Contents lists available at ScienceDirect

Image and Vision Computing

journal homepage: www.elsevier.com/locate/imavis

Adaptive pyramid mean shift for global real-time visual tracking

Shu-Xiao Li *, Hong-Xing Chang, Cheng-Fei Zhu

Institute of Automation, Chinese Academy of Sciences, Beijing, PR China

ARTICLE INFO

Article history:
Received 24 March 2008
Received in revised form 25 March 2009
Accepted 14 June 2009

Keywords:
Global visual tracking
Fast mean shift
Adaptive level
Kernel-based tracking
Tracking and pointing subsystem

ABSTRACT

Tracking objects in videos using the mean shift technique has attracted considerable attention. In this work, a novel approach for global target tracking based on mean shift technique is proposed. The proposed method represents the model and the candidate in terms of background weighted histogram and color weighted histogram, respectively, which can obtain precise object size adaptively with low computational complexity. To track targets whose displacements between two successive frames are relatively large, we implement the mean shift procedure via a coarse-to-fine way for global maximum seeking. This procedure is termed as adaptive pyramid mean shift, because it uses the pyramid analysis technique and can determine the pyramid level adaptively to decrease the number of iterations required to achieve convergence. Experimental results on various tracking videos and its application to a tracking and pointing subsystem show that the proposed method can successfully cope with different situations such as camera motion, camera vibration, camera zoom and focus, high-speed moving object tracking, partial occlusions, target scale variations, etc.

1. Introduction

Real-time visual tracking is the critical task in many computer vision applications such as driver assistance [1,28], aerial surveillance and exploitation [2,3], etc. The cameras used in these circumstances move with the carriers and may vibrate frequently. This demands a global search for target localization and tracking. Moreover, only a small percentage of the on-board processor's resources can be allocated for tracking, the rest being required for high-level tasks such as recognition, trajectory interpretation and threaten estimation, etc. Therefore, it is desirable to keep the computational consumption of a tracker as low as possible.

An abstract approach treats tracking as an inference problem [4]. The information characterizing the object is typically defined by the state sequence $\{\mathbf{x}_k\}_{k=0,1,...}$, whose evolution in time is specified by the dynamic model $\mathbf{f}_k$. The available measurements and its models can be represented by $\{\mathbf{z}_k\}_{k=0,1,...}$ and $\mathbf{h}_k$, respectively. Then a Bayesian method is employed to predict the next state and update the state when new measurements arrive. When $\mathbf{f}_k$ and $\mathbf{h}_k$ are linear functions, the optimal solution is provided by the Kalman filter (KF). When they are nonlinear, various techniques have been proposed. A direct approach linearizes the two models and this yields the Extended KF (EKF). A recent alternative to EKF is the Unscented KF (UKF) [5,6], which uses a set of weighted points by deterministic sampling rules to parameterize the mean and covariance of the posterior density. UKF has second order approximation and real-time performance on personal computer (PC), but it does not meet the sparse tracking need for on-board surveillances. The particle filter (PF), also known as the Condensation algorithm, constructs the probability density function in terms of a set of weighted particles through sequential importance sampling [7]. The main disadvantage of PF is the high computational complexity. A Markov chain Monte Carlo (MCMC) sampling step is developed by Khan et al. [8,9] to reduce the number of required particles to speed up PF for tracking multiple targets. Another trick for PF is to marginalize out the linear dynamics in the models to further accelerate it and this results in Rao-Blackwellized PF (RBPF) [10]. However, few PF meets the sparse tracking needs even under PC conditions.

Mean shift (MS) has been applied to image segmentation, visual tracking, nonparametric density analysis, etc. [11–14]. MS is a fixed point iteration procedure and is essentially a gradient ascent algorithm with an adaptive step size [15]. Fashing and Tomasi [16] indicate that MS is actually a quadratic bound optimization problem for both stationary and evolving sample sets. Since Comaniciu et al. [14] first introduce MS-based target tracking method, it has proven to be a promising alternative to popular UKF or PF trackers due to its low computational complexity, adaptive scale attribute, and immune to partial occlusions for various sequences. However, robustness for MS to copy with clutter scenes, to track objects of various sizes and to keep up with extremely fast moving objects still needs to be improved.

The basic MS tracker describes the targets with spatial masking color histograms or background weighted ones in RGB space [14]. In [17], features that perfectly discriminate object and background

* Corresponding author. Tel.: +86 10 13810771030.
E-mail addresses: shower140@163.com, shuxiao.li@ia.ac.cn (S.-X. Li).

0262-8856/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2009.06.012

are selected online to improve tracking effectiveness. Edge density and orientation features may also be integrated into the feature vector to make MS immune to illumination changes [18]. Recently, Jeyakar et al. [34] integrated color and edge histogram cues into the marginal mean shift framework. The over-all mean shift vector is obtained by weighting each of the partial mean shift vectors with the other Bhattacharyya coefficient. Furthermore, four local kernels are employed to copy with drift in tracking when the object undergoes partial occlusion. In [35], color likelihood ratio and boundary-related score are integrated to get better weight image for MS tracking. These tricks increase the computational complexity, approaching about 50 frames per second (fps) processing capability on common PC. However, for on-board surveillance applications, the algorithm is expected to run as fast as possible to reserve more process resources for high level operation. Different from previous works which describe the target model and candidate with the same expressions, we try to represent them by different ones. A complex and stable feature is selected for the model as it is computed only when the model is updated, while a feature with low computational complexity is adopted for the candidate. This strategy is able to achieve higher object localization precision than standard MS without increasing the computational complexity.

Scale selection is a difficult problem for algorithms like MS, which use a global color model of the target. A traditional way adapts window size by a fixed percentage and evaluates it using Bhattacharyya coefficient [14,20,35]. Although this does stop the window from growing too big, it is not sufficient to keep the window from shrinking too much. To overcome this problem, researchers tend to create a “likelihood” image (called weight image), with pixels weighted by similarity to the desired color (or other cues). Collins [13] proposes a method to generate a combined model which captures features at different scales. Then, MS is applied in the spatial and scale dimensions to the weight image, hence tracking blobs through scale space. Georgescu et al. [11] employ a pilot learning procedure to determine the optimal parameters of the clustering data structure. Jiang et al. [36] prove that the kernel bandwidth can be determined by maximizing the lower bound of log-likelihood function of Bhattacharyya coefficient, and the adaptive bandwidth is finally calculated by weighted standard deviation along the x- and y-directions, respectively. A similar approach is adopted in [34] with a different weight exploiting both foreground and background information. In this paper, we explore the response curve to scale changes to overcome the shrinking problem of the traditional way. Different thresholds for the scale incremental and reductive stages are applied. Together with the feature selection method, this scale selection method can track objects of varying sizes while the complexity remains unchanged.

Another important inherent drawback for MS is the local optimization. Standard MS tracker assumes that the initialization point falls within the basin of attraction of the desired mode, but this assumption may not hold when the displacement between successive frames is relatively large or when the camera vibrates. Shen et al. [19,20] proposed a global MS based tracker termed annealed MS as it shares similarities with annealed importance sampling procedure. This method gradually smoothes the cost function surface based on the idea of simulated annealing and annealed importance sampling and gently reaches the global peak. Recently, Jiang et al. [36] use multiscale images to localize target. For the low computational cost, they only use the images at level 0 and maximal level. We solve this problem by pyramid decomposition for efficient analysis. Moreover, a novel method is developed to determine the decomposition level adaptively to decrease the computational complexity, while in simulated MS or in [36] the level is fixed and is determined according to specific applications based on experiences [20]. The proposed method is termed as adaptive pyramid MS because it uses pyramid analysis with adaptive levels and scales. Comparing to simulated MS, adaptive pyramid MS is faster and more robust.

In summary, our key contributions comprise the following:

1. The effects of using different model and candidate expressions are proposed and analyzed, and the corresponding revised MS tracker is developed. This leads to high computational speed, adaptive object size, improved model stability and adaptive scale, which is described in detail in Section 2.3.
2. A novel adaptive pyramid MS tracker is developed which can efficiently and reliably cope with high-speed moving object, camera motion and vibration during tracking process as discussed in Section 3.
3. The adaptive pyramid MS and a mechanical stabilization board are integrated to implement a tracking and pointing subsystem where the camera moves fast and oscillates frequently, which is presented in Section 4.2.

The remaining contents include experimental results on several video clips in Section 4.1, and the conclusion of the paper is stated in Section 5 with a discussion of some significant issues.

2. MS tracker analysis

We first review the basic concepts of MS algorithm and MS tracker for completeness, and then show that the MS iteration procedure is independent of the target model expressions. This allows us to change model expressions properly, while the computational complexity remains unchanged if the same candidate expressions are employed. The background weighted histogram is selected for model and the color weighted histogram is adopted for candidate. Then the effects of this trick are analyzed. A corresponding revision to standard MS tracker about the adaptive scale decision rule is also presented below.

2.1. MS analysis

Let $k(x)$ be a profile (see [16] for definition), the density estimator can be expressed by [12]:

$$\widehat{f}_K(\mathbf{x}) = C \cdot \sum_{i=1}^{n} w_i k\left(\left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2\right), \tag{1}$$

where w_i is the weight for sample $\mathbf{x}_i$, h is the bandwidth, and C is a normalization constant. The optimization procedure of seeking the local modes is carried out by setting the gradient be zero:

$$\nabla f_K(\mathbf{x}) = C_1 \cdot \sum_{i=1}^{n} w_i g[r_i(\mathbf{x})] \cdot \mathbf{m}_G(\mathbf{x}) = 0, \tag{2}$$

where

$$\mathbf{m}_G(\mathbf{x}) = \frac{\sum_{i=1}^{n} w_i \mathbf{x}_i g[r_i(\mathbf{x})]}{\sum_{i=1}^{n} w_i g[r_i(\mathbf{x})]} - \mathbf{x}, \tag{3}$$

$$r_i(\mathbf{x}) = \left\|\frac{\mathbf{x}-\mathbf{x}_i}{h}\right\|^2, \tag{4}$$

and $g(x) = -k'(x)$. Here, $k(x)$ is called a shadow of $g(x)$ [16], C_1 is a constant and $\mathbf{m}_G(\mathbf{x})$ is the MS vector. Clearly, Eq. (2) holds if and only if:

$$\mathbf{x} = \frac{\sum_{i=1}^{n} w_i \mathbf{x}_i g[r_i(\mathbf{x})]}{\sum_{i=1}^{n} w_i g[r_i(\mathbf{x})]}. \tag{5}$$

The MS procedure can be considered as a fixed point problem as shown in Eq. (5) and various incremental iteration schemes can be employed to solve it. Direct iteration [14], Over-Relaxed iteration

[20] or Aitken iteration [21] can be employed to solve the problem at different convergence rates. In [16,20], the authors indicate that MS is a first order algorithm except for piece-wise profile for which the convergence rate is of order 2. A piece-wise profile called Epanechnikov profile [16] is adopted through this paper due to its low computational complexity and fast convergence rate:

$$k(x) = \begin{cases} \frac{1}{2} c_d^{-1}(d+2)(1-x), & x \leqslant 1, \\ 0, & \text{otherwise}. \end{cases} \tag{6}$$

