2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Multi-Target Detection and Tracking from a Single Camera in Unmanned Aerial Vehicles (UAVs)

Jing Li[1], Dong Hye Ye[1], Timothy Chung[2], Mathias Kolsch[2], Juan Wachs[1] and Charles Bouman[1]

Abstract— **Despite the recent flight control regulations, Unmanned Aerial Vehicles (UAVs) are still gaining popularity in civilian and military applications, as much as for personal use. Such emerging interest is pushing the development of effective collision avoidance systems. Such systems play a critical role UAVs operations especially in a crowded airspace setting. Because of cost and weight limitations associated with UAVs payload, camera based technologies are the de-facto choice for collision avoidance navigation systems. This requires multi-target detection and tracking algorithms from a video, which can be run on board efficiently. While there has been a great deal of research on object detection and tracking from a stationary camera, few have attempted to detect and track small UAVs from a moving camera.**

In this paper, we present a new approach to detect and track UAVs from a single camera mounted on a different UAV. Initially, we estimate background motions via a perspective transformation model and then identify distinctive points in the background subtracted image. We find spatio-temporal traits of each moving object through optical flow matching and then classify those candidate targets based on their motion patterns compared with the background. The performance is boosted through Kalman filter tracking. This results in temporal consistency among the candidate detections. The algorithm was validated on video datasets taken from a UAV. Results show that our algorithm can effectively detect and track small UAVs with limited computing resources.

I. INTRODUCTION

Increasing affordability, functionality and versatility are leading to a reality where unmanned aerial vehicles (UAVs) are pervasive in the sky for commercial and individual needs [1], [2], [3]. One of the most important challenges associated with UAV's use is collision avoidance or sense-and-avoid capability [4]. As opposed to large autonomous vehicles where most of the collision avoidance is done by LIDAR sensors, UAV have a limited payload to be effective. Inexpensive optical sensors such as Go-Pro color cameras enable low energy consumption, are light, and most suitable for use on UAVs. These provide a cost effective alternative to systems currently in use on larger aircrafts, like traffic collision avoidance system (TCAS).

Machine vision based collision avoidance systems require the detection and tracking of other UAVs from real-time video feeds to be usable [5], [6]. Once other UAVs are detected and tracked, strategies involving a sequence of maneuvers for collision avoidance are followed. For real-time operation, detection and tracking operations must run on-board. This allows continuous operations even when the connection between the drone and the control station is lost, or sensors fail. In this context, real-time object detection and tracking has been the subject of study by the computer vision community in large [7], [8].

Fig. 1. Challenge in detecting other UAVs: [*Left*] Original Video, [*Right*] Video with our detection and tracking; Other UAVs are very small and occluded by complex backgrounds (i.e. cloud) and thus not even recognizable by human eyes. Our proposed method detects and tracks multiple small UAVs successfully as highlighted in red boxes.

Specific challenges associated with object detection and tracking algorithms applied to UAV applications involve the following: (1) The video feed is acquired from a moving camera mounted on the UAV, which requires to stabilize the rapidly changing views (often requiring non-planar geometry based transformations); (2) Due to the speed of UAVs in converging orientations, the moving objects need to be detected in a far distance to enable timely warning before collision. The targets appear small in image frame and often occluded by clutter (e.g. clouds, trees, and specular light) (See Fig. 1 *Left*).

To tackle these challenges, a new approach is presented in this paper to detect and track small UAVs from a rapidly moving camera mounted on a UAV. Initially, the acquired video from the camera is parsed into a sequence of frames and the relative background motion between frames is estimated. The guiding assumption is that UAVs and the background have very different motion patterns. Thus, the moving object can be extracted by compensating the motion of the background. The background motion is calculated via the perspective transform model [9] taking into account globally smooth motion with camera projection. The moving objects' salient points are identified from a background subtracted image. The local motion of the moving objects is determined by applying the Lucas-Kanade optical flow algorithm [10]. Candidate objects (targets) are found by classifying spatio-

[1]Jing Li, Dong Hye Ye, Juan Wachs and Charles Bouman are with the Department of Electrical and Computer Engineering and Industrial Engineering, Purdue University, West Lafayette, IN 47907, USA {li1463,yed,jpwachs,bouman}@purdue.edu

[2]Timothy Chung and Mathias Kolsch are with the Department of System Engineering and Computer Science, Naval Postgraduate School, Monterey, CA 93943, USA {thchung,kolsch}@nps.edu

*This work was supported by the Naval Postgraduate School Grant NPS-BAA-14-004.

temporal features from the estimated local motion of the moving object. Finally, the Kalman filter tracking [11] is applied to reduce the intermittent miss-detections and false alarms found through the camera feed. This algorithm was tested on real videos from UAVs and target UAVs were detected even when they were not visible due to their size and the background complexity (See Fig. 1 *Right*).

