Moving Objects Detecting and Tracking for Unmanned Aerial Vehicle

Binpin Su, Honglun Wang, Xiao Liang and Hongxia Ji

Abstract Moving objects detecting and tracking is important for future Unmanned Aerial Vehicles (UAVs). We propose a new approach to detect and track moving objects from the flying UAV. First, estimate the global-motion of the background by tracking features selected by KLT algorithm from frame to frame. In order to avoid features located on the foreground objects participating in motion estimation, feature effectiveness evaluation is employed. Then compensate the background with the transform model computed by RANSAC. Define the undefined area before applying frame difference method to the compensated frame and the current frame. Then initialize the tracking module with information obtained from the detecting module, which overcomes shortcomings of artificial orientation of traditional tracking algorithms. For tracking fast and robustly from UAVs, we design a new tracking algorithm by fusing Kalman prediction and Mean Shift Search together. The experimental results presented effectiveness of the whole detecting and tracking approach.

Keywords Object detecting · Object tracking · Global-motion estimation · Mean Shift · Kalman prediction

1 Introduction

Recently, UAVs have been used more and more widely for news gathering, search, and rescue both in military and civil areas, especially in a reconnaissance and strike integrated system by aerial surveillance. Aerial surveillance is performed primarily using video cameras these days. Exploitation of UAV video data is increasingly

B. Su (✉) · H. Wang · X. Liang · H. Ji
Science and Technology on Aircraft Control Laboratory, Beijing University of Aeronautics and Astronautics, Beijing 100191, China
e-mail: binpin.su@gmail.com

critical for surveillance systems. Large numbers of videos from UAVs will overwhelm human operators for visually inspecting the data, so automatic video analysis and processing is essential, which will not only reduce the workload for humans but will also improve the fully-automatic surveillance ability for future UAVs.

Moving objects detecting and tracking for UAVs is a subject of active research. The detection of an interesting moving object is based on the motion information such as optical flow or background subtraction. Tracking is to maintain a consistent identity on the object based on the appearance, shape, or kinematic information over the frames. A large amount of research on moving-object detecting and tracking in a video has been performed in the past years, and a variety of methods have been developed, which include region-based [1–3], feature-based [4–6], and contour-based [7, 8] methods. Although many methods have been proposed, most can only be used for specific applications with reliable assumptions, such as a stationary application in a constrained and uncluttered environment. For UAV videos captured by a fast-moving camera, moving objects detecting and tracking has the following key challenges.

The camera is mounted on the moving UAV, so there are two independent motions involved: the motion of moving objects and the motion of camera. Unfortunately, those two motions are blended together. In order to detect moving objects robustly from the moving background, it should be able to decompose these two independent motions from sensor readings.

A UAV may move very fast, and an object may also move fast. UAV video objects tend to have low resolution and the background is usually cluttered. There are various types of noise added at various stages, including changing lighting conditions, changing appearance, rotations, and scales of objects. Furthermore, UAV objects may be so small that they are similar to noise, and may suffer partial or total occlusions.

UAV video analysis requires fast tracking of moving objects. Hence a compromise typically exists between the achieving of real-time tracking performance and the constructing of a robust tracking algorithm.

Frame difference, which compares two consecutive image frames and finds moving objects based on the difference, is perhaps the most intuitive and fastest algorithm for moving object detection, especially when the viewing camera is static. However, for UAV videos, straightforward differencing is not applicable because a large difference is generated by simply moving the camera even if nothing moves in the environment. The global-motion of the background should be eliminated so that the remaining motions, which are due to moving objects, can be detected.

To overcome the first problem mentioned above, various methods have been proposed to stabilize camera motions by tracking features [9] and computing optical flow [10]. These approaches focus on estimating the transformations between two image coordinate systems. The features associated with the foreground objects should be eliminated since they will lead to wrong estimation of the global-motion of the background.

Once moving objects has been identified, they need to be tracked. Two major components can be distinguished in a typical visual tracker [11]. Target Representation and Localization is mostly a bottom-up process which copes with the changes in the appearance of the target. Filtering and Data Association is mostly a top-down process dealing with the dynamics of the tracked object, learning of scene priors and evaluation of different hypotheses by using a Bayesian filter. The way the two components are combined and weighted is application dependent. They play a decisive role in the robustness and efficiency of the tracker.

In this paper we propose an approach to detect and track moving objects for UAV. The block diagram of the proposed approach is depicted in Fig. 1. It includes three steps: compensation of the image flow induced by a moving camera, detection of moving regions in each frame, and tracking of moving objects in time.