Here, c_d and d are constants which actually do not affect the iterations. In this case, $g(x)$ is constant and Eq. (5) reduces to the following incremental iteration:

$$\mathbf{x} \leftarrow \frac{\sum_{i=1}^{n} w_i \mathbf{x}_i}{\sum_{i=1}^{n} w_i}. \tag{7}$$

2.2. MS tracker

Two popular features used for target model $\mathbf{q} = \{q_u\}_{u=1,\dots,m}$ and target candidate $\mathbf{p} = \{p_u\}_{u=1,\dots,m}$ in MS tracker are color weighted histogram ($\mathbf{p}^c, \mathbf{q}^c$) and background weighted histogram ($\mathbf{p}^b, \mathbf{q}^b$) [14]. Specifically, $\mathbf{p}^c$ or $\mathbf{q}^c$ can be computed as:

$$I_u^c(\mathbf{x}) = C^c \sum_{i=1}^{n_h} k[\mathbf{r}_i(\mathbf{x})]\delta[b(\mathbf{x}_i) - u], \quad I = \{p, q\}, \tag{8}$$

where $C^c = 1/\sum_{i=1}^{n_h} k[\mathbf{r}_i(\mathbf{x})]$ is the normalization constant, δ is the Kronecher delta function, $b(\mathbf{x}_i)$ denotes the serial number of histogram bin for $\mathbf{x}_i$. The RGB space is quantized into $16 \times 16 \times 16$ bins and each pixel contributes a vote for its bin number with a kernel weight. Similarly, the representation for $\mathbf{p}^b$ or $\mathbf{q}^b$ is:

$$I_u^b(\mathbf{x}) = C^b(\mathbf{x}) v_u \sum_{i=1}^{n_h} k[\mathbf{r}_i(\mathbf{x})]\delta[b(\mathbf{x}_i) - u], \quad I = \{p, q\}, \tag{9}$$

where $C^b(\mathbf{x}) = 1/\sum_{i=1}^{n_h}\sum_{u=1}^{m} k[\mathbf{r}_i(\mathbf{x})] v_u \delta[b(\mathbf{x}_i) - u]$ is the normalization constant, v_u is the background weight for voting and can be calculated by: (1) get the normalized histogram of the background $\{o_u\}_{u=1,\dots,m}$ in a region around the target (usually 3 times of the object scale); (2) denote o^* be the minimum value between ε (i.e. $\varepsilon = 0.1$) and the smallest nonzero entry of the normalized background histogram; (3) v_u takes the minimum value between o^*/o_u and 1.

Let $\mathbf{p}(\mathbf{x})$ represent the target candidate at location $\mathbf{x}$. Given an initial point $\mathbf{x}_0$, the problem of tracking is to estimate an optimal displacement $\Delta\mathbf{x}$ so that the similarity measurement between $\mathbf{q}$ and $\mathbf{p}(\mathbf{x})$ at the new location best matches the target model $\mathbf{q}$:

$$\Delta\mathbf{x}^* = \arg\max_{\Delta\mathbf{x}} \{\rho[\mathbf{q}, \mathbf{p}(\mathbf{x}_0 + \Delta\mathbf{x})]\}, \tag{10}$$

where $\rho(\mathbf{q}, \mathbf{p})$ is the similarity function between $\mathbf{q}$ and $\mathbf{p}$, and the Bhattacharyya coefficient is used in this paper as in [14]:

$$\rho(x) \equiv \rho[\mathbf{q}, \mathbf{p}(\mathbf{x})] = \sum_{u=1}^{m} \sqrt{q_u p_u(\mathbf{x})}. \tag{11}$$

Using Taylor expansions around the value $\mathbf{p}(\mathbf{x}_0)$, the maximization problem in Eq. (10) can be expressed as:

$$\mathbf{x}^* = \arg\max_{\mathbf{x}} \left\{ \sum_{u=1}^{m} p_u(\mathbf{x}) \sqrt{\frac{q_u}{p_u(\mathbf{x}_0)}} \right\}. \tag{12}$$

It can be seen from Eq. (12) that different representations of $\mathbf{q}$ impose different weights on histogram bins and thus affect the optimal value of it, while the iteration procedure as well as the computational cost remains unchanged. This allows us to use sophisticated techniques to construct accurate model since it is computed only when updated. Meanwhile, features that need low computational complexity are selected for candidates to save processing resources. Inserting Eqs. (8) and (9) into (12), the tracking problem can be viewed as a MS procedure as shown in Eq. (1) with different weights for different expressions of $\mathbf{p}$:

$$\omega_i^c = \sum_{u=1}^{m} \sqrt{\frac{q_u}{p_u^c(\mathbf{x}_0)}} \delta[b(\mathbf{x}_i) - u], \tag{13}$$

$$\omega_i^b = C^b(\mathbf{x}) \sum_{u=1}^{m} \sqrt{\frac{q_u}{p_u^b(\mathbf{x}_0)}} v_u \delta[b(\mathbf{x}_i) - u]. \tag{14}$$

2.3. Target model and target candidate analysis

Most available methods [13,14,20,34,36] use color weighted histogram for $\mathbf{p}$ and $\mathbf{q}$ instead of background weighted histogram, though the latter can better characterize the model itself. However, there is little work to reveal the reasons and to overcome it. Since the iteration procedure does not depend on the expression of $\mathbf{q}$, we try to represent the target model and target candidate with different expressions and exploit the benefits from this trick. There are four possible combinations between $\mathbf{p}$ and $\mathbf{q}$: (1) $\mathbf{p}^c$ and $\mathbf{q}^c$; (2) $\mathbf{p}^c$ and $\mathbf{q}^b$; (3) $\mathbf{p}^b$ and $\mathbf{q}^c$; and (4) $\mathbf{p}^b$ and $\mathbf{q}^b$. We now discuss the model stabilization, adaptive object size, adaptive object scale and computational complexity of the above four combinations with a simple real-world example as shown in Fig. 1. More complex objects and cluttered backgrounds also produce similar results as shown in Section 4. The image size is 240×240, the location of the interested target (x_0, y_0) is $(122, 122)$, and the value of actual radius R_0 is about 35.

2.3.1. Model stability

Let $\mathbf{q}_0$ be the true target model, which can be approximately computed by using color weighted histogram at (x_0, y_0) with radius R_0. We use the Bhattacharyya coefficient $\rho(\mathbf{q}, \mathbf{q}_0)$ as the similarity measure between the acquired target model $\mathbf{q}$ and the ground truth model $\mathbf{q}_0$. Let ρ_c represents the similarity between $\mathbf{q}^c$ and $\mathbf{q}_0$, and ρ_b represents the similarity between $\mathbf{q}^b$ and $\mathbf{q}_0$. The influences of different initial object scales (denoted as $R_0 + r$, as shown in Fig. 1) are examined as shown in Fig. 2(a). It can be

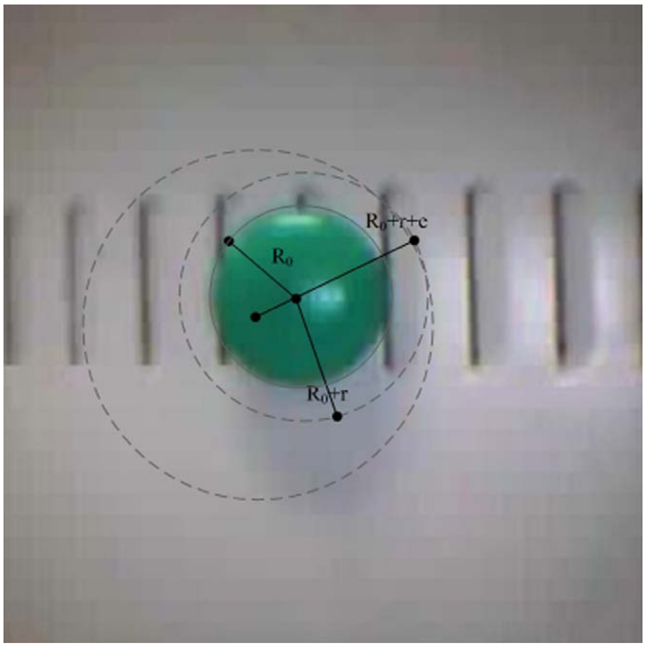


Fig. 1. An illustration to target model and target candidate selection effects.

seen that target model $\mathbf{q}^b$ has a good approximation to $\mathbf{q}_0$ when the value of r is larger than $0.2R_0$ because the background regions inside the initial scale are suppressed. This is very suitable for practical applications because a larger scale is always selected to get comprehensive descriptions of the target. A suitable value of r for background weighted histogram lies between $0.2R_0$ and $0.4R_0$, which is easy to be generated by human or by detection process. On the other hand, $\mathbf{q}^c$ decreases quickly with respect to r and has good approximations when the initial scale is relatively small, which is a rather strict constraint sometimes.

The affects of different initial location errors (denoted as e, as shown in Fig. 1) are examined as shown in Fig. 2(b). Based on the above scale analysis, the initial object scale changes with e according to the rule $R = R_0 + r + e$ to include the whole object inside the circle. The curves of $\mathbf{q}^b$ and $\mathbf{q}^c$ with respect to e when r equals to $0.2R_0$ or $0.4R_0$ are shown. It can be indicated that the estimation precision of $\mathbf{q}^b$ due to locating errors in this common-use initialization manner is always higher than that of $\mathbf{q}^c$. In addition,$\mathbf{q}^b$ is robust against scale changes while $\mathbf{q}^c$ decrease with respect to r. As a whole, the stability of $\mathbf{q}^b$ is quite high, while that of $\mathbf{q}^c$ is rather normal.

2.3.2. Adaptive object size

Denote the similarities between the above four possible combinations as ρ_1, ρ_2, ρ_3, and ρ_4, respectively. At this point, the target model is fixed with a certain initial scale value and the scale values of candidates are modified. Then, the similarity measures between the model and candidate when the initial scale takes the value of $1.2R_0$ or $1.4R_0$ are calculated, as shown in Fig. 3(a) and (b), respectively. It can be seen that no matter how much the value of object scale is initialized, the tracker of combination (2) (ρ_2) will reach the maximum of the curve where the ground truth scale lies, while the tracker of combination (1) (ρ_1) tends to converge to the initial scale and the other two trackers (ρ_3 and ρ_4) have inflexions nearby the initial scale. This indicates that the tracker of combinations (1), (3) and (4) will converge to the initial scale with different precisions and are blind to the actual object scale. By exploiting the benefits from representing the target model and candidate with different expressions, the tracker of combination (2) have a trend to converge to ground true object scale, and we term this ability as adaptive object size ability.