II. RELATED WORK

Previous approaches to detect and track moving objects using camera-based systems mounted on UAVs [12] rely mainly on the extraction of salient features in individual frames and use machine learning techniques to prototype the shape and appearance of the target objects in the training dataset [13]. Classifiers including Convolutional Neural Networks (CNN) [14] and Random Forests (RF) [15] to mention a few, have been used for this task, and achieved robust performance even in challenging environments (e.g. variable illumination and background clutter). However, in most cases, the authors assume sufficiently large and clearly visible moving objects with distinguishable shape and appearance features. This assumption does not hold for warning systems since they need to alert early enough to avoid the imminent otherwise collision.

A different approach relies strictly on motion information from the moving object for further detection and tracking. Such approach is suitable for characterizing small moving objects since their motions can be estimated in local regions between frames. Motion-based approaches are divided into two main categories: (1) Background Subtraction based and (2) Optical Flow based. Background subtraction methods identify groups of pixels which brightness remain constant over time and then subtract those pixels from the image to detect the moving objects [7], [16]. These background subtraction based methods work best when background motion can be easily compensated, which is not the case for a fast moving camera. Alternatively, optical flow based methods find corresponding image regions between frames. Then, based on the local motion vectors, the moving objects are detected [10], [17]. The quality of local motion vectors is critical for accurate detection. Blurred images can lead to poor motion vector assessments.

We propose to combine background subtraction and optical flow methods to obtain the best of both worlds. We use background motion estimation for the moving camera as a first approximation, and then subtract most of homogeneous regions in the image to isolate the target objects. These constitute salient regions in the background subtracted image enable to find good points for optical flow matching. More importantly, by comparing background motion and flow vector based approaches, we can extract spatio-temporal features which have shown to be useful for moving object detection and tracking.

III. MULTI-TARGET DETECTION AND TRACKING

As illustrated in Fig. 2, we propose an efficient multi-target detection and tracking algorithm for UAVs. We first estimate the background motion between two sequential video frames and subtract the background to highlight the regions where changes have occurred. Additional moving object detection operations are performed to differentiate target objects from spurious noise. Finally, we use spatio-temporal characteristics of each detected object to identify actual UAV and incorporate the temporal consistency of detected objects through tracking. In following, we describe each component of our algorithm in details.

Fig. 2. An overview of our proposed method: We first estimate the background motion between two sequential frames. From resulting background-subtracted image, we detect the moving objects by pruning spurious noise. Among detected objects, we differentiate UAVs from false alarms using spatio-temporal characteristics and track them for temporal consistency.

A. Background Motion Estimation

For background motion estimation, we assume that the background moves smoothly, not allowing local warping. From a sequence of video frames, we extract a set of points and estimate local motion fields on those selected points. The local motion estimation procedure can be computationally expensive, so it is only performed on a sparse set of selected points based on saliency with appropriately uniform distribution. The computed local motion fields are then fit into a global transformation which represents the background motion.

1) Identify Salient Points: First, we identify salient points in a video frame. Here, we use Shi-Tomasi corner detector [18] due to efficiency. Shi-Tomasi corner detector is based on the assumption that corners are associated with the local autocorrelation function.

Given an image X, we define the local autocorrelation function C at the pixel s as following:

$$C(s) = \sum_{W} [X(s+\delta s) - X(s)]^2 \tag{1}$$

where δs represents a shift and W is a window around s.

The shifted image $X(s+\delta s)$ is approximated by a first-order Taylor expansion and then eq. (1) can be rewritten as following:

$$\begin{aligned} C(s) &= \sum_{W} [\nabla X(s) \cdot \delta s]^2 \\ &= \delta s^T \sum_{W} [\nabla X(s)^T \nabla X(s)] \delta s \\ &= \delta s^T \Lambda \delta s \end{aligned} \tag{2}$$

where ∇X is the first order derivative of the image and Λ is the precision matrix.

