooperative tracking in real-time in this paper.

The contents of this paper are organized as follows. In Section II, a hybrid tracking algorithm that switches between a correlation filters based tracker and a cascaded detector, is developed for the long-term robust tracking for a single UAV. The cooperation strategy among different UAVs is then presented in Section III. In Section IV, the developed algorithms are demonstrated using both a public database and a laboratory mock-up. Finally, in Section V, conclusions will be drawn.

II. Long-term Robust Tracking for a Single UAV

In this section, we propose a hybrid tracking algorithm to achieve long-term robust tracking for a single UAV. In particular, a correlation filters based tracker, which can deal with scale change and process in real-time [16], is utilized to track the target when it is not occluded. In addition, a sliding window-based cascaded detector [21], will be used to re-detect the target object in case of tracking failure.

A. Correlation Filters based Tracker

The correlation filters based tracker [16] models the appearance of objects using a two-dimensional translation correlation filter and a one-dimensional scale correlation filter trained on example images. Different from [13], [14], [15], it can deal with scale change of the object and still achieve real-time processing.

Consider a d-dimensional feature map and let f be a rectangular patch of the target extracted from this feature map. We denote feature dimension number $l \in \{1, \cdots, d\}$ of f by f^l. The objective is to find an optimal correlation filter h, consisting of one filter h^l per feature dimension. This is achieved by minimizing the cost function:

$$\varepsilon = \| \sum_{l=1}^{d} h^l \star f^l - g \|^2 + \lambda \sum_{l=1}^{d} \| h^l \|^2 \tag{1}$$

where $\star$ denotes circular correlation and g is the desired correlation output associated with the training example f. The parameter $\lambda \geq 0$ controls the impact of the regularization term. The solution to (1) is:

$$H^l = \frac{\bar{G} \odot F^l}{\sum_{k=1}^{d} \bar{F^k} \odot F^k + \lambda} \tag{2}$$

where capital letters denote the Discrete Fourier Transforms (DFTs) of the corresponding functions. Moreover, $\bar{G}$ represents the complex conjugate of G and $\odot$ denotes point-wise multiplication. To obtain a robust approximation, the numerator A_t^l and denominator B_t of the correlation filter H_t^l in (2) are updated separately as:

$$A_t^l = (1-\eta)A_{t-1}^l + \eta \bar{G}_t \odot F_t^l \tag{3}$$

$$B_t = (1-\eta)B_{t-1} + \eta \sum_{k=1}^{d} \bar{F_t^k} \odot F_t^k \tag{4}$$

where η is a learning rate parameter. The new target state is then found by maximizing the score y.

$$y = \mathscr{F}^{-1}\left(\frac{\sum_{l=1}^{d} \bar{A}^l Z^l}{B+\lambda}\right) \tag{5}$$

Moreover, the training example f for updating the scale filter is computed by extracting features using variable patch sizes centred around the target. Let $P \times R$ denote the target size in the current frame and S be the size of the scale filter. For each $n \in \{\lfloor -\frac{S-1}{2} \rfloor, \cdots, \lfloor \frac{S-1}{2} \rfloor\}$, we extract an image patch J_n of size $a^n P \times a^n R$ centred around the target. Here a denotes the scale factor between feature layers. The value $f(n)$ of the training example f at scale level n is set to the d-dimensional feature descriptor of J_n.

B. Cascaded Detector

In object detection, each subwindow is tested independently whether it contains the object of interest. Cascaded object detectors aim at rejecting as many non-relevant subwindows with a minimal amount of computation. We restrict the search space to a subspace $\mathscr{R}$ by employing the following constraints. First, we assume that the object of interest retains its aspect ratio. Furthermore, we introduce margins d_x and d_y between two adjacent subwindows and set d_x and d_y to be $\frac{1}{10}$ of the values of the original bounding box. In order to employ the search on multiple scales, we use a scaling factor $s = 1.1^a$, $a \in \{-5, \cdots, 5\}$ for the original bounding box of the object of interest. We also consider subwindows with a minimum size of 15×15 only. The three stages that we use for image classification are as follows.