2.3.3. Adaptive scale

Adaptive scale refers to the ability that the scale of target candidate can change along with the variation of image resolution. This can be reflected by the responses to scale changes as shown in Fig. 3 too. It can be observed that the curve of combination (1) or combination (2) has a single peak which indicates good adaptive scale ability, while the abilities of combination (3) and (4) are rather poor. For combinations (1) and (2) which have single peaks, the scale which gets the highest similarity measure can be adopted to update the object scale as is used in [14]. However, this may fail when the object model drifts due to illumination and view angle changes, or when the user initializes a tight bounding box. Fig. 4 shows the similarity measure to scale changes with several initial scale values of target model. When the initial scale becomes smal-

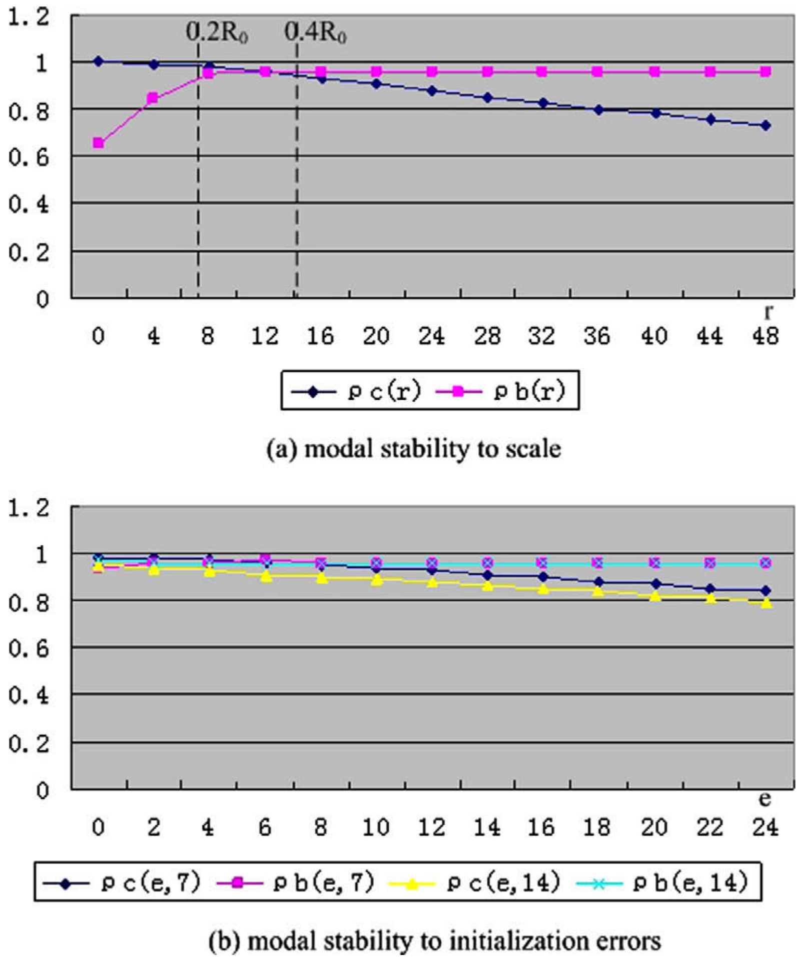
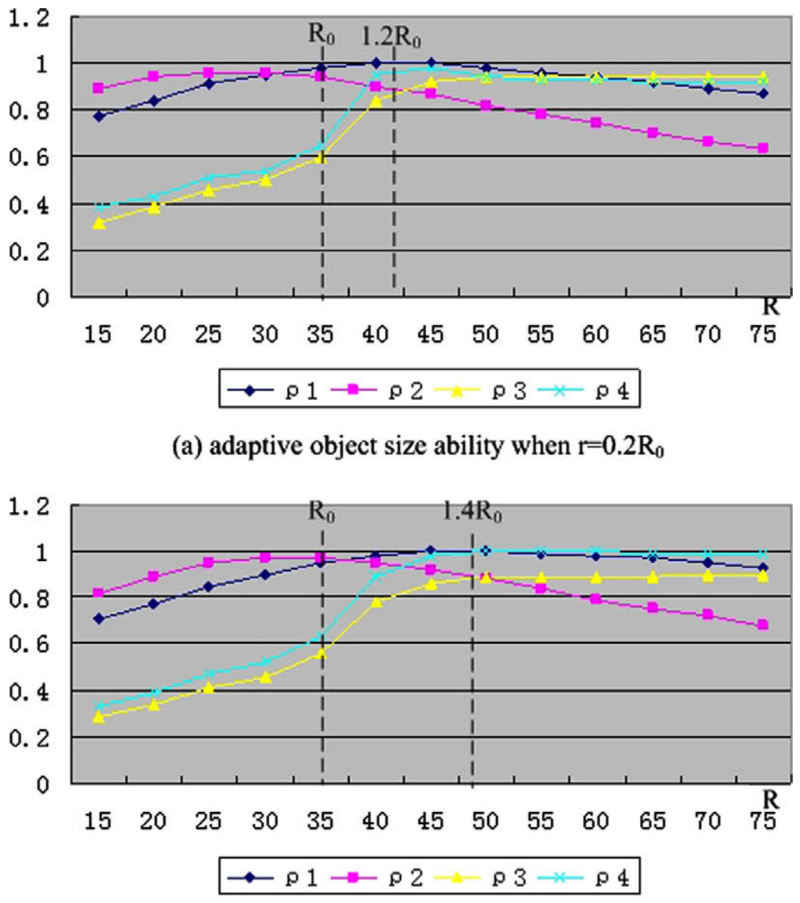


Fig. 2. Modal stability analysis of the four combinations.



(a) adaptive object size ability when r=0.2R₀

(b) adaptive object size ability when r=0.4R₀

Fig. 3. Adaptive object size ability analysis of the four combinations.

ler, the peak gradually gets weak and eventually disappears. By experiences, the candidate scale has more likely to shrink than to enlarge because a smaller area contains less noises. So a corresponding adaptive scale rule is proposed and presented below.

Let h_{prev} denote the bandwidth in the previous frame. We measure the bandwidth h in the current frame by running the target tracking algorithm three times, with bandwidths $h_l = (1+a)h_{\text{prev}}$, $h_2 = h_{\text{prev}}$, and $h_3 = (1-a)h_{\text{prev}}$ where the default value for a is 0.05. Let the similarity measure values of these trackers be ρ_1, ρ_2 and ρ_3, respectively. The decision rule can be expressed as below.

$$h = \begin{cases} h_1, & \Delta\rho_1 \geqslant 0, \quad \Delta\rho_1 \geqslant \Delta\rho_3, \\ h_3, & \Delta\rho_3 \geqslant 0, \quad \Delta\rho_3 \geqslant \Delta\rho_1, \\ h_2, & \text{others}, \end{cases} \tag{15}$$

where

$$\Delta\rho_1 = \rho_1 - \rho_2 - e_1; \quad \Delta\rho_3 = \rho_3 - \rho_2 - e_2, \tag{16}$$

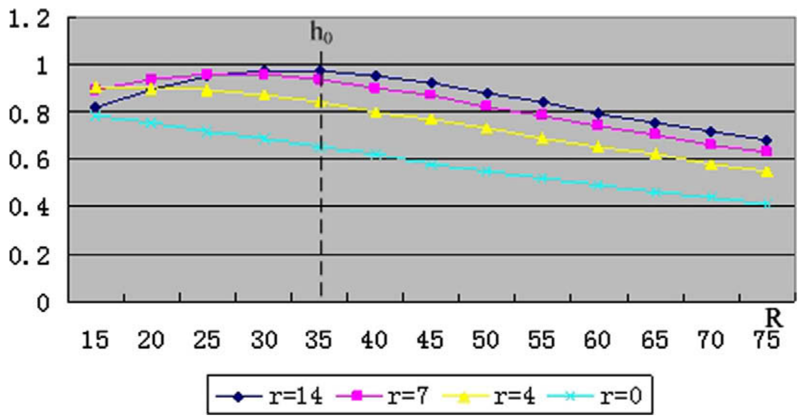


Fig. 4. Illustration of adaptive scale rule selection for combination (2).

and e_1, e_2 are constants. The proper values for them are −0.005 and 0.05 in our experiments. Different thresholds for the scale incremental and reductive stages are applied because the response curve to scale changes has distinct characteristics between the shrinking and the enlarging processes.

2.3.4. Computational complexity

Now that the model is computed merely when updated and its expression does not affect the MS iteration, the computational complexity difference mainly lies in the calculation of $\mathbf{p}^c$ and $\mathbf{p}^b$. From Eqs. (8) and (9), the cost of the two candidates is approximately determined by:

$$C^c = n_h(c_k + c_m + c_a) + n_f c_m, \tag{17}$$

$$C^b = n_h(c_k + 2c_m + c_a) + 8n_h c_a + 3n_f c_m, \tag{18}$$

where n_h is the number of sample set, n_f is the number of histogram bins, c_k, c_m, and c_a are the cost of profile calculation, a multiplication and an addition, respectively. C^c and C^b are the approximate costs for calculating $\mathbf{p}^c$ and $\mathbf{p}^b$. The middle term in Eq. (18) indicates background histogram cost and the last term represents the cost of histogram normalization. From Eq. (6), we know that c_k is approximately $3c_m + 4c_a$, and this implies

$$\frac{C^c}{C^b} = \frac{n_h(4c_m + 5c_a) + n_f c_m}{n_h(5c_m + 13c_a) + 3n_f c_m}. \tag{19}$$

In DSP applications where the cost of c_m and c_a are the same, the ratio of the cost of $\mathbf{p}^b$ to that of $\mathbf{p}^c$ is reduced to

$$\frac{C^b}{C^c} = \frac{18n_h + 3n_f}{9n_h + n_f}, \tag{20}$$

which indicates that the cost of $\mathbf{p}^b$ is approximately 2 times to that of $\mathbf{p}^c$.

Table 1 summarizes the above four properties of the possible combinations of $\mathbf{p}$ and $\mathbf{q}$. As can be seen, the overall performances of combination (2) which we use in this paper are superior to that of the other three combinations. This conclusion coincides with the experience that more sophisticated methods for target model (e.g. $\mathbf{q}^b$) can get stable object features while simple ones for target candidate (such as $\mathbf{p}^c$) can run efficiently at the same time. In addition, since the model has little disturbances, the target candidate has the trend to discard redundant regions and thus has the adaptive object size ability. This property relaxes the precision of the initial bounding box, resulting in more convenient and efficient object tracking.

3. Adaptive pyramid MS

Let $\mathbf{I}_m$ $(m = 0, 1, \ldots, M)$ be the pyramid image sequence with sampling frequency factor a and maximum level M. The default value of a is 2, and M depends on the maximum displacement between two successive frames in domain fields. Then profile $k(x)$ with bandwidth h as in Eq. (6) is applied to the image sequence to get similarity image sequence $f_m(\mathbf{x})$ $(m = 0, 1, \ldots, M)$ using Eq. (11), and the tracking process can be expressed to be the maximum value seeking problem of $f_0(\mathbf{x})$ given initial point $\mathbf{x}_0$. In aerial video surveillance applications, $f_0(\mathbf{x})$ usually has many local modes and the gradient based seeking algorithm (e.g. MS) will be trapped into a local mode when $\mathbf{x}_0$ does not fall within the basin of attraction of the desired mode.