In Shi-Tomasi corner detection, a saliency Q is computed according to eigenvalues of Λ.

$$Q(s) = \min\{\lambda_1, \lambda_2\} \tag{3}$$

where λ_1 and λ_2 are eigenvalues of Λ.

After thresholding on Q, we find a set of salient points. To ensure appropriately uniform spatial distribution, we discard points for which there is a stronger corner points at a certain distance.

2) Find Local Motion Fields on Salient Points: We now find the local motion fields from the previous frame X_{t-1} to the current frame X_t on identified salient points. We denote p_{t-1} as one of Shi-Tomasi corner points in X_{t-1}. Then, we compute the motion vector u_t from the point p_{t-1} using Lucas-Kanade method [10] assuming that our local motion is optical flow.

In Lucas-Kanade method, all neighbor points around the given pixel should have the same motion. So, the local motion can be computed by solving the least square problem.

$$u_t = \arg\min_{u} \sum_{s \in \mathcal{N}(p_{t-1})} |X_t(s+u) - X_{t-1}(s)|^2 \quad (4)$$

where $\mathcal{N}(p_{t-1})$ is the neighborhood around p_{t-1}. It is worth noting that eq. 4 is easy to solve with a closed-form solution. Furthermore, we use bi-directional verification to obtain accurate motion vector such that $||u_t + (u_t)^{-1}||_2$ has small value.

3) Fit Local Motion Fields to a Global Transformation: After finding a set of local motion fields u_t, we fit them into a global transformation. We now denote $p_t = p_{t-1} + u_t$ as the corresponding point in the current frame X_t through optical-flow matching.

We then find the global transformation H_t which regularizes local motion fields to be smooth in the entire image.

$$H_t = \arg\min_{H} \sum_{p_t \in \mathbf{P}_t, p_{t-1} \in \mathbf{P}_{t-1}} ||p_t - H \circ p_{t-1}||_2^2 \quad (5)$$

where $\mathbf{P}_t$ and $\mathbf{P}_{t-1}$ represent a set of corresponding points in X_t and X_{t-1}, respectively, and $\circ$ is the warping operation.

Then, there are many widely-used global transformation models such as rigid or affine transformation model. Here, we choose the perspective transformation model [9] reflecting the fact that a UAV occupy a small portion of the field of view. The perspective transformation model is efficient to compute because it requires only 9 parameters to describe and it can take account into projection based on the distance from the camera. We assign the resulting perspective transformation in eq. 5 as the background motion between two consecutive frames.

B. Moving Object Detection

Given the estimated background motion, we compute the background subtracted image to highlight moving objects which have more complex motion. Then, we identify the salient points in the background subtracted image and use appearance information to find the local motion vector on those points. The additional test is performed to prune spurious noise assuming that motion of target objects is largely different from the background motion.

1) Compute Background Subtracted Image: We can subtract the background by taking difference between original image and background motion compensated image. However, the estimated background motion may not be accurate as single plane assumption on the perspective model can be violated in a video. Therefore, we use the background motions estimated from multiple previous frames to obtain more accurate background subtracted image.

We now denote H_{t-1} as the perspective transform between two previous frames from X_{t-2} to X_{t-1}. Then, we compute the background subtracted image E_{t-1} for X_{t-1} by taking average of forward and backward tracing.

$$E_{t-1} = \frac{1}{2}|X_{t-1} - H_{t-1} \circ X_{t-2}| + \frac{1}{2}|X_{t-1} - (H_t)^{-1} \circ X_t| \quad (6)$$

where $(H_t)^{-1}$ is the inverse transform of H_t. It is worth noting that we compute the background subtracted image for the previous frame X_{t-1} for symmetry.

2) Find Salient Points on Moving Objects: The moving objects are highlighted in the background subtracted image E_{t-1}. Then, we need to find the corresponding regions in X_t to detect moving objects in the current frame. Toward this, we first extract Shi-Tomasi corner points in E_{t-1} (refer Section III-A.1) and propagate them to appearance image X_{t-1}. For each propagated corner point from X_{t-1}, we find the corresponding point in X_t by applying Lucas-Kanade method (refer Section III-A.2).