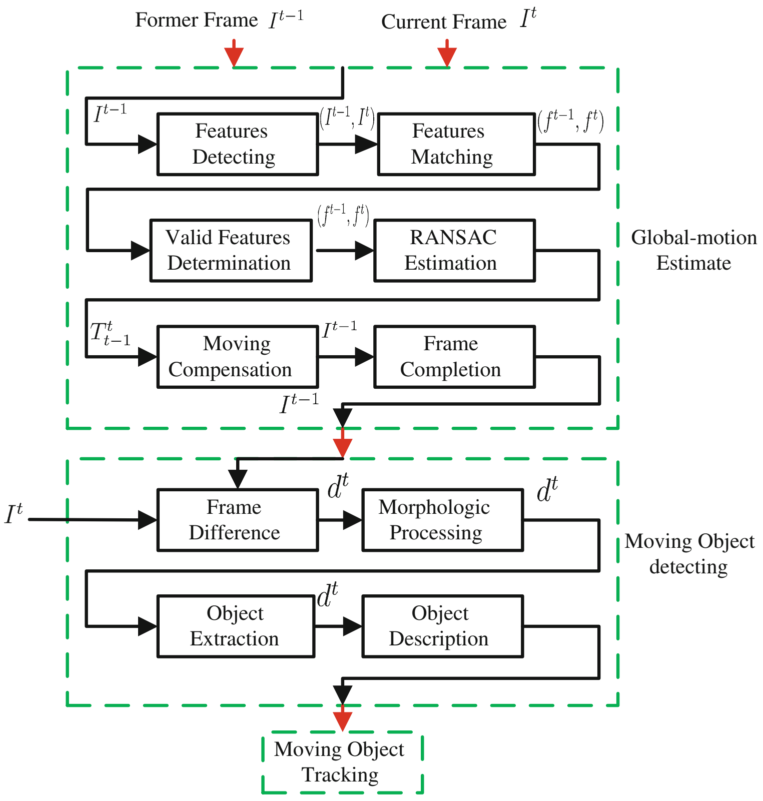


Fig. 1 The proposed detecting and tracking approach

2 Global-Motion Estimation

The motion induced by a moving camera must be canceled before motion detection. The global-motion estimation can be estimated by tracking features between images. When the camera moves, two consecutive images, I^t (the image at time t) and I^{t-1} (the image at time $t-1$), are in different coordinate systems. Global-motion estimation is a transformation from the image coordinates of I^{t-1} to that of I^t so that the two images can be compared directly. The transformation can be estimated using two corresponding feature sets: a set of features f^{t-1} in I^{t-1} and a set of corresponding features f^t in I^t.

2.1 *Feature Detecting and Matching*

In general, two basic questions must be answered: how to select features and how to track them from frame to frame. Two most successful visual features are corner [12] and SIFT (Scale Invariant Scale Transform) [13] features. A corner feature is characterized by the high intensity changes in both horizontal and vertical directions. SIFT feature is a more advanced visual feature, which is known to be relatively invariant to image translation, scaling, and rotation and partially invariant to changes in illumination and local image deformations. Considering the computational intensive, it is not suitable for real-time applications, unless the algorithm is implemented in hardware. We adopt the KLT [4] feature (a famous corner feature) selection algorithm for corresponding feature set selection due to its computation efficiency. The feature selection algorithm runs on image I^{t-1}, and generates features f^{t-1}. A corresponding feature set f^t is constructed by tracking the same features on the image I^t. Figure 2a shows the outdoor environment. Figure 2b shows the feature selected from the image I^t, and Fig. 2c shows the same features tracked over 5 frames on the image I^{t+5}, while Fig. 2d implies the optical flow between the two frames.

2.2 *Valid Feature Determination*

Once the correspondence $f = \left\{f^{t-1}, f^t\right\}$ is known, the geometric transform to align $I^{t-1}$and I^t can be estimated using a transformation model. However, as mentioned before, some of the features associated with the foreground moving objects may be included, which will lead to an inaccurate estimation of the global-motion of background. Thus, those features should be eliminated from the feature set before the transform is computed. We name features belonging to background as valid features (or insiders) while foreground features are invalid features (or outliers).

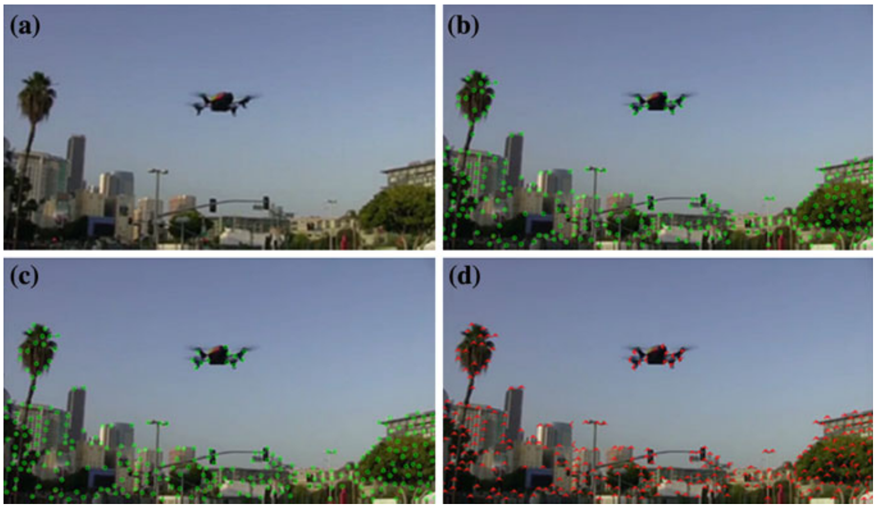


Fig. 2 **a** out door environment **b** features detected on image It **c** features detected on image It+5 **d** optical flows between the two images