In density estimation and target tracking fields, it has been proved that with a sufficiently large bandwidth h for profile $k(x)$, the similarity function in this case is strictly uni-model [20,22]. Another fact is that increasing the bandwidth h for $k(x)$ is equivalent to decreasing the resolution of the original image called pyramid analysis. This implies that $f_M(\mathbf{x})$ will be uni-model with a large M value, and the start point will not affect the mode detection of $f_M(\mathbf{x})$. We first evaluate the local mode $\mathbf{x}^k$ in a higher level image, and get $\mathbf{x}^{k-1}$ initialized from $\mathbf{x}^k$ in the lower level image. Then $\mathbf{x}^0$ is the final global mode. This coarse-to-fine strategy results in global visual trackers, which are appropriate for tracking objects under different conditions such as high-speed moving targets, camera vibrations, etc.

Fig. 5 illustrates an example of the proposed global tracking algorithm. The white circle in the image center is the object of interest, while the initial point A is rather far away. The MS tracker will be trapped into the small white circle if it is directly used. In our algorithm instead, the object is first tracked in the highest level, and then the MS tracker is applied again for the lower level with the tracking result in the higher level as the initial point. The procedure iterates until we get the mode of level 0, as shown in the bottom row of Fig. 5. The pyramid MS tracker locates the object correctly and achieves the global tracking performances.

Another global MS tracker termed annealed MS is proposed in [20] recently. Annealed MS has similar performances to ours when their bandwidths are selected with a scale factor a. But we achieve the goal in a different way by pyramid analysis, while theirs are stimulated by annealed sampling. Further, the proposed method operates on fixed bandwidth rather than enlarged ones as in An-

Table 1
Performance comparison for the four possible combinations.

Combination index	Model stability	Adaptive object size	Adaptive scale	Speed
1	Normal	Bad	Good	Fast
2	Good	Good	Good	Fast
3	Normal	Bad	Bad	Normal
4	Good	Bad	Bad	Normal

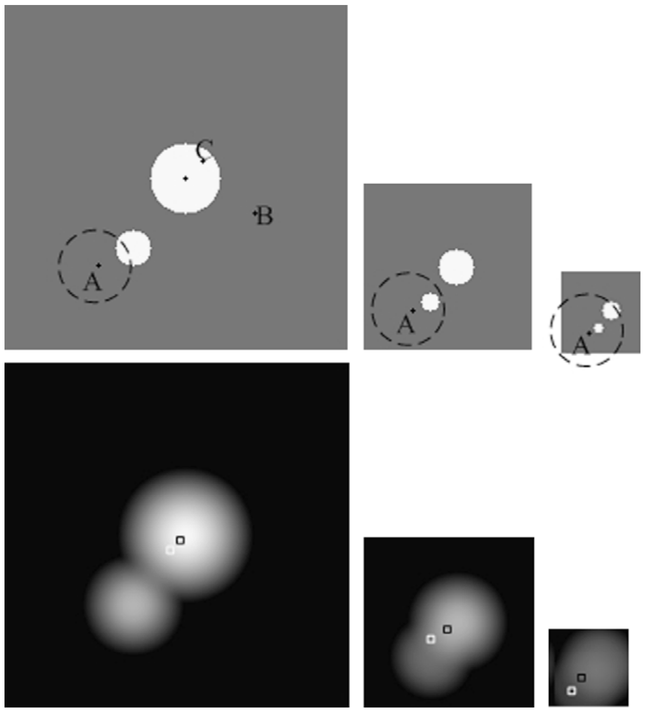


Fig. 5. Pyramid MS tracking procedure example. (Top) Pyramid image sequence $\{\mathbf{I}_0, \mathbf{I}_1, \mathbf{I}_2\}$, the white circle in the image center is the object of interest, and points A, B, and C are possible tracking results in the previous frame. (Bottom) Similarity function image sequence $\{f_0(\mathbf{x}), f_1(\mathbf{x}), f_2(\mathbf{x})\}$, the white box and the black small box indicate the initial and the mode point, respectively.

nealed MS, resulting in relatively lower computational complexity. In addition, we propose a method for determining the pyramid decomposition level on-line first time in the following to decrease the iteration times, while annealed MS operates on fixed levels. We discuss adaptive level method bellow, followed by a summary of the adaptive pyramid MS algorithm.

3.1. Adaptive level

Let $\mathbf{q}$ represents the object model $\mathbf{q}^b$ for short, and $\mathbf{p}^l$ represents the histogram within bandwidth h for image level l by:

$$\mathbf{p}_u^{l,h}(\mathbf{x}) = \sum_{i=1}^{n_h} \delta[b_l(\mathbf{x}_i) - u], \tag{21}$$

where $b_l(\mathbf{x}_i)$ denotes the serial number of histogram bin for $\mathbf{x}_i$ in image level l, and n_h is the number of pixels within band h. The histogram is not weighted because the object could be in any direction and possibly be distant from the center. If the image consists of piece wise constant objects and backgrounds before sampling, after simple manipulations we can get:

$$\mathbf{p}_u^{l,h}(\mathbf{x}_l) = a^2 \mathbf{p}_u^{l+1,\frac{h}{2}}(\mathbf{x}_{l+1}) \leqslant a^2 \mathbf{p}_u^{l+1,h}(\mathbf{x}_{l+1}). \tag{22}$$

The above relationship usually holds even if the image contains complicated scenes. Thus we get a monotonously incremental function which is appropriate for determining the adaptive level for pyramid MS:

$$c(l) = a^{2(l-1)} \frac{\langle \mathbf{p}^l, \mathbf{q} \rangle}{n_h \langle \mathbf{q}, \mathbf{q} \rangle}, \tag{23}$$

where $\langle \mathbf{p}, \mathbf{q} \rangle$ denotes the inner product between $\mathbf{p}$ and $\mathbf{q}$. The first term is used to compensate level changes and the denominator of the second term is normalized constants for model characteristics and object size. The monotonously incremental property of $c(l)$ can be derived from the linear property of inner product and Eq. (22) directly. Another property is that when the object lies within the bandwidth, $c(l)$ is approximate to or even larger than 1. A rigorous method for determining the pyramid level can be achieved by analyzing the value difference of Eq. (23) between level l and level $l+2$, because the object definitely enters the bandwidth entirely within two levels. However, due to the monotonously incremental and normalized properties, a hard threshold is usually enough to determine the adaptive level for pyramid MS. Our experimental results indicate that $c(l)$ which is larger than 0.6 is enough in most cases, and for more robust target tracking, we determine the adaptive level by:

$$L = \min_l \{c(l) > 0.8 : l \leqslant M\}. \tag{24}$$

Fig. 6 shows the measures under different levels for points A, B, and C in the top-left image as shown in Fig. 5. The algorithm selects pyramid levels adaptively, and we term this ability as adaptive level. Though the measure is not unique, this measure does work well in our experiments as shown in Section 4.

3.2. Adaptive pyramid MS algorithm

The adaptive pyramid MS algorithm is summarized in Fig. 7.

4. Experimental results and applications

Concerning the function of adaptive pyramid MS algorithm, the method is applied to several video clips and a tracking and pointing subsystem. The images are captured by a Sony EVI-D70P P/T/Z camera with image card MV-U2000. The image resolution is

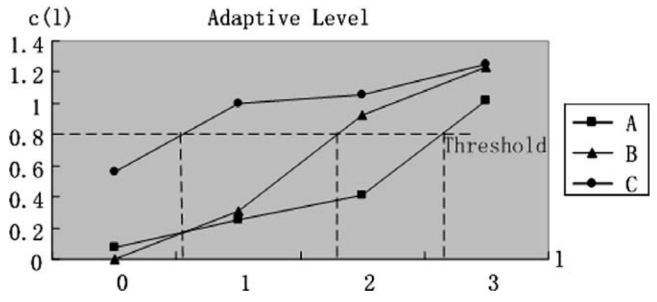
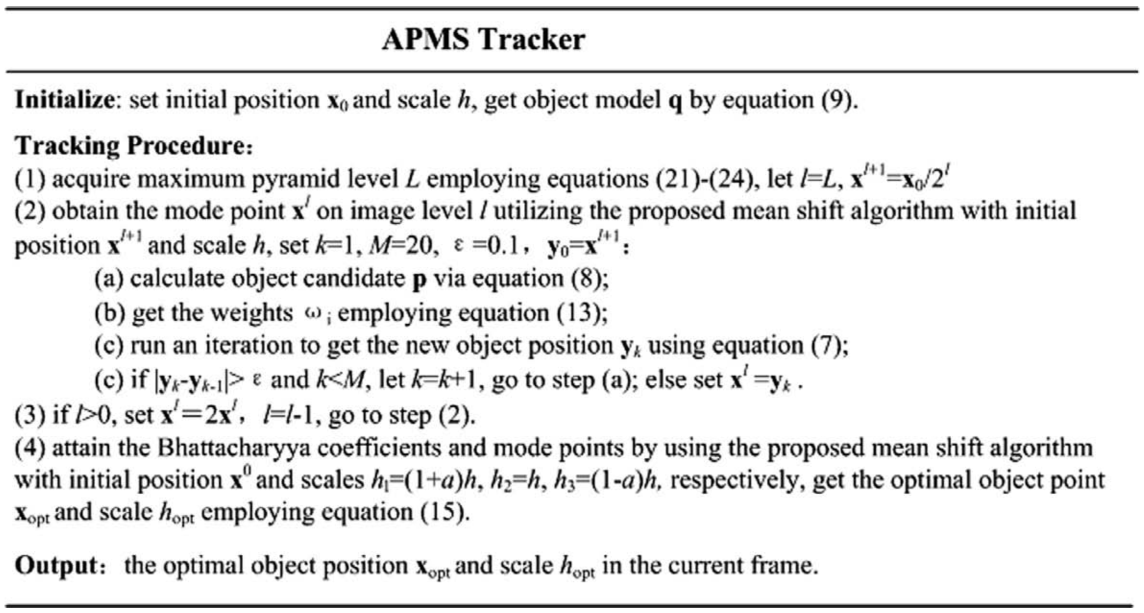


Fig. 6. Level measures for adaptive level decision of points A, B, and C in the top-left image in Fig. 5.



APMS Tracker

Initialize: set initial position $\mathbf{x}_0$ and scale h, get object model $\mathbf{q}$ by equation (9).