We now denote q_{t-1} as the corner point in X_{t-1} propagated from E_{t-1}. Then, the local motion field v_t is computed as following:

$$v_t = \arg\min_{v} \sum_{s \in \mathcal{N}(q_{t-1})} |X_t(s+v) - X_{t-1}(s)|^2 \quad (7)$$

where $\mathcal{N}(q_{t-1})$ is the neighborhood around q_{t-1}. It is worth noting that we do not use the background subtracted image but the appearance image to estimate the local motion.

3) Prune Salient Points based on Motion Difference: We now have the corresponding points in X_t from E_{t-1}, which can be used to detect moving objects. However, we may have points on spurious noise (i.e. edge of background) due to incorrect background motion estimation.

Therefore, we prune the points according to the difference between the estimated background and local motion. This is based on the assumption that target object has very different motion compared with background. We now define the motion difference d_t between the background and moving object as following:

$$d_t = h_t - v_t \quad (8)$$

where h_t is interpolated motion vector from the perspective transform H_t at the point q_{t-1}. We then find the pruned point r_t according to the magnitude of motion difference.

$$r_t = q_{t-1} + v_t \quad \text{if } ||d_t||_2 > T \quad (9)$$

where T is the empirical threshold for pruning.

By applying connected component labeling [19] on the set of pruned points, we can cluster them according to

spatial proximity. We generate the bounding box for each cluster of points which represents our detection for a single moving object.

C. Target Classification and Tracking

While we expect our moving object detection to be effective, we still encounter false alarms among the detected objects. Therefore, we obtain a set of spatio-temporal features for each detected object and determine whether the object is our target or not. In addition, in order to prevent intermittent miss-detection and false alarm, we apply a tracking technique enforcing coherent temporal signatures of detection.

1) Classify Target Objects: Given the detected objects, we perform classification to reject outliers from true targets. We now denote $\mathbf{R}_t^{(n)}$ as a cluster of points to represent the n^{th} object in X_t. Then, we compute the two features which encode spatio-temporal characteristics of the object.

The first feature characterizes the coherency of motion difference vectors in $\mathbf{R}_t^{(n)}$. Here, we assume that the target is non-deformable object, and thus the motion vectors on the target object are consistent. We now define the feature $f_t^{(n)}$ as the angle variance of motion difference vectors.

$$\begin{aligned} f_t^{(n)} &= \frac{\sum_{d_t \in \mathbf{D}_t^{(n)}} |\arctan d_t - \mu_\theta^{(n)}|^2}{S_t^{(n)}} \\ \mu_\theta^{(n)} &= \frac{\sum_{d_t \in \mathbf{D}_t^{(n)}} \arctan d_t}{S_t^{(n)}}, \end{aligned} \tag{10}$$

where $\mathbf{D}_t^{(n)}$ is the set of motion difference vectors for $\mathbf{R}_t^{(n)}$ and $S_t^{(n)}$ is the number of points in $\mathbf{R}_t^{(n)}$.

The second feature characterizes the spatial distributions of points in the object. Here, we assume that there are densely distributed salient points for the target object. The feature $g_t^{(n)}$ is then defined as the point density in $\mathbf{R}_t^{(n)}$.

$$g_t^{(n)} = \frac{S_t^{(n)}}{B_t^{(n)}} \tag{11}$$

where $B_t^{(n)}$ is the area of minimum bounding box that encloses all points in $\mathbf{R}_t^{(n)}$.

Given these features, we build a classifier to identify target objects. We denote $y_t^{(n)}$ as a classification label where the positive value indicates that n^{th} object is the target. Then, the target classifier is defined as following:

$$y_t^{(n)} = \begin{cases} 1, & \text{if } f_t^{(n)} < T_1 \text{ and } g_t^{(n)} > T_2 \\ -1, & \text{otherwise} \end{cases} \tag{12}$$

where T_1 and T_2 are the empirical thresholds for angle variance and point density in $\mathbf{R}_t^{(n)}$.

2) Track Target Objects: Even though our target classifier reduces the false alarms, we also expect to have intermittent miss-detections and false alarms. These intermittent miss-detections and false alarms can be corrected by observing the temporal characteristics of the detected objects. Therefore, we apply tracking techniques to enforce coherent temporal signatures of detected objects. Specifically, we use the Kalman filter [11] for object tracking.