Since features belonging to the background move in a uniform way, assume that the distribution of the optical flow vectors follows Gaussian distribution. The probability out of the range of $[\mu - 3\sigma, \mu + 3\sigma]$ is very small according to the Gaussian theory, where μ, σ denote the expected value and variance respectively. When the vector of two corresponding features is out of the range, the features are probably belonging to foreground objects. Let V_i represent the optical flow vector of a selected feature between two consequent frames. $\|V_i\|$, $Ang(V_i)$ denotes its vector norm and vector direction, respectively, while $\left(\mu_{\|\cdot\|}, \sigma_{\|\cdot\|}\right)$ and $\left(\mu_{Ang}, \sigma_{Ang}\right)$ represent the expected value, variance of the norm, and direction angle of all selected features respectively. The determinant condition (outlier algorithm) can be defined as follows:

$$\begin{cases} f_i \in F_{\text{in}} & \text{if } \left| \|V_i\| - \mu_{\|\cdot\|} \right| < 3 \cdot \sigma_{\|\cdot\|} \text{ and } \left| Ang(V_i) - \mu_{Ang} \right| < 3 \cdot \sigma_{Ang} \\ f_i \in F_{\text{out}} & \text{otherwise} \end{cases} \tag{1}$$

where F_{in} and F_{out} denote valid and invalid features respectively. Figure 3 shows the result of the valid feature determination algorithm. F_{in} is marked with red circles and F_{out} is marked with green circles. In the frame, the aircraft is moving while the buildings and trees are static. All the features associated with the aircraft are detected and eliminated. Note that the valid feature determination will be violated when moving objects are very close to the camera since they will occupy most of the areas of an image and the features optical flow will not conform to Gaussian distribution.

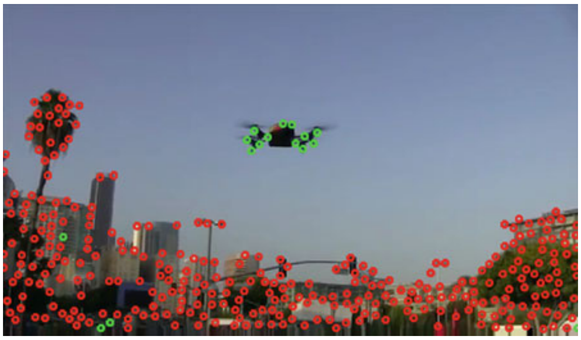


Fig. 3 Inliers (*red circle*) and outliers (*green circles*)

2.3 Transform Model

Using the remaining insiders (valid features set), we can compute the relatively accurate transformation model. The most-used models include affine model, bilinear model, and pseudo-perspective model. When the interval between consecutive images is very small, most global-motion can be estimated using an affine model, which can cover translation, rotation, shearing, and scaling motions. Hence, we use an affine model here:

$$\begin{bmatrix} f_x^t \\ f_y^t \end{bmatrix} = \begin{bmatrix} af_x^{t-1} + bf_y^{t-1} + t_1 \\ cf_x^{t-1} + df_y^{t-1} + t_2 \end{bmatrix} \tag{2}$$

We compute the transformation model T_{t-1}^t by using RANSAC (RANdom SAmple Consensus) algorithm [14], an iterative method, to estimate the parameters of the affine model due to its ability for robust estimation.

2.4 Moving Compensation

For frame differencing, image I^{t-1} is converted using the transformation model before being compared to the image I^t in order to eliminate the effect of global-motion. For each pixel (x, y)

$$I_{\text{comp}}^{t-1}(x, y) = I^{t-1}\left(\left(T_{t-1}^t\right)^{-1}(x, y)\right) \tag{3}$$

where I_{comp}^{t-1} represents the compensation of global-motion of image I^{t-1}.

3 Moving Object Detecting

The difference image between two consecutive images is computed using the compensated image:

$$I_{\text{diff}}(x,y) = \left| I_{\text{comp}}^{t-1}(x,y) - I^{t}(x,y) \right| \tag{4}$$

Note that some pixels around the border moved out of the frame after compensation, causing undefined area in the compensation frame. To obtain a complete frame, we redefine the difference frame as follows:

$$I_{\text{diff}}(x,y) = \begin{cases} \left| I_{\text{comp}}^{t-1}(x,y) - I^{t}(x,y) \right| & \text{if}(x,y) \in \text{definedarea} \\ 0 & \text{if}(x,y) \in \text{undefinedarea} \end{cases} \tag{5}$$

Figure 4 compares the result of two cases: frame difference without global-motion compensation (Fig. 4a) and with global-motion compensation (Fig. 4b).

With the image processes introduced above, we can detect the moving object in theory, but in fact due to illumination and background variance, there are noises in the frame. Hence, we continue to perform a morphologic closing operation on the result to remove camera noise and to connect object pixels into regions. The morphologic process include erosion and dilation algorithm, which define as follows:

$$A \odot B = \left\{ x \middle| (B)_x \subseteq A \right\} \tag{6}$$

$$A \oplus B = \left\{ x \middle| \left(\hat{B}\right)_x \cap A \neq \varnothing \right\} \tag{7}$$

where A is the image to be processed and B is the erosion matrix of 20×20 pixels. The resultant image is then size-filtered. Only connected regions of size bigger than a threshold are kept. Figure 5 shows the difference of the difference image with noise and with morphologic process. We can extract the detecting moving object clearly after morphologic.