Tracking Procedure:

```
(1) acquire maximum pyramid level $L$ employing equations (21)-(24), let $l=L$, $\mathbf{x}^{l+1}=\mathbf{x}_0/2^l$
(2) obtain the mode point $\mathbf{x}^l$ on image level $l$ utilizing the proposed mean shift algorithm with initial
position $\mathbf{x}^{l+1}$ and scale $h$, set $k=1$, $M=20$, $\varepsilon=0.1$, $\mathbf{y}_0=\mathbf{x}^{l+1}$:
        (a) calculate object candidate $\mathbf{p}$ via equation (8);
        (b) get the weights $\omega_i$ employing equation (13);
        (c) run an iteration to get the new object position $\mathbf{y}_k$ using equation (7);
        (c) if $|\mathbf{y}_k-\mathbf{y}_{k-1}|>\varepsilon$ and $k<M$, let $k=k+1$, go to step (a); else set $\mathbf{x}^l=\mathbf{y}_k$ .
(3) if $l>0$, set $\mathbf{x}^l=2\mathbf{x}^l$, $l=l-1$, go to step (2).
(4) attain the Bhattacharyya coefficients and mode points by using the proposed mean shift algorithm
with initial position $\mathbf{x}^0$ and scales $h_1=(1+a)h$, $h_2=h$, $h_3=(1-a)h$, respectively, get the optimal object point
$\mathbf{x}_{opt}$ and scale $h_{opt}$ employing equation (15).
```

Output: the optimal object position $\mathbf{x}_{opt}$ and scale h_{opt} in the current frame.

Fig. 7. Summary of the proposed tracking algorithm.

640 × 480 of RGB format. We first discuss and analyze the experimental results on several video clips, and then apply the algorithm to tracking and pointing (T&P) subsystem for hand-held or UAV-carried cameras.

4.1. Experimental results on video clips

We apply adaptive pyramid MS to several video clips and present some representative results in this section. The experimental

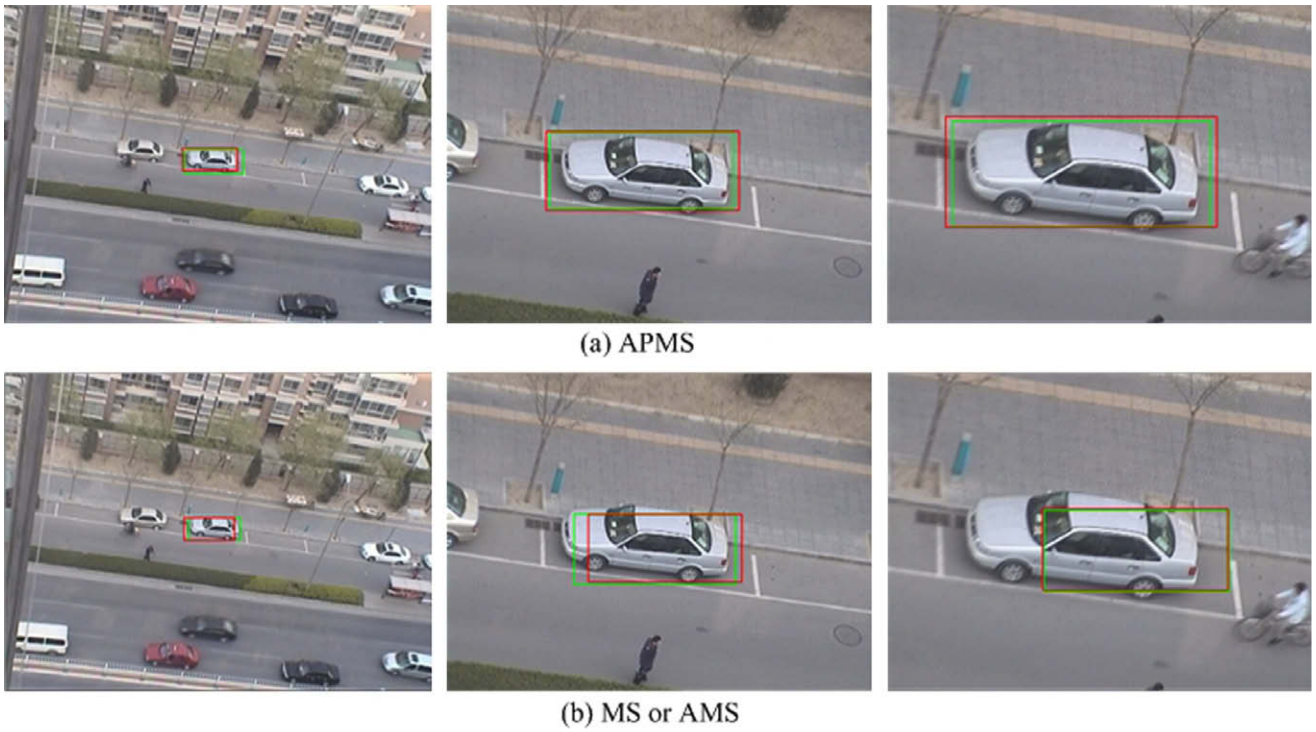


Fig. 8. Focus and zoom sequence. The frames 2, 52, and 62 are shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

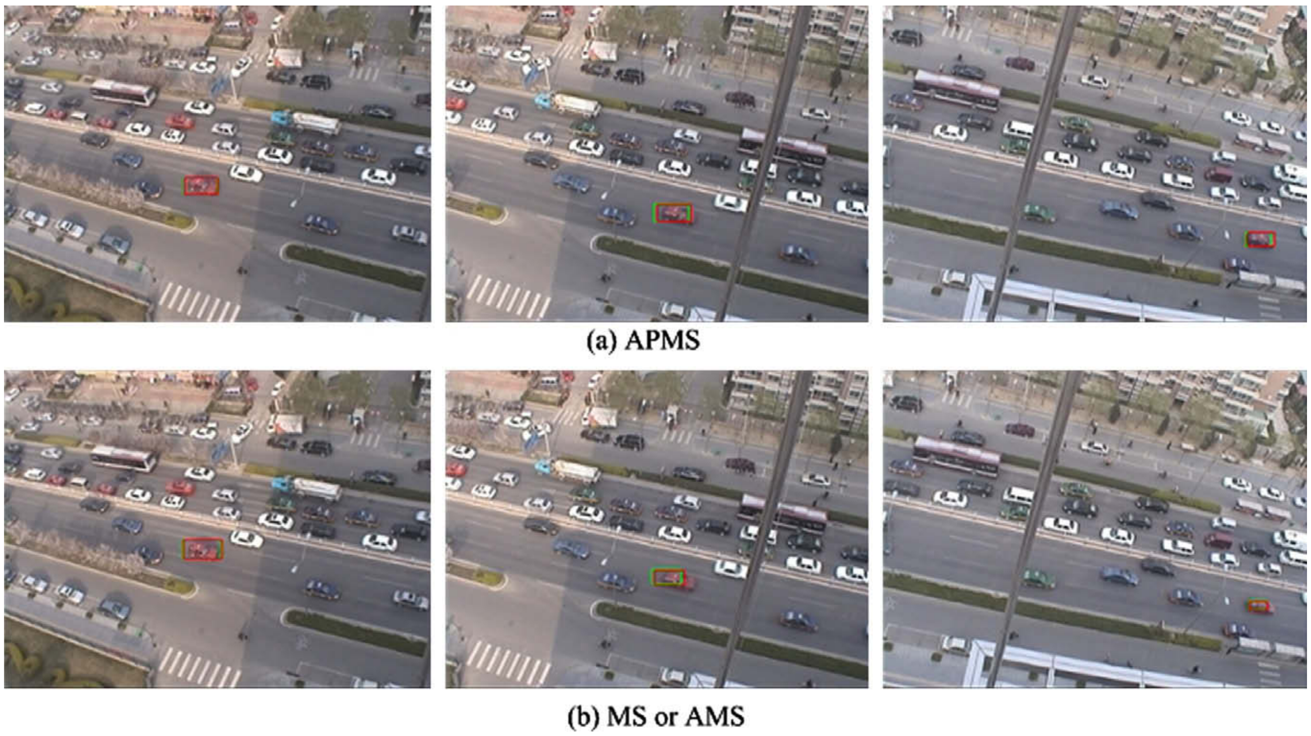


Fig. 9. Results to illustrate tracking accuracy for MS, AMS and APMS. The frames 12, 32, and 72 are shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

results on adaptive scale ability, target tracking accuracy, adaptive object size ability, global tracking ability and computational speed of the proposed method compared with classical MS and annealed MS are discussed below.

4.1.1. Adaptive scale ability

In this case, the maximum pyramid level for adaptive pyramid MS (APMS) and the simulated level for annealed MS (AMS) are set to be 1. APMS then degenerates to MS with the only difference on the model and candidate selection strategy as discussed in Section 2.3 and AMS becomes the same as MS. Fig. 8 shows an example of tracking a car. The camera focuses on a white car first, and then zooms out. APMS locks the car successfully while traditional MS or AMS fails after the frame number 62. The reason is that APMS employs an improved adaptive scale selection strategy. Additional experiments on several videos exhibit similar results. It can be concluded that, the adaptive scale ability of APMS is better than that of MS or AMS.

4.1.2. Target tracking accuracy

Another sequence is shown in Fig. 9 to illustrate the tracking accuracy for MS, AMS and APMS. The initial bounding box is carefully toned to include most foreground pixels and exclude clutters in background at the same time for MS and AMS. However, MS or AMS locks the interested object with a relatively large location error (frame 32) or scale error (frame 72). On the contrary, APMS can get the accurate object position and scale throughout the whole sequence and other tested sequences. This is because that APMS has high model stability and employs an improved adaptive scale selection rule, resulting in an overall better tracking accuracy of APMS than that of MS or AMS.

4.1.3. Adaptive object size ability

A good tracker must provide a convenient way to be initialized, either by hand or by a detection process. For this purpose, a tracker is desired to be immune to initial errors in both location and scale spaces and is expected to converge to the true state at the same time. Fig. 10 shows a tracking example when the initial bounding box is obviously larger than the object scale. The bounding box of MS or AMS remains almost unchanged and the tracking result is sensitive to surroundings as shown in frame 22, while that of APMS shrinks fast until it becomes close to the object boundary due to its adaptive object size ability as described in Section 2.3. As a result, the performances of MS or AMS are highly depended on the initial bounding box while APMS is relatively robust against it and easy to use in practices.

4.1.4. Global tracking ability

At this point, we set the maximum level to be 3 for APMS and annealed MS (AMS) to copy with different situations like high-speed moving objects, serious occlusions, camera vibrations, etc. Fig. 11 shows a barrow tracking example. It is indicated that MS loses the object after frame 274 when serious occlusions occur, while AMS and APMS track it successfully. The difference between AMS and APMS lies in the model stability and tracking accuracy as described above. Another difference is the computational speed which will be discussed in the following item.

Fig. 12 displays the tracking results of “egtest04” sequence. There are occlusions (frame 80), camera vibrations (frame 188) and illumination changes (frame 200) in this clip and these make the task very difficult. It can be seen that MS losses the target on frame 188 when the movement between successive frames is large due to camera vibrations while APMS can cope with these difficulties.