Kalman filter predicts the current state b_t from previously estimated states $\hat{b}_{t-1}$ with transition model and updates the current measurement c_t with the current state b_t as below:

$$\begin{aligned} b_t &= A\hat{b}_{t-1} + \omega_t \\ c_t &= Mb_t + \varepsilon_t \end{aligned} \tag{13}$$

where A is state transition matrix, ω_t controls the transition modeling error, M is measurement matrix, and ε_t represents the measurement error. The estimated output $\hat{b}_t$ is then computed with Kalman gain K:

$$\begin{aligned} \hat{b}_t &= A\hat{b}_{t-1} + K(c_t - Mb_t) \\ K &= V_\omega M^T (MR_\omega M^T + V_\varepsilon) \end{aligned} \tag{14}$$

where V_ω and V_ε are the covariance of ω_t and ε_t, separately.

In our application, we assign the size and location of bounding box for the detected object as state variable b_t and use the constant velocity model to set A and M. To initialize the Kalman filter, we find the corresponding objects from optical flow matching in L previous frames and start track if the classification labels $y_{t-1}^{(n)}, \cdots, y_{t-L}^{(n)}$ are consistent. Then, we recover the miss-detection for the positive label track and delete the false alarm of the negative label track based on the Kalman filter output at the current frame. We dismiss the track if we do not have detected objects in the Kalman filter estimation for L frames.

IV. EXPERIMENTS

We evaluate our multi-target detection and tracking algorithm for UAVs on a video data set provided by Naval Postgraduate School. The videos are taken in outdoor environment including real-world challenges such as illumination variation, background clutter, and small target objects.

A. Experimental Setup

1) Data Set: The data set comprises 5 video sequences of 1829 frames with 30 fps frame rate. They are recorded by a GoPro 3 camera (HD resolution: 1920×1080 or 1280×960) mounted on a custom delta-wing airframe. As a preprocessing, we mask out the pitot tube region which is not moving in the videos. For each video, there are multiple target UAVs (up to 4) which have various appearances and shapes. We manually annotate the targets in the videos by using VATIC software [21] to generate ground-truth dataset for performance evaluation.