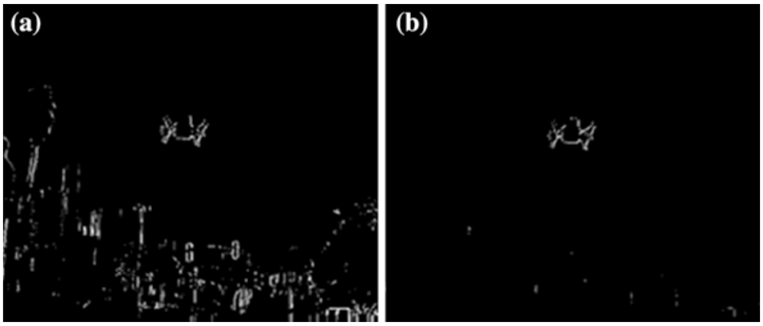


Fig. 4 **a** frame difference without global-motion compensation **b** frame difference with global-motion compensation

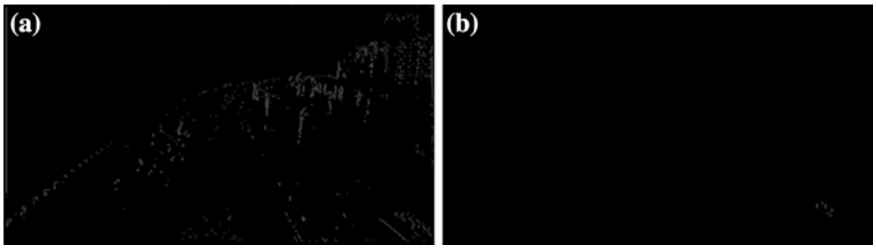


Fig. 5 **a** frame difference without morphologic process **b** frame difference with morphologic process

4 Object Tracking

In the previous work, tracking object was always initialized with a hand-drawn region. As in the surveillance of the aerial video, it is difficult to point the exact interesting region due to the fast video speed, and the region including noise and unrelated information will lead to wrong track. In this paper we propose an approach to initialize the region by the information about the moving object obtained from the object detecting module, which overcomes shortcoming of artificial orientation.

Although KLT algorithm can track feature points, it cannot identify which feature point belongs to the target we want to track. Hence, we choose other feature space. Mean Shift, which is based on the motion of "kernel", is an efficient technique that can automatically sort out local model within a set of data. Here, we employ the Mean Shift algorithm which chooses color information as the object feature.

4.1 Object Tracking Using Mean Shift Algorithm

The Mean Shift tracking algorithm is divided into three parts: target representation, similarity function measure, and target location computation [15]. First, the reference target model is represented by its probability density function (pdf) $\hat{q} = \{\hat{q}_u\}_{u=1\ldots m}, \sum_{u=1}^{m} \hat{q}_u = 1$. A target candidate which is defined at location y is characterized by the pdf $\hat{p}(y) = \{\hat{p}_u(y)\}_{u=1\ldots m}, \sum_{u=1}^{m} \hat{p}_u = 1$. m represents m-bin histograms. Second, the similarity between the target model and the target candidates in the next frame is measured using the metric derived from the Bhattacharyya coefficient $\hat{\rho}(y) \equiv \rho[\hat{p}(y), \hat{q}] = \sum_{u=1}^{m} \sqrt{\hat{p}_u(y)\hat{q}_u}$. The coefficient determines whether it is the most similar result to a given model, if not then search around. The bigger the value, the more similar the model is to the target. The search procedure uses gradient information which is provided by the Mean Shift vector. Mean Shift was originally presented in 1975 by Fukunaga and Hostetler

[16]. We use Mean Shift iterations to search for the maxima density function of the coefficient, then we can obtain the most reliable candidate. The Mean Shift algorithm calculates local optimal result fast and effectively which is superior to blindness search. Besides color feature using for tracking is simple and effective in most situations.

However, the Mean Shift tracking algorithm has some shortcomings. It cannot be applied to track fast-moving objects. Because the most reliable location y was obtained after some manipulations using Taylor expansion around the probability location $\hat{y}_0$ on $\hat{\rho}(y)$:

$$\rho[\hat{p}(y), \hat{q}] \approx \frac{1}{2}\sum_{u=1}^{m} \sqrt{\hat{p}_u(\hat{y}_0), \hat{q}_u} + \frac{1}{2}\sum_{u=1}^{m} \hat{p}_u(\hat{y})\sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{y}_0)}} \tag{8}$$

When the target moves fast, the new location is far from the old one which is not fit for neighborhood expansion analysis.

Besides, the feature color, being described as the target and candidates in Mean Shift tracking algorithm, is a relative weak feature. When some noises which are similar to the target appear in the background, it will cause wrong tracking.

4.2 Kalman Prediction

Mean Shift tracking algorithm is a bottom-up appearance-based tracking method, while Kalman is totally a top-down process. Kalman is an optimal recursive data processing algorithm, which consists of two steps: prediction and correction. In the former step, the state is predicted with the dynamic model, while in the latter step, it is corrected with the observation model. Since the error covariance of the estimator is minimized, it can be regarded as an optimal estimator.

