Int J Comput Vis
DOI 10.1007/s11263-015-0819-8

A Robust Tracking System for Low Frame Rate Video

Xiaoqin Zhang[1] · Weiming Hu[2] · Nianhua Xie[2] · Hujun Bao[3] · Stephen Maybank[4]

Received: 17 April 2013 / Accepted: 18 March 2015

Abstract Tracking in low frame rate (LFR) videos is one of the most important problems in the tracking literature. Most existing approaches treat LFR video tracking as an abrupt motion tracking problem. However, in LFR video tracking applications, LFR not only causes abrupt motions, but also large appearance changes of objects because the objects' poses and the illumination may undergo large changes from one frame to the next. This adds extra difficulties to LFR video tracking. In this paper, we propose a robust and general tracking system for LFR videos. The tracking system consists of four major parts: dominant color-spatial based object representation, bin-ratio based similarity measure, annealed particle swarm optimization (PSO) based searching, and an integral image based parameter calculation. The first two parts are combined to provide a good solution to the appearance changes, and the abrupt motion is effectively captured by the annealed PSO based searching. Moreover, an integral image of model parameters is constructed, which provides a look-up table for parameters calculation. This greatly reduces the computational load. Experimental results demonstrate that the proposed tracking system can effectively tackle the difficulties caused by LFR.

Keywords Low frame rate · Tracking · Dominant color · Bin-ratio matching metric · Particle swarm optimization

Communicated by M. Hebert.

✉ Xiaoqin Zhang
zhangxiaoqinnan@gmail.com

Weiming Hu
wmhu@nlpr.ia.ac.cn

Nianhua Xie
nhxie@nlpr.ia.ac.cn

Hujun Bao
bao@cad.zju.edu.cn

Stephen Maybank
sjmaybank@dcs.bbk.ac.uk

[1] Institute of Intelligent System and Decision, Wenzhou University, Zhejiang, China

[2] National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, China

[3] Department of Computer Science, Zhejiang University, Zhejiang, China

[4] Department of Computer Science and Information Systems, Birkbeck College, London, UK

1 Introduction

Tracking in LFR videos has received more and more attention because of its wide applications in micro embedded systems (e.g. mobile vision systems and car vision systems) and visual surveillance. There are two major situations in which LFR videos are produced: (1) LFR videos may be produced when image frames are missed because of hardware delay in the image acquisition system or the limitation of transmission bandwidth; (2) the frame rate of the data streams may be down-sampled because of limitations in the storage or the processing power of the CPU. The design of robust tracking systems for LFR videos is an important and challenging issue in the tracking literature.

The frame rate for LFR video is usually less than ten frames per second, resulting in large appearance changes and abrupt motion between successive image frames. However, most traditional tracking algorithms are based on the state continuity hypothesis, in which it is assumed that the changes in object motion and appearance in successive image

Published online: 08 April 2015

frames are small. In typical applications of particle filtering to tracking (Isard and Blake 1998), motion prediction is based on the tracking results in the previous frame. For the iterative optimization based tracking algorithms, such as Kanade–Lucas–Tomasi (KLT) (Tomasi and Kanade 1991), template matching algorithm (Hager and Belhumeur 1998) and mean shift (Comaniciu et al. 2003), the initialization of the optimizer is based on the tracking results in the previous frame. So the above classical tracking algorithms do not yield satisfactory results when applied to LFR video.

Tracking in LFR videos has been much less investigated than tracking in full frame rate videos. Porikli and Tuzel (2005, 2006), extend the mean shift algorithm to a multi-kernel mean shift algorithm, and apply it to the motion map which is obtained from background subtraction, in order to overcome the abrupt motion caused by LFR video. A cascade particle filter which integrates conventional tracking and detection is proposed by Li et al. (2008) for LFR video tracking. In this approach, detectors learned from different ranges of samples are adopted to detect the moving object. This approach requires complex detectors and time-consuming off-line training, and the detectors only work well in face tracking. In Carrano (2009), the object motion is detected from background subtraction, and then four features, namely phase cross correlation, average intensity difference, velocity difference, and angle difference are combined for matching between consecutive image frames. Zhang et al. (2009) adopt region based image differencing to estimate the object motion. The predicted motion is used to guide the particle propagation in a particle filter. In summary, all of the above work assumes that the only problem with LFR tracking is abrupt motion. However, in practice, LFR not only causes abrupt motion, but also large appearance changes because the object pose and the illumination may undergo large changes from one frame to the next. As a result, it is necessary to take both the abrupt motion and the appearance changes of the object into consideration. In order to design a robust tracking system for LFR videos, we start with the following two requirements: (1) robust appearance modeling, to deal with the changes in object pose and illumination; (2) effective searching methods, to capture the abrupt motion of the object. Below, we will investigate the related work on appearance modeling and motion searching.