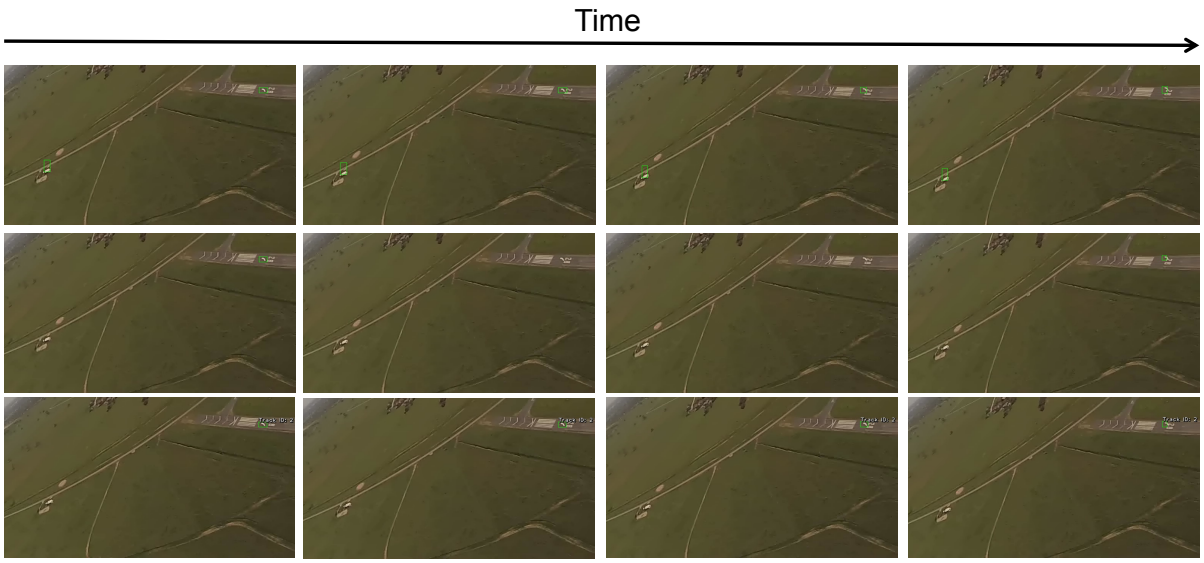

Fig. 3. Results of multi-target detection and tracking algorithms for four consecutive frames: [*Top-Row*] Background subtraction method [20], [*Middle-Row*] Our method with target classification only, [*Bottom-Row*] Our method with target classification and tracking; Green boxes represent the detected objects. Background subtraction method (*Top-Row*) detects false alarms on the complex background (i.e. building). By using the target classifier (*Middle-Row*), we reject false alarms but miss the detection on target UAVs occluded by background. Target tracking (*Bottom-Row*) enforces temporal consistency of our detection recovering intermittent miss-detection. Images are zoomed for better display. See full images in supplementary files.

2) Parameter Exploration: There are important parameters in our multi-target detection and tracking algorithm. To begin with, we extract Shi-Tomasi corner points in original image X_{t-1} (for background motion estimation in III-A.1) and background subtracted image E_{t-1} (for moving object detection in III-B.2), respectively. We set higher saliency threshold ($Q_E = 0.15$) for E_{t-1} than that ($Q_X = 0.001$) for X_{t-1} as we find the sparser set of points in the background subtracted image where only moving objects should be identified. In addition, we use 15×15 block size for Lucas-Kanade optical flow matching (Section III-A.2 and III-B.2). Next, we set the threshold $T = 1.8$ to prune the points with large motion difference (Section III-B.3) and the thresholds $T_1 = 5$ and $T_2 = 0.02$ for target classifier with angle variance and point density features (Section III-C.1). Finally, we use $L = 6$ for Kalman filter where we start the track if we detect the object in six previous frames (Section III-C.2).

B. Quantitative Evaluation

The overall goal of this experiment is to measure the detection accuracy of identifying targets in videos. We also analyze the computational time as our algorithm needs to be run on board efficiently.

TABLE I
DETECTION ACCURACY

	F
Background Subtraction [20]	0.552
Target Classification Only (Ours)	0.777
Target Classification / Tracking (Ours)	0.866

1) Detection Accuracy: To measure detection accuracy, we report F-score which is the harmonic mean of recall and precision rates computed as following:

$$\text{Recall} = \frac{\text{Number of Detected Targets in all Frames}}{\text{Number of Ground-Truth Targets in all Frames}}$$

$$\text{Precision} = \frac{\text{Number of Detected Targets in all Frames}}{\text{Number of Detected Objects in all Frames}}$$

$$F = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall + Precision}}$$

Here, we define the detected target if our detection has overlap with ground truth.

We compare our proposed method with the state-of-the art background subtraction method [20] which was developed for pedestrian detection with a static camera. We also report detection accuracy only with our target classification to highlight the importance of tracking. Table I summarizes the accuracy scores. The background subtraction method shows low F-score. This is because fast moving background in the video from UAV causes many false alarms. Our method significantly improves F-score indicating that our target classifier based on background and motion difference correctly identify target objects. By using our target tracking, F-score was further improved due to reduced intermittent miss-detections and false alarms.

2) Computational Time: We run our algorithm on a standard 3.5GHz clock rate Intel processor desktop with 8GB memory. We implement single-threaded Python codes with OpenCV library. The average computational time for each frame is $112.06 \pm 23.36ms$. The main computational bottleneck is to compute the background subtracted image since applying a global transformation to the large HD

Fig. 4. Shi-Tomasi corner points detected from background subtracted images in two videos: Red and green dots represent pruned and deleted points based on the magnitude of motion difference vector, respectively. We preserve the points around target UAV (red), while deleting the points located at corner of building structures (green). This indicates that the magnitude of difference vector between estimated background and local motion is effective to separate the points in the targets from complex backgrounds. Images are cropped for better display.

image (1920×1080 or 1280×960) is a heavy computational burden. By down-sampling the video by factor of 2 and using multi-thread implementation, our algorithm is efficient enough to run on board near real-time.

C. Visual Inspection

We complement the quantitative evaluation above with qualitative visual inspection. Fig. 3 shows exemplar results from our method with target classifier only and with tracking. For reference, we also illustrate the result with background subtraction method [20]. We notice that background subtraction method generates false alarms on complex backgrounds such as buildings. Our method rejects most false alarms thanks to target classifier based on motion difference but misses the detection on targets due to background clutter. Our Kalman filter tracking significantly improves the detection on targets by enforcing temporal consistency of the detection.