1) Patch Variance: The variance of an image patch is a measure for uniformity. The stage exploits the fact that gray-value variance of a patch p can be expressed as

$$\sigma^2 = \mathbb{E}(p^2) - \mathbb{E}^2(p) \tag{6}$$

where the expected values $\mathbb{E}(p)$ and $\mathbb{E}(p^2)$ can be measured in constant time using integral images [18]. This stage rejects all patches for which gray-value variance is smaller than 50% of variance of the patch that was selected for tracking. This stage typically rejects more than half of non-object patches.

2) Ensemble Classifier: In the second stage of the detection cascade, we employ an ensemble classification method presented in [23], [24] that is known as random fern classification. The input to the ensemble is an image patch that was not rejected by the variance filter. The ensemble consists of n basis classifiers. Each base classifier i performs a number of pixel comparisons on the patch, resulting in a binary code x which indexes to an array of posteriors $P_i(y|x)$, where $y \in \{0, 1\}$. In particular, 10 random trees, each with 13 features, are used in the ensemble classifier. The posteriors of individual base classifier are averaged and the ensemble classifies the patch as the object if the average posterior is larger than 50%.

3) Template Matching: After filtering the patches by the variance filter and the ensemble classifier, we are typically with several of bounding boxes that are not decided yet. Therefore, we can use the online model and classify the patch using a nearest neighbor (NN) classifier. This stage is more restrictive than the ensemble classification method since the comparison is performed on a pixel-by-pixel level. All patches are resized to 15×15 pixels. For comparing two patches p_1 and p_2, we employ a similarity whose value is between 0 and 1 as follows.

$$S(p_1, p_2) = 0.5(NCC(p_1, p_2) + 1) \tag{7}$$

where $NCC(p_1, p_2)$ is the Normalized Correlation Coefficient (NCC) of p_1 and p_2. The patch is classified as the object if the similarity value is larger than 50%.

C. Integration

It is noted that the correlation tracker itself works well if the target object never disappears in the view. However, it will fail when the object is fully occluded or disappears in the view. Although the cascaded detector itself can deal with full occlusion, it is too time-consuming to achieve real-time processing. The integrator combines the correlation tracker and cascaded detector, and generates a single bounding box or a flag indicating the object is not visible. For the purpose of real-time processing, the tracker will have high priority in the whole algorithm. In other words, the detector will only be activated when the tracker fails. Once the object is re-detected, the detector will be deactivated and the tracker will be re-activated. In particular, the tth iteration of the proposed algorithm will run as follows:

If the previous object state is nonempty, the tracker will be performed on the current frame. If the confidence value for the tracking result is bigger than τ_t, the bounding box obtained by the tracker will be treated as the output. Moreover, the tracker and the detector will be updated when the confidence value for the tracking result is higher than $\alpha\tau_t$ $(\alpha > 1)$.

If the tracker fails, the detector will be activated to search the current frame. If the confidence value for the detection result is bigger than τ_d, the bounding box generated by the detector will be treated as the output and the tracker will be re-activated. The detailed framework for the integrated algorithm is shown in Alg. 1.

Algorithm 1 Robust Long-term Tracking Algorithm: iteration at time step t

Input: Previous object state B_{t-1}
Output: Current object state B_t

```
 1: $B_t = \emptyset$
 2: if $B_{t-1}$ is nonempty then
 3:     Get the tracking result $T_t$ using the tracker
 4:     if Conf($T_t$) $> \tau_t$ then
 5:         $B_t = T_t$
 6:         if Conf($T_t$) $> \alpha\tau_t$ then
 7:             Update the detector and the tracker
 8:         end if
 9:     end if
10: end if
11: if $B_t$ is empty then
12:     Get the detection result $D_t$ using the detector
13:     if Conf($D_t$) $> \tau_d$ then
14:         $B_t = D_t$
15:     end if
16: end if
```

III. Cooperation among Different UAVs

With the robust long-term tracking algorithm, we proceed to develop a cooperative strategy to achieve long-term cooperative tracking using multiple UAVs. In our setting [22], multiple cameras are placed at the same altitude to monitor the area of interest and track moving vehicles. To initialize the tracking process, the camera ID and corresponding bounding box of the target for the first frame need to be provided by the user as input. At any time instant, the ground control station needs to determine both the active camera and the corresponding bounding box (or a flag indicating the target is not visible).