1.1 Related Work

1.1.1 Appearance Model

The appearance model of the object is a basic issue to be considered in tracking algorithms. An image patch model (Hager and Belhumeur 1998), which takes the set of pixels in the target region as the model representation, is a direct way to model the target, but it loses the discriminative information that is contained in the pixel values. The color histogram Comaniciu et al. (2003), Nummiaro et al. (2003) provides global statistical information about the target region which is robust to noise, but it has two major problems: (1) the histogram is very sensitive to illumination changes; (2) the relative positions of the pixels in the image are ignored. A consequence of (2) is that trackers based on color histograms are prone to lose track if the object is near to other objects with a similar appearance. In Isard and Blake (1998), curves or splines are used to represent the apparent boundary of the object, and the Condensation algorithm is developed for contour-based tracking. Due to the simplistic representation, which is confined to the apparent boundary, the algorithm is sensitive to image noise, leading to tracking failures in cluttered backgrounds. Stauffer and Grimson (1999) employ a Gaussian mixture model (GMM) to represent and recover the appearance changes in consecutive frames. Jepson et al. (2003) develop a more elaborate Gaussian mixture model which consists of three components S, W, L, where the S component models temporally stable images, the W component models the two-frame variations, and the L component models data outliers, for example those caused by occlusion. An online EM algorithm is employed to explicitly model appearance changes during tracking. Later, Zhou et al. (2004) replace the component L with a component F, which is a fixed template of the target, to prevent the tracker from drifting away from the target. This appearance based adaptive model is embedded into a particle filter to achieve robust visual tracking. Wang et al. (2007) present an adaptive appearance model based on a mixture of Gaussians model in a joint spatial-color space, referred to as SMOG. SMOG captures rich spatial layout and color information. However, these GMM based appearance models consider each pixel independently and with the same level of confidence, which is not reasonable in practice. In Porikli et al. (2006), the object to be tracked is represented by a covariance descriptor which enables efficient fusion of different types of features and modalities. Another category of appearance models is based on subspace learning. For example, in Black and Jepson (2004), a view-based eigenbasis representation of the object is learned off-line, and applied to form a tracking algorithm which matches successive views of the object. However, it is very difficult to collect training samples that cover all possible viewing conditions. Therefore, this algorithm is only feasible under those conditions for which training data has been obtained. Later, some researchers update the object subspace in the tracking process to capture the changes of the appearance. The pioneering work on applying the incremental subspace learning to tracking is by Lim et al. (2004), where they extend the Sequential Karhunen–Loeve (SKL) (Levy and Lindenbaum 2000) algorithm to effectively learn the variations of both appearance and illumination in an incremental way.

However, the aforementioned appearance models ignore the background information and do not perform well when the background is noisy and cluttered. In order to deal with these cases, a set of discriminative based appearance models are proposed (Collins et al. 2005; Lin et al. 2004; Avidan 2004; Avidan 2007; Grabner et al. 2006; Grabner et al. 2008; Saffari et al. 2010; Babenko et al. 2011). For example, Collins et al. (2005) firstly note the importance of background information for object tracking, and formulate tracking as a binary classification problem between the tracked object and its surrounding background. In Lin et al. (2004), a two-class Fisher discriminant analysis (FDA) based model is proposed to learn a discriminative subspace to separate the object from the background. In Avidan (2007), an ensemble of online weak classifiers are combined into a strong classifier. A probability map is constructed by the classifier to represent the probabilities of the pixels belonging to the object or the background. Grabner et al. (2006) adopt online boosting to select discriminative local tracking features. Each selected feature corresponds to a weak classifier that separates the object from the background. Saffari et al. (2010) introduce a novel on-line random forest algorithm for feature selection that allows for on-line building of decision trees. In Babenko et al. (2011), Babenko et al. use multiple instance learning instead of traditional supervised learning to learn the weak classifiers. This strategy is more robust to the drifting problem.

Overall, the complex appearance models (such as the subspace model, the Gaussian mixture model and learning based appearance models) do not adapt well to the large appearance changes, while the simple models (such as color histogram) are not robust and discriminative enough. Since there are large appearance changes of the object in LFR video, it is necessary to find a balance between adaptability and robustness.