As for tracking, the information characterizing the target is defined by the state sequence $\{X_k\}_{k=0,1,\ldots} = \{x_k, y_k, v_{xk}, v_{yk}\}$, representing the target's location and its velocity, whose evolution in time is specified by the dynamic equation $X_k = FX_{k-1} + w_k$. The available measurements $\{Z_k\}_{k=1,\ldots} = \{x_k, y_k\}$ are related to the corresponding states through the measurement equation $Z_k = HX_k + v_k$. The matrix F is called the system matrix and H is the measurement matrix. The noise sequence w_k and v_k are Gaussian, $w_k \sim N(0, \sigma_{\mathrm{w}}^2)$, $v_k \sim N(0, \sigma_{\mathrm{v}}^2)$.

Considering a target may not always runs with uniform velocity, it moves fast and sometimes slowly, turns around quickly, or brakes suddenly. We suppose that it moves in a linear motion with random acceleration noise. We design the state sequence updates in the following way:

$$x_k = x_{k-1} + \upsilon_{x_{k-1}} t + \frac{1}{2} w_k t^2 \tag{9}$$

$$y_k = y_{k-1} + \upsilon_{y_{k-1}} t + \frac{1}{2} w_k t^2 \tag{10}$$

$$\upsilon_{x_k} = \upsilon_{x_{k-1}} + w_k t \tag{11}$$

$$\upsilon_{y_k} = \upsilon_{y_{k-1}} + w_k t \tag{12}$$

where t denotes the interval frames. Hence, we design the dynamic equation and the measurement equation as follows:

$$\begin{pmatrix} x_k \\ y_k \\ \upsilon_{x_k} \\ \upsilon_{y_k} \end{pmatrix} = \begin{pmatrix} 1 & 0 & t & 0 \\ 0 & 1 & 0 & t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \\ \upsilon_{x_{k-1}} \\ \upsilon_{y_{k-1}} \end{pmatrix} + \begin{pmatrix} \frac{t^2}{2} \\ \frac{t^2}{2} \\ t \\ t \end{pmatrix} w_k \tag{13}$$

$$\begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \\ \upsilon_{x_{k-1}} \\ \upsilon_{y_{k-1}} \end{pmatrix} + \begin{pmatrix} 1 \\ 1 \end{pmatrix} v_k \tag{14}$$

where $\sigma_{\mathrm{w}} = \sigma_{\mathrm{v}} = 5$. The object of using Kalman prediction is to estimate the state X_k given all the measurement $Z_{1:k}$ up to that moment, or equivalently to construct the probability density function $p(x_k|Z_{1:k})$.

4.3 Fusing Kalman Prediction and Mean Shift Search

Considering the advantages and disadvantages of the Mean Shift and Kalman, we combine Mean Shift search and Kalman prediction together to track fast and robustly. The framework is shown in Fig. 6. Kalman predicts position of the target first, and Mean Shift searches in the confidence region estimated by Kalman which largely improves the search efficiency since it reduces the search area. The search result of Mean Shift is used in turn as the measurement value of Kalman to predict the position of the target in the next frame. As the search result of Mean Shift may not be accurate which will lead to unreliable prediction computed by Kalman, we add a determination. $(\hat{x}_k, \hat{y}_k)$ which represents predicted value of Kalman, while (x_k, y_k) denotes the search value of Mean Shift. The two values are highly similar in general and $e(k) = \sqrt{(x_k - \hat{x}_k)^2 + (y_k - \hat{y}_k)^2}$ are small. When the difference is large, we can infer that Mean Shift result is invalid so it cannot be used as the measure value of Kalman.

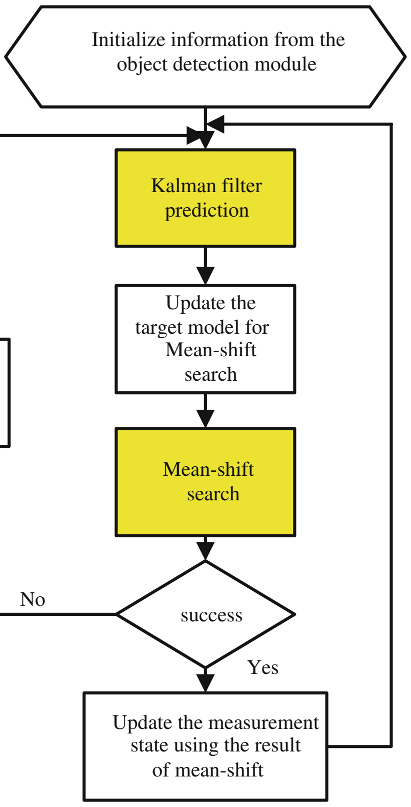

Initialize information from the object detection module
Kalman filter prediction
Update the target model for Mean-shift search
Mean-shift search
success
No
Yes
Using the prediction result as the measurement
Update the measurement state using the result of mean-shift

Fig. 6 New tracking algorithm of fusing Mean Shift search and Kalman prediction

5 Experiments

We have developed a system implementation for testing the detecting and tracking method, using visual C++ 2010 platform and OpenCV library. The computer processor is Intel Xeon @2.80 GHz.

5.1 Moving Object Detecting Result

The city video was taken from an airplane. The three rows shown in Fig. 7 represent detection results of frames 37, 41, and 44, respectively, which aims at detecting a moving car along the road. The first column shows the frame difference without global-motion compensation while the second column shows the frame difference with compensation. In the third column, we mark out the detected moving region with red rectangle. There is only one moving car in the scene. We compute the location and velocity of the moving region to prepare for the tracking module. The total process time is 215 ms.