Fig. 13 shows another two tracking examples for APMS. In “eyeball” sequence, the eyeball moves very fast and may disappear when the boy blinks his eyes. In the “plane2” clip, the pose of the target plane changes and there exists clutters due to mirages. As is revealed by the results, APMS is able to track these targets successfully with high tracking accuracy.

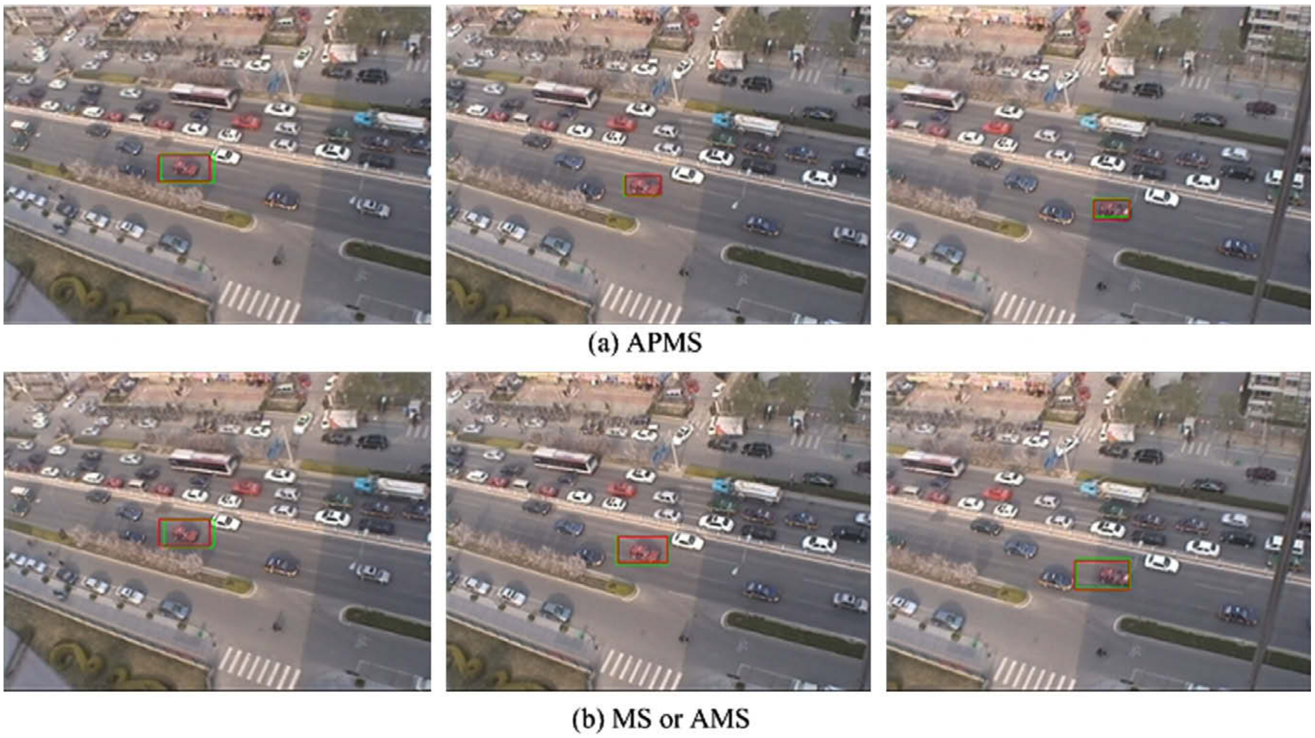


Fig. 10. Results to illustrate adaptive object size ability. The frames 2, 12, and 22 are shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

Fig. 14 shows the tracking results of "egtest02" sequence. There are two other cars that have similar color in this example. The tracker locked the interested car correctly when it encounters the first similar car (frame 547), but fails when meeting the second car which has almost the same color (frame 668). This is because that APMS (or MS, AMS) tends to select the target which has the most similar color to the model. So, when the neighboring vehicles are of similar color, it will select the more likely one and may track the other object. The data association method or a more complex image model can be incorporated into the algorithm to address these situations in future work.

4.1.5. Computational speed

Since MS fails frequently in our experimental circumstances, we merely compare the computational speeds between AMS and APMS. It is noticed that the computational speeds of the two methods only depend on the iteration times because they employ the same candidate features and MS procedure. So only the average iteration times for tracking a specific target in each video sequence are compared. The maximum level for APMS is set to be 3, while that for AMS is set to be 3 or 2 according to specific tracking video clips. This indicates that the level parameter for AMS needs to be carefully toned, while APMS is easier to use in practices.

Fig. 15 shows the average iteration times for 12 testing sequences. The iteration times for sequence number 6 between AMS and APMS is close to each other because the target appearances change at the very beginning due to illumination changes, resulting in a low value of $c(l)$ in Eq. (23). This can be improved by selecting appropriate model updating strategy. The average iteration times to achieve convergence are approximately 11 for APMS, and are about 23 for AMS. This indicates that the computational speed of APMS is about 2 times to that of AMS due to using an adaptive level decision strategy, which greatly improves the real-time tracking performances.

We tested APMS on a 2.4 GHz PC with 512 MB memory. The algorithm is written in C++ code in the frame of Visual C++ 6.0. In these conditions, the cost for tracking a 50 by 50 object in a single iteration is about 0.5 ms, which indicates that a rate of 150 fps can be achieved for APMS.

Table 2 summarizes the above results for MS, AMS and APMS. As can be seen, APMS has better or similar performances compared to MS or AMS, resulting in more efficient and effective object tracking.

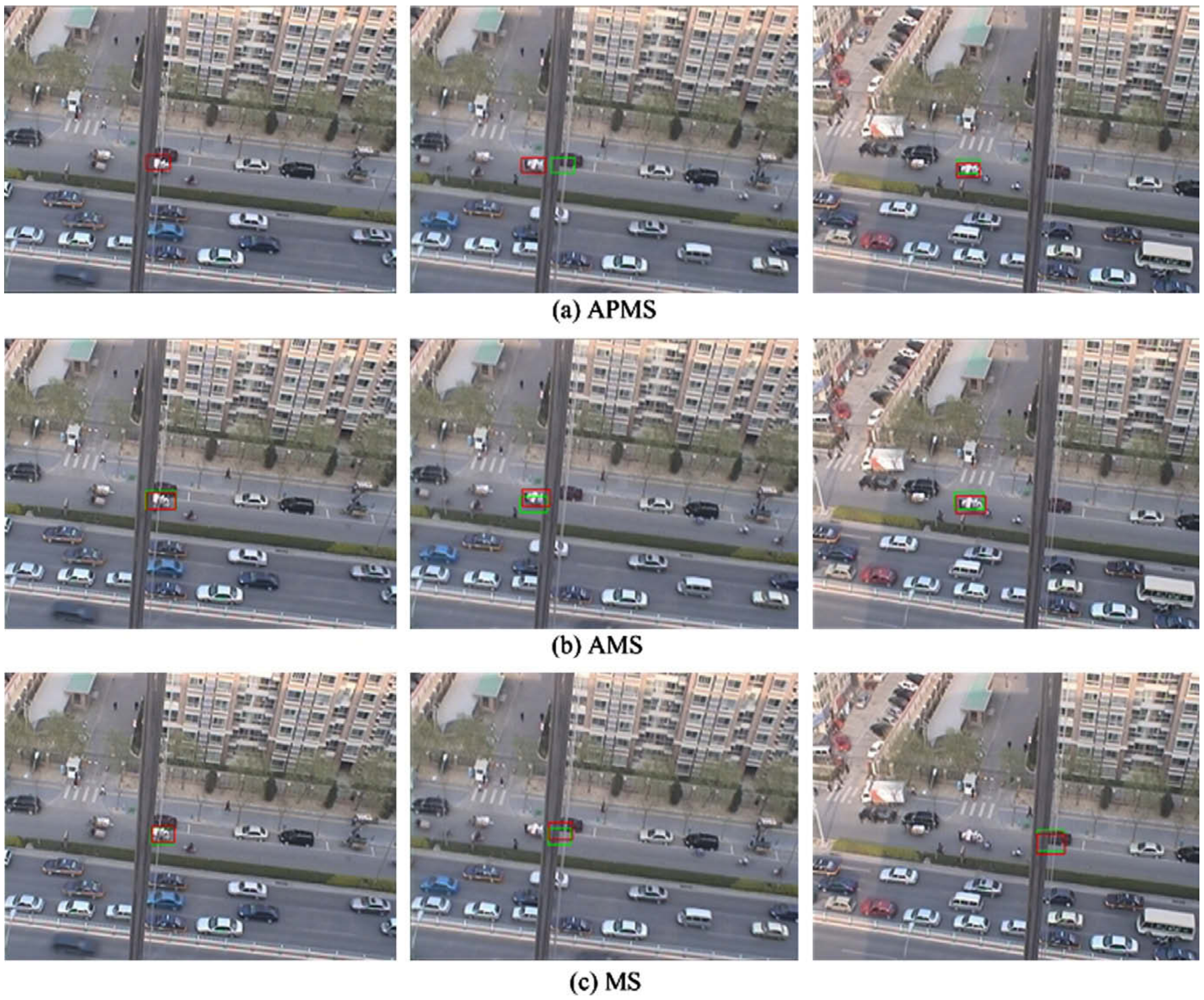


Fig. 11. Results to illustrate global tracking ability. The frames 245, 274, and 332 are shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

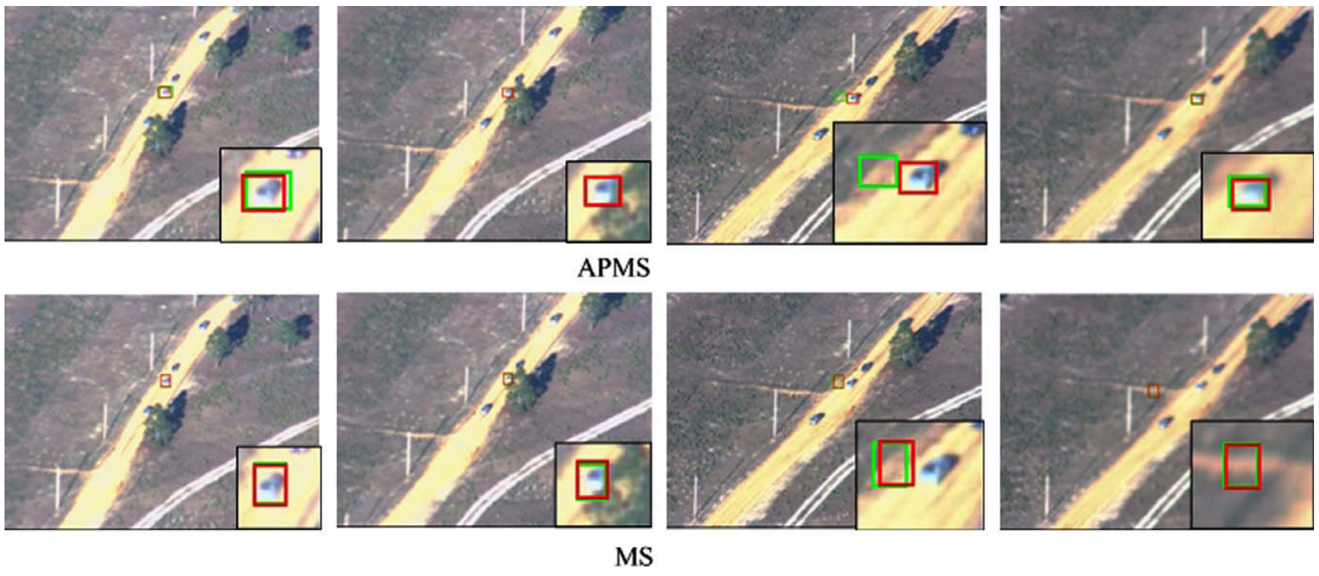


Fig. 12. Results of "egtest04" sequence for MS and APMS. The frames 2, 80, 188, and 200 are shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