In total, there are three operation modes: (1) tracking in the frame obtained by the previous active camera, (2) detection in the frame obtained by the previous active camera, and (3) detection in neighbor cameras of the previous active camera. The priorities of these three modes decrease gradually in the whole procedure. The detailed framework for the cooperative tracking algorithm is shown in Alg. 2.

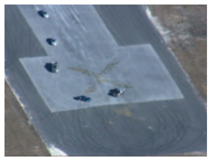
IV. Experimental Results

In this section, we evaluate the proposed algorithms on a public dataset —VIVID (Video Verification of Identity) [25] shown in Fig. 1, and a laboratory mock-up shown in Fig. 2.

Algorithm 2 Cooperative Tracking Algorithm: iteration at time step t

```
Input: Previous active camera $ID_{t-1}$, previous object state $B_{t-1}$
Output: Current active camera $ID_t$, current object state $B_t$
 1: $ID_t = ID_{t-1}$
 2: $B_t = \emptyset$
 3: if $B_{t-1}$ is nonempty then
 4:     Get the tracking result $T_t$ in camera $ID_{t-1}$
 5:     if $\mathrm{Conf}(T_t) > \tau_t$ then
 6:         $B_t = T_t$
 7:         $ID_t = ID_{t-1}$
 8:         if $\mathrm{Conf}(T_t) > \alpha\tau_t$ then
 9:             Update the detector and the tracker
10:         end if
11:     end if
12: end if
13: if $B_t$ is empty then
14:     Get the detection result $D_t$ in camera $ID_{t-1}$
15:     if $\mathrm{Conf}(D_t) > \tau_d$ then
16:         $B_t = D_t$
17:         $ID_t = ID_{t-1}$
18:     else
19:         for Each neighbor camera $ID^i$ do
20:             Get the detection result $D^i$ in camera $ID^i$
21:         end for
22:         if $\max \mathrm{Conf}(D^i) > \tau_d$ then
23:             $B_t = D_{\arg\max \mathrm{Conf}(D^i)}$
24:             $ID_t = ID_{\arg\max \mathrm{Conf}(D^i)}$
25:         end if
26:     end if
27: end if
```

(a) redteam


(b) egtest01

(c) egtest02

(d) egtest03

(e) egtest04

(f) egtest05

(g) pktest01

(h) pktest02

(i) pktest03

Fig. 1. The VIVID database.

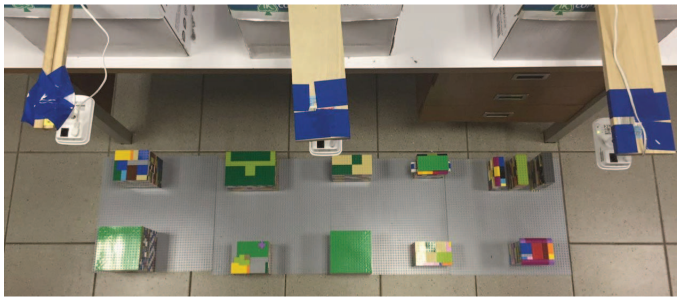


Fig. 2. The laboratory mock-up.

A. Implementation

The regularization parameter of (1) is set to $\lambda = 0.01$. The size of the search window for translation estimation is set to 2 times of the target size. We use $S = 33$ numbers of scales with a scale factor of $a = 1.02$. The learning rate in (4) is set to $\eta = 0.025$. The thresholds in Alg. 1 and Alg. 2 are set as $\tau_t = 0.15$, $\alpha = 2$, and $\tau_d = 0.6$.

The Graphic User Interface (GUI) for the cooperative tracking system is shown in Fig. 3.