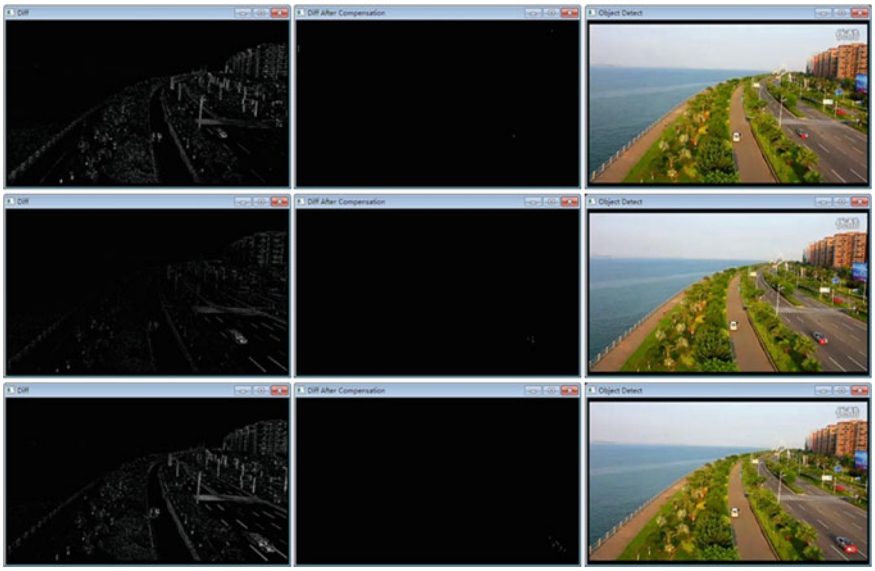


Fig. 7 Moving object detection from a moving camera. The frames 37, 41, and 44 are shown