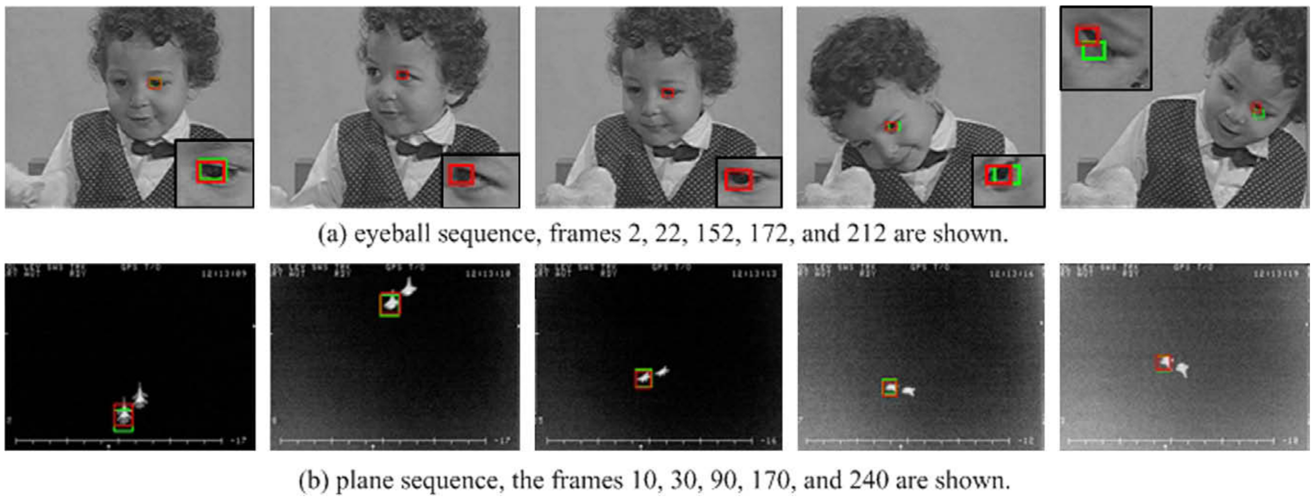

(a) eyeball sequence, frames 2, 22, 152, 172, and 212 are shown.
(b) plane sequence, the frames 10, 30, 90, 170, and 240 are shown.

Fig. 13. Additional results are shown to illustrate the tracking performances for APMS. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

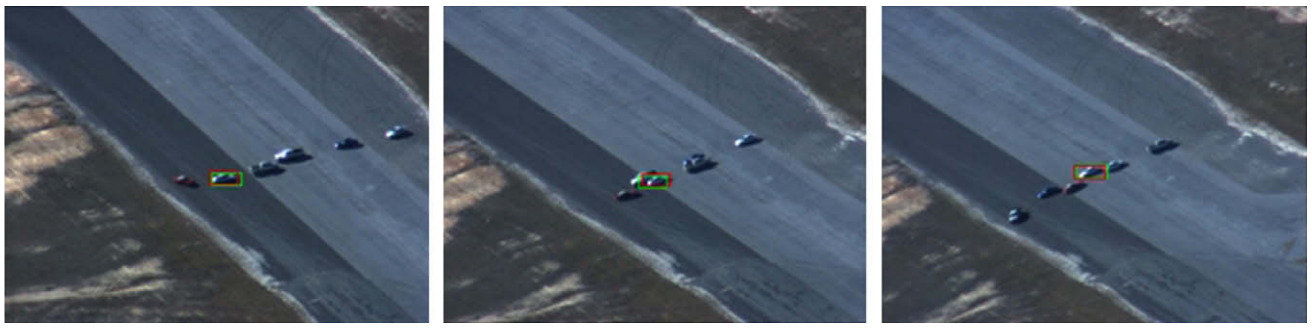


Fig. 14. Results of "egtest02" sequence for APMS. The frames 461, 547, and 668 are shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame. We can see that the tracker may fail when the surrounding object has similar color.

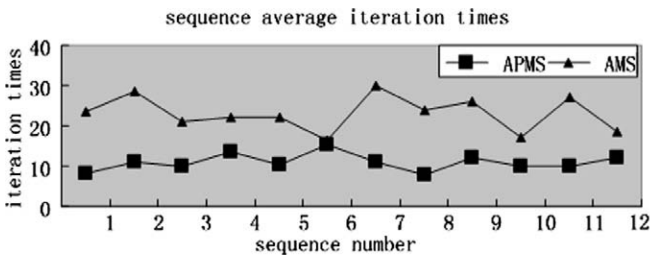


Fig. 15. Average iteration times for AMS and APMS.

Table 2
Performance comparison for MS, AMS and APMS.

	Adaptive scale ability	Target tracking accuracy	Adaptive object size ability	Global tracking ability	Computational speed
MS	Normal	Normal	Bad	Bad	Fast
AMS	Normal	Normal	Bad	Good	Slow
APMS	Good	Good	Good	Good	Fast

4.2. Application to T&P subsystem

In this section, we examine T&P subsystem for an integrated aerial video surveillance (AVS) system. The camera vibrates frequently due to engine quivers and poses may change abruptly because of external forces such as sudden gales. Thus the object may be out of view and the tracking process could be interrupted. When UAV flies at high altitudes, merely a small angle change caused by vibrations can introduce large movements of view on the ground. For example, if the UAV flies at a height of 2000 m and the camera rotates 1° because of vibrations, this brings in a displacement of about 34.9 m for the view of the ground. So a fast global tracking algorithm as well as camera stabilization mechanics are needed to get robust tracking and pointing tasks for AVS system.

Fig. 16 illustrates the T&P subsystem which includes a tracking component and a camera stabilization component. The camera stabilization mechanics employ three gyros for the *X*-, *Y*- and *Z*-axis, respectively, to obtain the angle velocities at any instant. Then a method called quaternion is employed to get the angle changes at a short time interval [23–25]. An Atmel AVR MCU is adopted to compute the angle changes which are then transferred to the camera control center to adjust the poses of the camera for *X*- and *Y*-axis with two LCG50 gyros. This component is implemented and integrated into a custom-built circuit as shown in Fig. 17.

As for the tracking component, a fast global tracking algorithm is needed. Fast trackers are required because the computational ability on board is limited and parts of the resources are expected to be left for high level use. The target displacements between two successive frames may be relatively large during the camera stabilization process or due to high-speed movements, so only global trackers are competent for this application. APMS satisfies all these needs and so exhibits good performances in the T&P subsystem.

A difference between video tracking and T&P subsystem lies in the initial manners for the trackers. The difficult problem of object detection and localization for UAV is under considered, so fully automatic initialization is still not available. For video tracking, we have enough time to initialize the object before tracking, for example drawing a bounding box. However, it is quite difficult for T&P subsystem to initialize in this manner because the on-line tracked object moves very fast and little time is left. Instead, a point generation algorithm is employed to get the bounding box

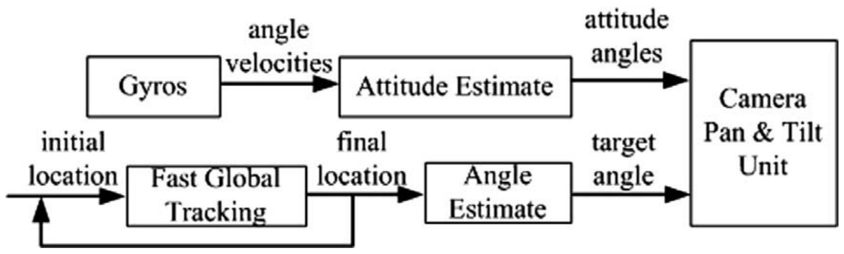


Fig. 16. Components of a T&P subsystem.

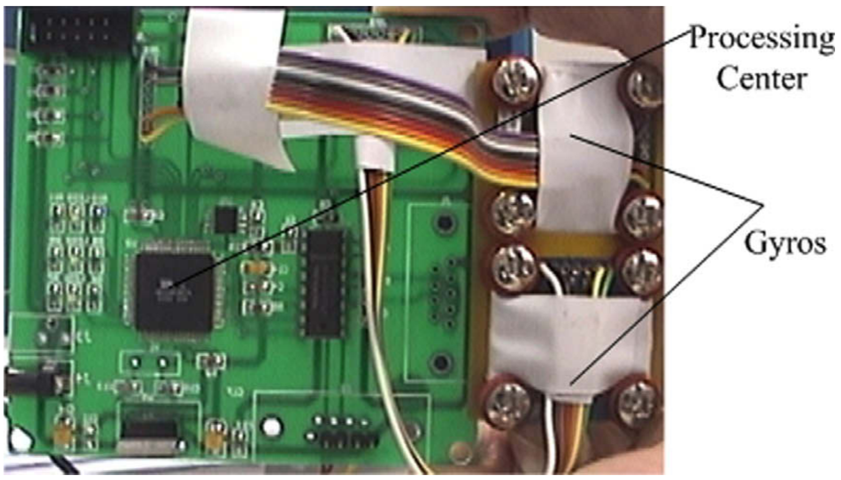


Fig. 17. Camera stabilization board.