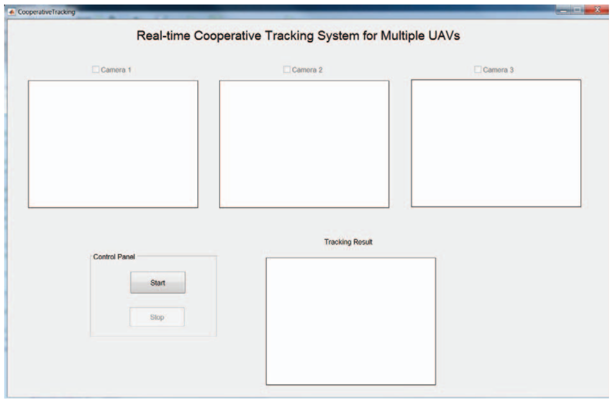


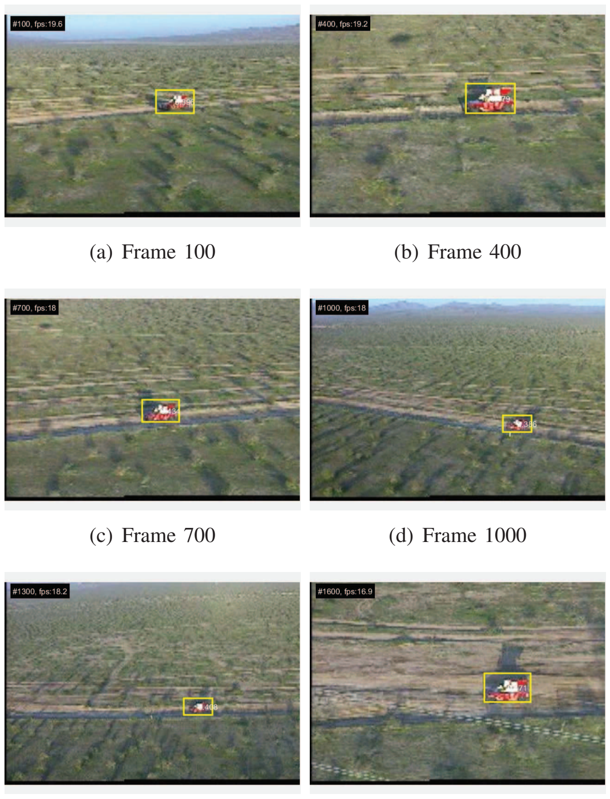
Fig. 3. GUI for the cooperative tracking system.

B. Robust Long-term Tracking on VIVID

Due to space limitation, we only show the tracking results of three video clips in the VIVID dataset.

1) redteam: In this video, a single vehicle drives on straight road through the desert, then turns a corner at the end. The major difficulty is the long shadows along the road. The tracking results for this video are shown in Fig. 4. It is shown that good tracking performance can be achieved with an average speed around 20 fps.

2) egtest05: In this video, the target is to track a pickup truck along a dirt road in a wooded area. The main challenges are full occlusion by trees, and illumination changes as truck passes in and out of tree shadows. The tracking results for this video are shown in Fig. 5. As can be seen from Fig. 5, good tracking performance can be achieved with an average speed around 20 fps. However, in the 700th frame, the tracking result is actually not the vehicle we are interested

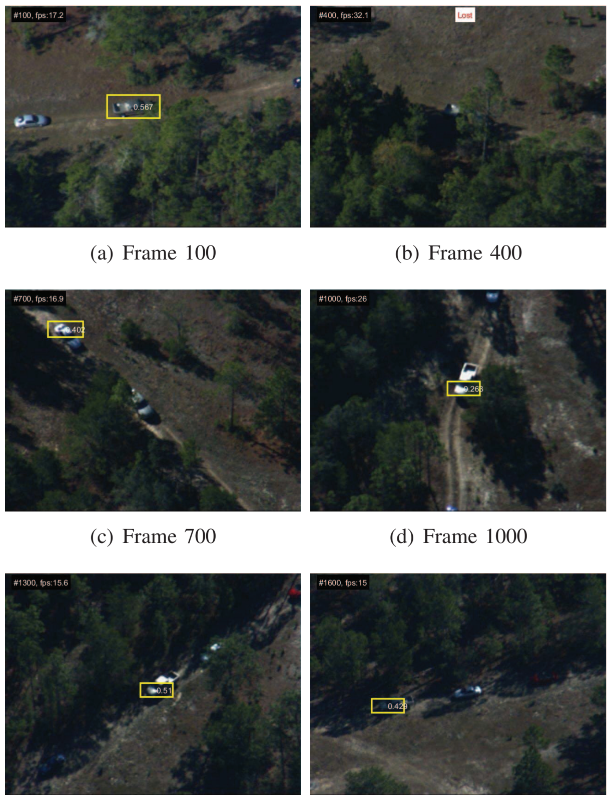


(a) Frame 100 (b) Frame 400

(c) Frame 700 (d) Frame 1000

(e) Frame 1300 (f) Frame 1600

Fig. 4. Tracking results for redteam.