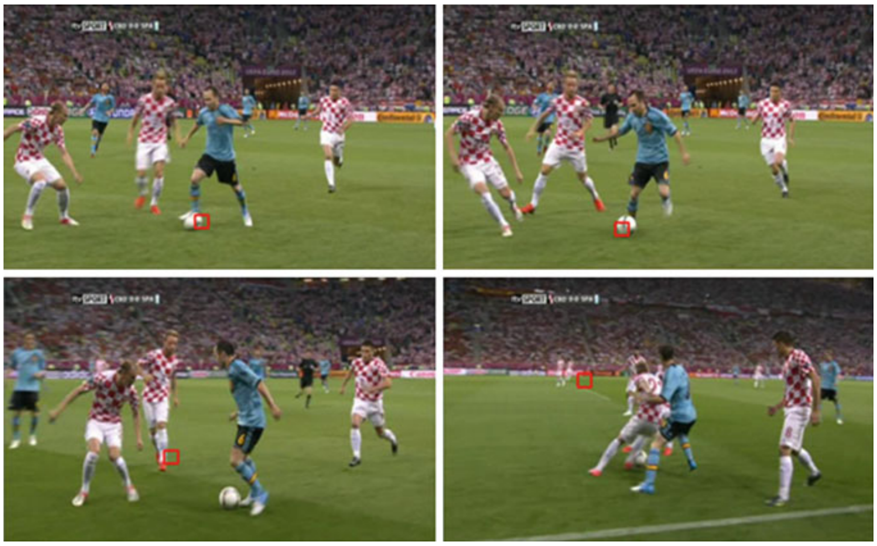


Fig. 8 Results of football tracking with Mean Shift

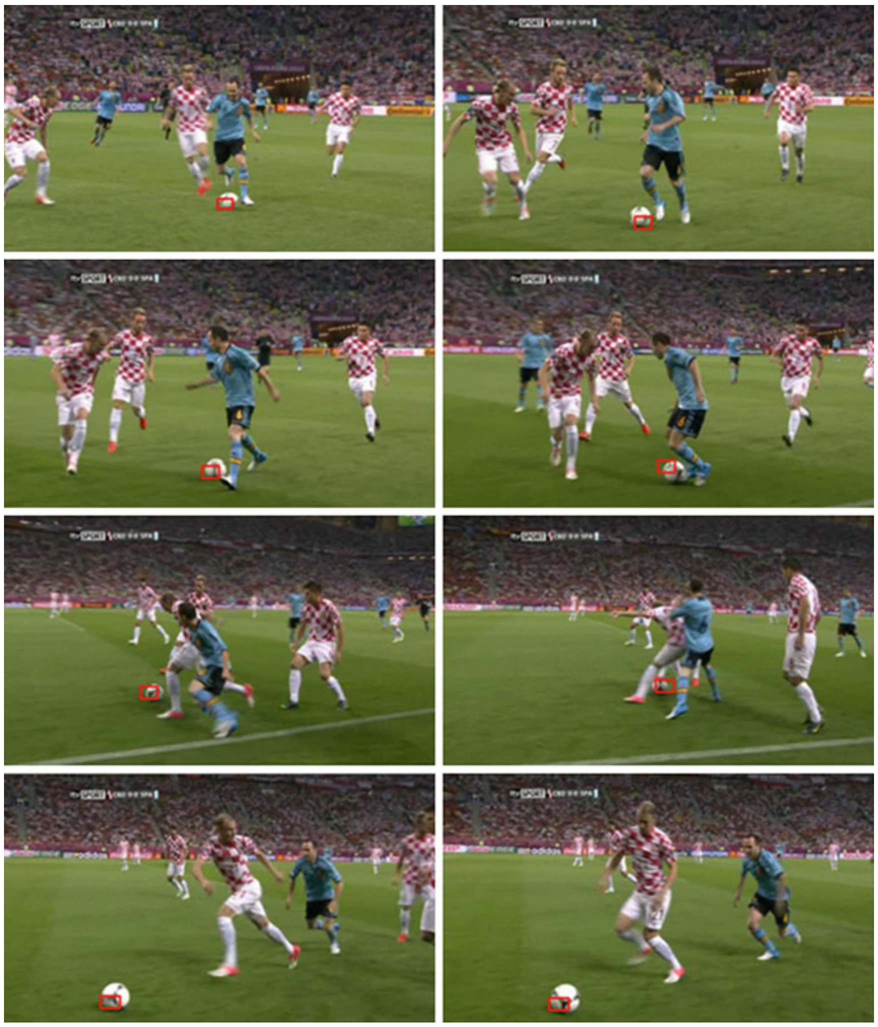


Fig. 9 Results of football tracking by fusing Mean Shift and Kalman. Frames 1, 28, 32, 45, 56, 69, 80, 88 (*left* to *right*, *up* to *down*) are shown

5.2 Moving Object Tracking Results

In a sequence of the game of football, the ball runs fast and moves in a random way. Note that the motion characters of the tracked targets from UAVs are similar to football, so we use the football sequence to test the performance of the tracking algorithms. When tracking the football using Mean Shift only (Fig. 8), it shows

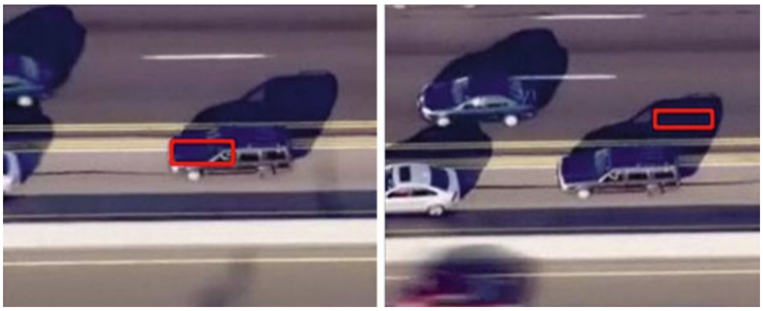


Fig. 10 Results of car tracking with Mean Shift

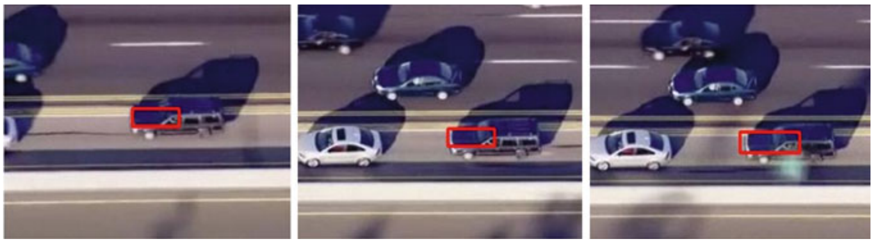


Fig. 11 Results of tracking by fusing Mean shift and Kalman. Frame1, 110, 171 (*left to right*) are shown

that the ball can be tracked well in the first few frames, but fails later due to its fast movement, just as the theory analyzed in the fourth part.

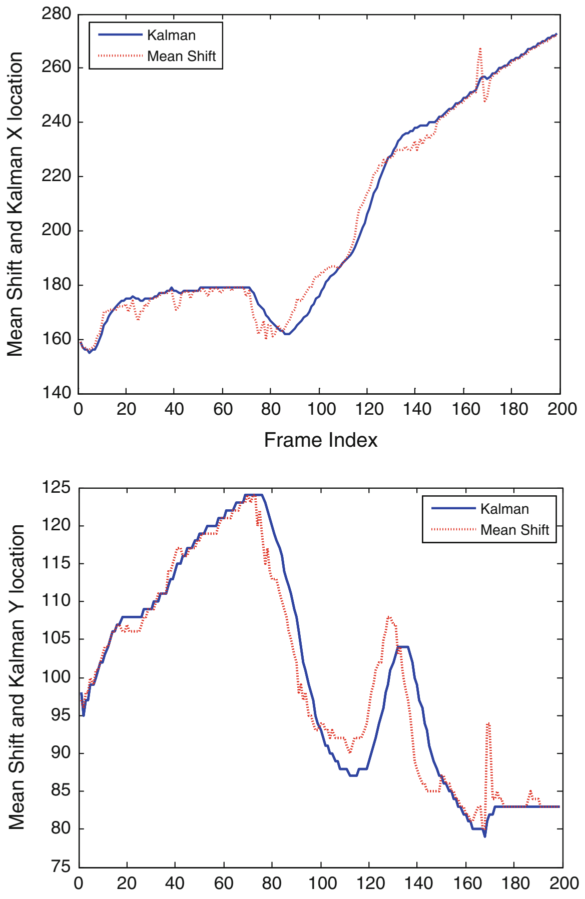
Figure 9 shows the tracking result of the running football in the same football sequence using the method of fusing Mean Shift search and Kalman prediction together. Whether the football runs relatively slowly which was brought along by the player (frame 1, 28, 32, 45), runs very fast because of being shoot (frame 80, 88), or partially occlusion (frame 69) by the dual meet players, it can all be tracked robustly.