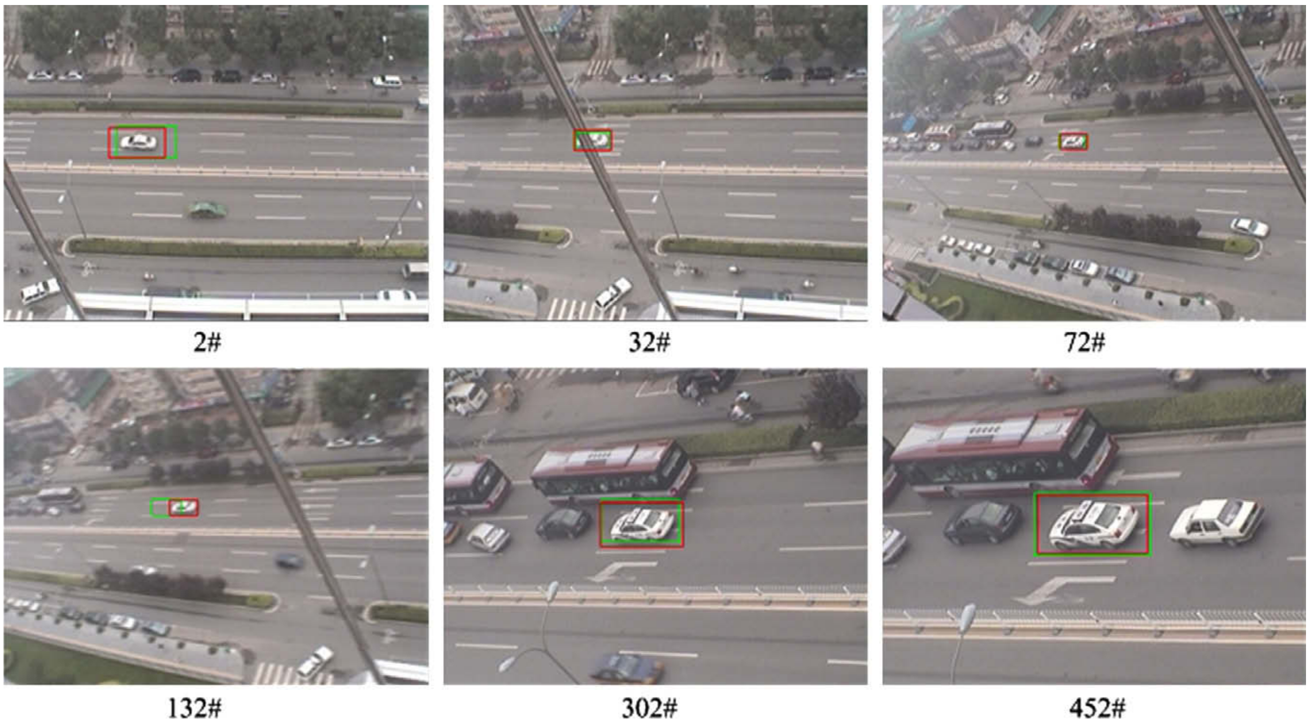


Fig. 18. Typical result sequence of the T&P subsystem is shown. The green box indicates the state in the previous frame, and the red box is the tracking result of the current frame.

from a single point which is easier to operate. First, target model is generated using the method described in Section 2 by selecting a small region around the given point as object and a large area away as backgrounds. Then the block surrounding the point is segmented using the acquired target model by choosing a proper threshold (0.01 in our experiments). Connected component analysis (CCA) [4] is followed to get the bounding box of the largest object. Finally, a larger version of the bounding box (1.4 times of the original) is used to initialize the tracker to compensate localization errors. This method works well in our T&P subsystem.

The subsystem is tested on roof of a 13th floor building to simulate the AVS circumstances. The P/T/Z camera is held by hand. Once the object is initialized by selecting a point, the system can generate the bounding box and keep the object in the center of the view using the above localization and tracking algorithms together with the camera control units. When the object is around the center, the camera zooms out automatically to get fine details. Fig. 18 shows a typical output sequence of the proposed T&P subsystem. The system locks the car successfully and gets fine details of it for high level use such as object recognition.

5. Conclusions and discussion

In this paper, adaptive pyramid analysis as well as a feature selection trick is employed for MS based target tracking, termed APMS. Improvements over normal MS include global seeking ability, better model stability, and adaptive object size. Comparing to annealed MS, APMS is more efficient and accurate. Promising results are obtained in video tracking and T&P subsystem.

A problem for APMS may exist when illumination changes seriously. Model drifts influent the iteration times as discussed in Section 4.1, and the process may be interrupted when illumination changes seriously. There are two possible ways to solve it. One is to select features that are invariant to illumination and view angle changes as target model and candidate, and the other is to employ appropriate model updating strategy. Edge orientation histograms [18,29], corners calculated by FAST [30], multi-feature [31,34,35] or multi-sensor [32] fusing techniques for feature selection, as well as adaptive weighted average [33], weighted composite reference function (WCRF) [26] or ensemble update [27] techniques for model updating will be explored in future work. Consideration will also be given to exploring the application of the proposed method to other surveillance systems.

Acknowledgments

This work is supported by China 863 Programming (2007AA04Z233). The authors thank the anonymous reviewers for their constructive advice and for the revision of the paper.

References

[1] S. Avidan, Support vector tracking, in: IEEE conference on Computer Vision and Pattern Recognition, vol. I, 2001, pp. 184–191.
[2] R. Kumar, H. Sawhney, S. Samarasekera, et al., Aerial video surveillance and exploitation, IEEE Third Generation Surveill. Syst. 89 (10) (2001) 1518–1539.
[3] Y. Guo, S. Hsu, H.S. Sawhney, et al., Robust object matching for persistent tracking with heterogeneous features, IEEE Trans. Pattern Anal. Mach. Intell. 29 (5) (2007) 824–839.
[4] D.A. Forsyth, J. Ponce, Computer Vision – A Modern Approach, Pearson Education Inc., 2003.
[5] S. Julier, J. Uhlmann, A new extension of the Kalman filter to nonlinear systems, SPIE 3068 (1997) 182–193.
[6] Y. Chen, Y. Rui, T.S. Huang, Multicue HMM-UKF for real-time contour tracking, IEEE Trans. Pattern Anal. Mach. Intell. 28 (9) (2006) 1525–1529.
[7] S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking, IEEE Trans. Signal Process. 50 (2) (2002) 174–188.
[8] Z. Khan, T. Balch, F. Dellaert, MCMC-based particle filtering for tracking a variable number of interacting targets, IEEE Trans. Pattern Anal. Mach. Intell. 27 (11) (2005) 1805–1819.
[9] Z. Khan, T. Balch, F. Dellaert, MCMC data association and sparse factorization updating for real time multi-target tracking with merged and multiple measurements, IEEE Trans. Pattern Anal. Mach. Intell. 28 (12) (2006) 1960–1972.
[10] X. Xu, B. Li, Adaptive Rao-Blackwellized particle filter and its evaluation for tracking in surveillance, IEEE Trans. Image Process. 16 (3) (2007) 838–849.

[11] B. Georgescu, I. Shimshoni, P. Meer, Mean shift based clustering in high dimensions: a texture classification example, in: IEEE International Conference on Computer Vision, vol. 2, 2003, pp. 456–463.
[12] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (5) (2002) 603–619.
[13] R. Collins, Mean shift blob tracking through scale space, in: IEEE conference on Computer Vision and Pattern Recognition, vol. 2, 2003, pp. 234–240.
[14] D. Comaniciu, V. Ramesh, P. Meer, Kernel-based object tracking, IEEE Trans. Pattern Anal. Mach. Intell. 25 (5) (2003) 564–577.
[15] Y. Cheng, Mean shift, mode seeking, and clustering, IEEE Trans. Pattern Anal. Mach. Intell. 17 (8) (1995) 790–799.
[16] M. Fashing, C. Tomasi, Mean shift is a bound optimization, IEEE Trans. Pattern Anal. Mach. Intell. 27 (3) (2005) 471–474.
[17] R. Collins, Y. Liu, M. Eeordeanu, Online selection of discriminative tracking features, IEEE Trans. Pattern Anal. Mach. Intell. 27 (10) (2005) 1631–1643.
[18] T. Liu, H. Chen, Real-time tracking using trust-region methods, IEEE Trans. Pattern Anal. Mach. Intell. 26 (3) (2004) 397–402.
[19] C. Shen, M. Brooks, A. Hengel, Fast global kernel density mode seeking with applications to localization and tracking, in: IEEE International Conference on Computer Vision, vol. 2, 2005, pp. 1516–1523.
[20] C. Shen, M. Brooks, A. Hengel, Fast global kernel density mode seeking: applications to localization and tracking, IEEE Trans. Image Process. 16 (5) (2007) 1457–1469.
[21] C. Xu, S. Sun, An Introduction to Calculation Methods, second ed., High Education Press, Beijing, China, 2002.
[22] P. Hall, M.C. Minnotte, C. Zhang, Bump hunting with non-Gaussian kernels, Ann. Stat. 32 (5) (2004) 2124–2141.
[23] F. Zhang, X.B. Cao, J.X. Zou, A new large-scale transformation algorithm of quaternion to Euler angle, Chin. J. Nanjing Univ. Sci. Technol. 24 (4) (2002) 376–380.
[24] Y.Z. Liu, X.M. Ma, Quaternion-based adaptive control of a spacecraft with reaction wheels, Chin. J. Shanghai Jiaotong Univ. 37 (12) (2003) 1957–1960.
[25] M.G. Wang, J.Q. Zhai, Design and simulation of miniature AHRs based on DSP, Chin. J. Comput. Simul. 22 (8) (2005) 50–53.
[26] A. Dawoud, M.S. Alam, A. Bal, C. Loo, Target tracking in infrared imagery using weighted composite reference function-based decision fusion, IEEE Trans. Image Process. 15 (2) (2006) 404–410.
[27] S. Avidan, Ensemble tracking, IEEE Trans. Pattern Anal. Mach. Intell. 29 (2) (2007) 261–271.
[28] Z.H. Sun, G. Bebis, R. Miller, On-road vehicle detection: a review, IEEE Trans. Pattern Anal. Mach. Intell. 28 (5) (2006) 694–711.
[29] C.J. Yang, R. Duraiswami, L. Davis, Fast multiple object tracking via a hierarchical particle filter, in: IEEE International Conference on Computer Vision, 2005, pp. 212–219.
[30] E. Rosten, T. Drummond, Fusing points and lines for high performance tracking, in: IEEE International Conference on Computer Vision, 2005, pp. 1508–1515.
[31] J.Q. Wang, Y.S. Yagi, Integrating color and shape-texture features for adaptive real-time object tracking, IEEE Trans. Image Process. 17 (2) (2008) 235–240.
[32] J.S. Cui, H.B. Zha, H.J. Zhao, R. Shibasaki, Multi-model tracking of people using laser scanners and video camera, Image Vis. Comput. 26 (2) (2008) 240–252.
[33] R.V. Babu, P. Perez, P. Bouthemy, Robust tracking with motion estimation and local kernel based color modeling, Image Vis. Comput. 25 (8) (2007) 1205–1216.
[34] J. Jeyakar, R.V. Babu, K.R. Ramakrishnan, Robust object tracking with background-weighted local kernels, Comput. Vis. Image Understand. 112 (3) (2008) 296–309.
[35] I. Leichter, M. Lindenbaum, E. Rivlin, Tracking by affine kernel transformations using color and boundary cues, IEEE Trans. Pattern Anal. Mach. Intell. 31 (1) (2009) 164–171.
[36] Z.L. Jiang, S.F. Li, D.F. Gao, An adaptive mean shift tracking method using multi-scale images, in: International Conference on Wavelet Analysis and Pattern Recognition, 2007, pp. 1060–1066.