(a) Frame 100 (b) Frame 400

(c) Frame 700 (d) Frame 1000

(e) Frame 1300 (f) Frame 1600

Fig. 5. Tracking results for egtest05.

in (the first car with light color) but another one which is very similar to the target. This is understandable since the detector cannot distinguish these two vehicles when the target is lost from the tracker.

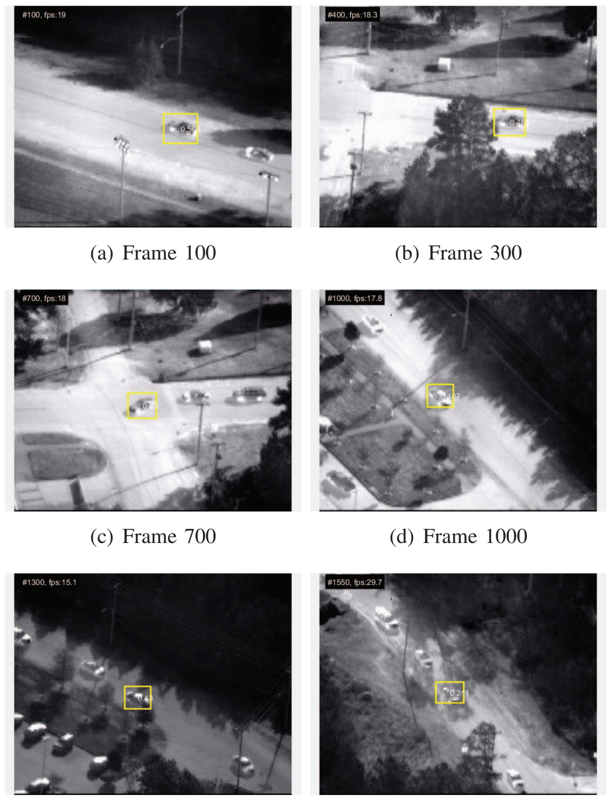
3) pktest02: This is a thermal IR video of a line of vehicles, which pause at the intersection then continue. The major challenges are full occlusion by trees and passing through shadows. The tracking results for this video are shown in Fig. 6. As can be seen from Fig. 6, good tracking performance can be achieved with an average speed around 20 fps. This example shows that our algorithm can be well generalized to thermal IR videos.

C. Cooperative Tracking on a Laboratory Mock-up

As can be seen from Fig. 2, the three cameras from right to left correspond to cam 1, cam 2 and cam 3 respectively in Fig. 3. In our experiment, one white car and one red car drive straightly from right to left. The white car will be manually selected as the object of interest. The cooperative tracking results are shown in Fig. 7. It shows that good tracking performance is achieved even when the target moves from the view of one camera to another one.

V. CONCLUSIONS

In this paper, a long-term cooperative tracking algorithm was proposed for object tracking in large areas using multiple UAVs. In particular, a hybrid tracking algorithm, which