In a car sequence (Fig. 10), which has 200 frames of 512*288 pixels, we intend to track the black car. Since the histogram of the shade is similar to the car itself, Mean Shift tracking converges to the shade.

Figure 11 shows the result of tracking the black car using the fusing method proposed in this paper. Observe that the overall algorithm is able to track the car in the presence of similar objects (car shadow in frame 110) and partially occlusion by the obstacle on the road (frame 171).

In a more concrete analysis of the car sequence, similar objects in the neighborhood and occlusion increase the measurement uncertainty. In Fig. 12, we present the measurements (dotted line) obtained by Mean Shift and the estimated locations of the object computed by Kalman (solid line) in the direction of X and Y

Fig. 12 Measurement and estimated state of car sequence



respectively. In most of the frames, two values are very close to each other. But the presence of car shadow from frame 72 to 149, which determines an increase in state uncertainty, give two values which vary. The sharp curve from frame 169 to 175 in the dotted line represents invalid search by Mean Shift caused by feature occlusion (see Fig. 11). By fusing Mean Shift search and Kalman prediction together, we can conduct a robust track.

Besides achieving a robust detecting and tracking method, we also care much about its real-time performance. When searching the target in one frame of 512*288 pixels, the average cost time using Mean Shift is 31 ms, while the time cost by fusing Mean Shift and Kalman is 26 ms. By fusing Mean Shift search and Kalman prediction together, we can also conduct a faster track. Thus the fusing track algorithm is fit for our aerial video tracking.

6 Conclusion and Future Work

A real-time method for moving object detecting and tracking on an unmanned aerial vehicle was introduced. The global-motion of the background was estimated using corresponding valid feature sets first. Then detect the moving object after compensating the consecutive frames. Use the location and velocity information of the moving object obtained from the object detecting module for object tracking module initialization. Finally we integrated the Mean Shift algorithm and the Kalman prediction to track objects fast and robustly. The results show that the method is general and computationally inexpensive which is fit for aerial video surveillance.

Additional research work is needed to improve the performance of the method, such as optimizing the object detection algorithm to minimize the computation cost. On the other hand, the current system can only estimate the location, direction and speed of the moving objects. Since a single camera has limit on retrieving depth information, the information from a camera alone is not rich enough to construct full 3-dimensional models of moving objects. We can use a laser rangefinder, which provides the depth information of a single plane.

Acknowledgments We appreciate the contributions made by the open source communities to the humanity. We especially thank the developers and sponsors of OpenCV libraries for their valuable work.

References

1. Wang D (1998) Unsupervised video segmentation based on watersheds and temporal tracking. Circuits Syst Video Technol IEEE Trans 8(5):539–546
2. Wong K, Spetsakis M (2002) Motion segmentation and tracking. In: International conference on vision interface, Citeseer, pp 80–87
3. Cavallaro A, Ebrahimi T (2000) Video object extraction based on adaptive background and statistical change detection. Proc of SPIE 4310:465
4. Shi J, Tomasi C (1994) Good features to track. In: Proceedings of the CVPR'94. IEEE computer society conference on computer vision and pattern recognition, pp 593–600
5. Tomasi C, Kanade T (1991) Detection and tracking of point features. Carnegie Mellon Univ, School of Computer Science
6. Trucco E, Petillot Y, Ruiz I, Plakas K, Lane D (2000) Feature tracking in video and sonar subsea sequences with applications. Comput Vis Image Underst 79:92–122
7. Jurie F (1998) Tracking objects with a recognition algorithm. Pattern Recogn Lett 19(3):331–340
8. Drummond T, Cipolla R (2002) Real-time visual tracking of complex structures. Pattern Anal Mach Intell IEEE Trans 24(7):932–946
9. Jung B, Sukhatme G (2004) Detecting moving objects using a single camera on a mobile robot in an outdoor environment. In: International conference on intelligent autonomous systems, Citeseer, pp 980–987

10. Srinivasan S, Chellappa R (1997) Image stabilization and mosaicking using the overlapped basis optical flow field. In: The proceedings of the IEEE international conference on image processing, vol 3, pp 356–359
11. Comaniciu D, Ramesh V, Meer P (2003) Kernel-based object tracking. Pattern Anal Mach Intell IEEE Trans 25(5):564–577
12. Harris C, Stephens M (1988) A combined corner and edge detector. In: Alvey vision conference
13. Lowe D (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vision 60(2):91–110
14. Xin S (2006) The matching method based on ransac algorithm for estimation of the fundamental matrix. J Shanghai Dianji Univ 9(004):66–69
15. Comaniciu D, Meer P (1999) Mean shift analysis and applications. In: The proceedings of the seventh IEEE international conference on computer vision, vol 2, pp 1197–1203
16. Fukunaga K, Hostetler L (1975) The estimation of the gradient of a density function, with applications in pattern recognition. Inf Theory IEEE Trans 21(1):32–40