In Fig. 4, we display the Shi-Tomasi corner points detected from background subtracted images. We use different colors for preserved (red) and deleted (green) points by applying thresholding on the magnitude of motion difference. We observe that preserved and deleted points are mostly located around the target UAV and building structures, respectively. This reflects that we supplement the background subtraction by pruning the points based on motion difference, improving the detection accuracy.

V. CONCLUSIONS

In this paper, we proposed a multi-target detection and tracking algorithm for UAVs. Our method first estimates the background motion from a fast moving camera via perspective transformation model. We then find the sparse set of salient points from background motion compensated image and estimate local motion on those points through optical flow matching. By comparing the difference between background and local motions, we identify candidate moving objects and classify whether each moving object is target or not. We further refine the detection using temporal information from Kalman filter, reducing intermittent miss-detections and false alarms. Experimental results on actual videos taken from UAVs show that the proposed method can achieve high accuracy scores in terms of recall and precision rates and be efficient to run on board for collision avoidance system.

REFERENCES

[1] P. S. Lin, L. Hagen, K. Valavanis, and H. Zhou, "Vision of unmanned aerial vehicle (UAV) based traffic management for incidents and emergencies," in *World Congress on Intelligent Transport Systems*, 2005.

[2] M. Neri, A. Campi, R. Suffritti, F. Grimaccia, P. Sinogas, O. Guye, C. Papin, T. Michalareas, L. Gazdag, and I. Rakkolainen, "SkyMedia - UAV-based capturing of HD/3D content with WSN augmentation for immersive media experiences," in *IEEE International Conference on Multimedia and Expo*, 2011.

[3] F. Nex and F. Remondino, "UAV for 3D mapping applications: a review," *Applied Geomatics*, vol. 6, no. 1, pp. 1–15, 2013.

[4] D. Xing, D. Xu, and F. Liu, "Collision Detection for Blocking Cylindrical Objects," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[5] C. Hane, C. Zach, J. Lim, A. Ranganathan, and M. Pollefeys, "Stereo Depth Map Fusion for Robot Navigation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[6] S. Roelofsen, D. Gillet, and A. Martinoli, "Reciprocal collision avoidance for quadrotors using on-board visual detection," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015.

[7] N. Oliver, B. Rosario, and A. Pentland, "A Bayesian Computer Vision System for Modeling Human Interactions," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 831–843, 2000.

[8] L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "PIXHAWK: A System for Autonomous Flight Using Onboard Computer Vision," in *IEEE International Conference on Robotics and Automation*, 2011.

[9] I. Carlbom and J. Paciorek, "Planar Geometric Projections and Viewing Transformations," *ACM Computing Surveys*, vol. 10, no. 4, pp. 465–502, 1978.

[10] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," in *International Joint Conference on Artificial Intelligence*, 1981.

[11] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," University of North Carolina at Chapel Hill, Tech. Rep., 1995.

[12] A. Rozantsev, V. Lepetit, and P. Fua, "Flying Objects Detection from a Single Moving Camera," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[13] P. Dollar, Z. Tu, P. Perona, and S. Belongie, "Integral Channel Features," in *British Machine Vision Conference*, 2009.

[14] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust Object Recognition with Cortex-Like Mechanisms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 411–426, 2007.

[15] A. Bosch, A. Zisserman, and X. Munoz, "Image Classification Using Random Forests and Ferns," in *International Conference on Computer Vision*, 2007.

[16] N. Seungjong and J. Moongu, "A New Framework for Background Subtraction Using Multiple Cues," in *Asian Conference on Computer Vision*, 2013.

[17] T. Brox and J. Malik, "Large Displacement Optical Flow: Descriptor Matching in Variational Motion Estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 500–513, 2011.

[18] J. Shi and C. Tomasi, "Good features to track," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1994.

[19] H. Samet and M. Tamminen, "Efficient Component Labeling of Images of Arbitrary Dimension Represented by Linear Bintrees," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 4, pp. 579–586, 1988.

[20] Z.Zivkovic and F. der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 773–780, 2006.

[21] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently Scaling up Crowdsourced Video Annotation," *International Journal of Computer Vision*, vol. 101, no. 1, pp. 184–204, 2012.