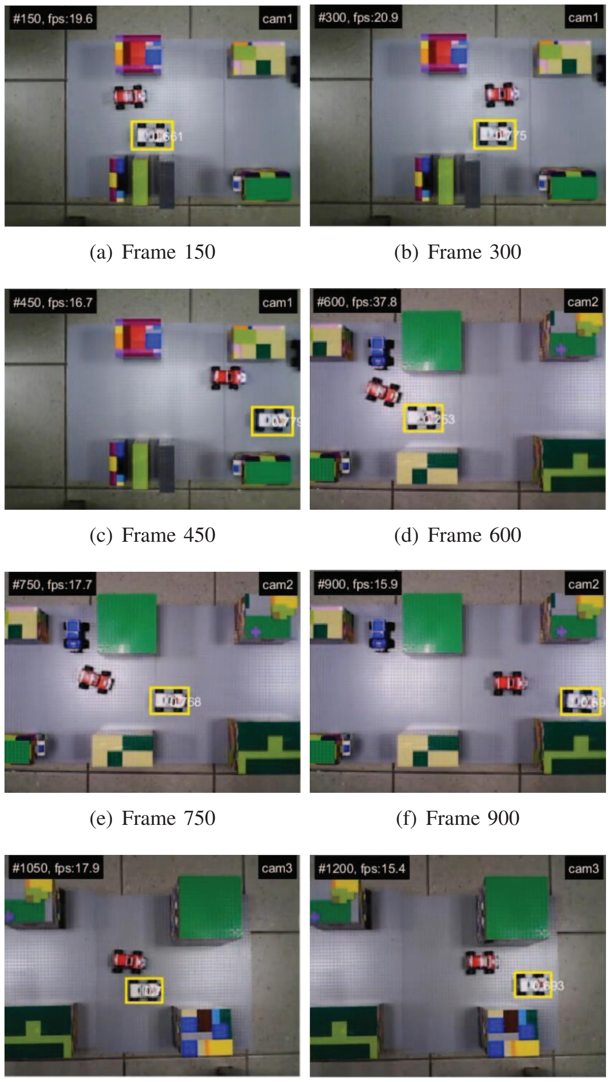


(a) Frame 100 (b) Frame 300

(c) Frame 700 (d) Frame 1000

(e) Frame 1300 (f) Frame 1550

Fig. 6. Tracking results for pktest02.



(a) Frame 150 (b) Frame 300

(c) Frame 450 (d) Frame 600

(e) Frame 750 (f) Frame 900

(g) Frame 1050 (h) Frame 1200

Fig. 7. Cooperative tracking results.

switches between a correlation filters based tracker and a cascaded detector, was derived for a single UAV. Based on this tracking algorithm, a cooperative strategy was then proposed to achieve object tracking in large areas. Finally, the effectiveness of our proposed algorithms was demonstrated in both a public dataset and a laboratory mock-up.

ACKNOWLEDGMENTS

The work in this paper is funded by TDSI, Singapore under the project R-263-000-A62-592/422/232.

REFERENCES

[1] T. Samad, J. S. Bay, and D. Godbole, "Network-centric systems for military operations in urban terrain: the role of UAVs," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 92–107, 2007.
[2] V. Shaferman and T. Shima, "Unmanned aerial vehicles cooperative tracking of moving ground target in urban environments," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 5, pp. 1360–1371, 2008.
[3] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, vol. 26, no. 1, pp. 63–84, 1998.
[4] A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi, "Robust online appearance models for visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1296–1311, 2003.
[5] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2006.
[6] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008.
[7] X. Mei and H. Ling, "Robust visual tracking using l_1 minimization," in *Proceedings of IEEE International Conference on Computer Vision*, 2009.
[8] S. Avidan, "Support vector tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 8, pp. 1064–1072, 2004.
[9] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1631–1643, 2005.
[10] H. Grabner, M. Grabner, and H. Bischof, "Real-time tracking via online boosting," in *Proceedings of British Machine Vision Conference*, 2006.
[11] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *Proceedings of IEEE International Conference on Computer Vision*, 2011.
[12] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 8, pp. 1619–1632, 2011.
[13] D. S. Bolme, J. R. Beveridge, B. Draper, Y. M. Lui, *et al.*, "Visual object tracking using adaptive correlation filters," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2010.
[14] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proceedings of European Conference on Computer Vision*, 2012.
[15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.
[16] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proceedings of British Machine Vision Conference*, 2014.
[17] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.
[18] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2001.
[19] O. Williams, A. Blake, and R. Cipolla, "Sparse bayesian learning for efficient visual tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 8, pp. 1292–1304, 2005.
[20] M. Isard and A. Blake, "Condensationłconditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.
[21] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 7, pp. 1409–1422, 2012.
[22] Y. Yang, K. Mohammad, C. Xiang, S. H. Teo, B. M. Chen, and T. H. Lee, "Wide area surveillance of urban environments using multiple mini-vtol uavs," in *Proceedings of the 41st Annual Conference of the IEEE Industrial Electronics Society*, 2015.
[23] V. Lepetit and P. Fua, "Keypoint recognition using randomized trees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1465–1479, 2006.
[24] M. Ozuysal, P. Fua, and V. Lepetit, "Fast keypoint recognition in ten lines of code," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
[25] R. Collins, X. Zhou, and S. K. Teh, "An open source tracking testbed and evaluation web site," in *Proceedings of IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, 2005.