1.1.2 Searching Method

Most of the existing tracking algorithms can be formulated as optimization processes, which are typically tackled using either deterministic searching methods (Kass et al. 1988; Hager and Belhumeur 1998; Comaniciu et al. 2003) or stochastic searching methods (Isard and Blake 1998; Nummiaro et al. 2003; Bray et al. 2007). Deterministic searching methods usually involve a gradient descent search to minimize a cost function. The snakes model introduced by Kass et al. (1988) is a good example. The aim is to obtain a tight contour enclosing the object by minimizing an energy function. In Hager and Belhumeur (1998), the cost function is defined as the sum of squared differences between the observation candidate and a fixed template. Then the motion parameters are found by minimizing the cost function through a gradient descent search. Mean shift, which firstly appeared in Fukunaga and Hostetler (1975) as an approach for estimating the gradient of a density function, is applied by Comaniciu et al. (2003) to visual tracking, in which the cost function between two color histograms is minimized through the mean shift iterations. In general, deterministic searching methods are usually computationally efficient but they easily become trapped in local minima. In contrast, stochastic searching methods have a higher probability of reaching the global optimum of the cost function. For example, in Isard and Blake (1998), Nummiaro et al. (2003), object tracking is viewed as an online Bayesian inference problem, which is solved by randomly generating a large number of particles to find the maximum of the posterior distribution. Bray et al. (2007) use the stochastic meta-descent strategy to adapt the step size of the gradient descent search, and thus avoid local minima of the optimization process, when applied to articulated structure tracking. Leung and Gong (2007) incorporate random subsampling into mean shift tracking to boost its efficiency and robustness for low-resolution video sequences. Compared with the deterministic counterparts, stochastic searching methods are usually more robust, but they incur a large computational load, especially if the state space has a high dimension.

As stated in Sect. 1, both the deterministic searching methods and stochastic searching methods are based on the state continuity hypothesis in tracking applications. For this reason, they do not yield satisfactory results in case of LFR video.

1.2 Our Work

In view of the forgoing discussions, we analyze the tracking problem in case of LFR video from the following three aspects: (1) object representation, (2) matching criterion, (3) searching method, and design a robust and general tracking system for LFR video. Although our tracking system is designed with the LFR video in mind, it can handle the similar tracking problems in the normal video arising from fast moving objects or camera motion. The main contributions of our work are:

1. The object is represented by the dominant color-spatial distribution of the object region. This representation can extract the dominant color modes of the object region, and simultaneously remove the noisy image pixels. In addition, the spatial distribution of the dominant color mode is utilized to improve the discriminative ability of the representation.
2. A bin-ratio based similarity measure is used to evaluate the difference between the dominant color of the object template and the dominant color of the candidate region. This measure is robust under (1) illumination changes, (2) partial occlusion, (3) contamination by background pixels.

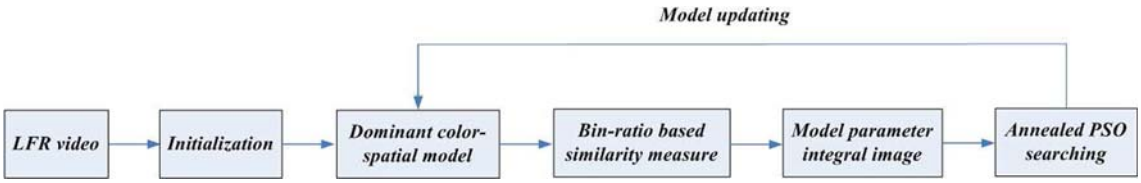


Fig. 1 The flow chart of the proposed tracking system

3. In order to handle the abrupt motions caused by LFR video, we propose an annealed particle swarm optimization (PSO) method. Inspired by the social behavior of bird flocking, the particles in this searching method cooperate and communicate with each other, and the shared information guides the evolution of the particles. This mechanism realizes robust searching for the abrupt motions.
4. Each particle specifies a candidate image region. The fitness value of a particle is evaluated during the PSO iteration process, using the similarity between the candidate image region and the object template. Many candidate image regions may overlap, and the image pixels inside the overlapping region will be used in many separate calculations of the similarity between the candidate image region and the object template. This involves a large and unnecessary computational load. In order to avoid the unnecessary computation, we calculate a integral image of model parameters which establishes a parameter-particle look-up table. In this way, each pixel inside the candidate region needs to be calculated only once. As a result, the computational complexity is greatly reduced.

An overview of the proposed tracking system for LFR video is systematically presented in Fig. 1. There are four major components in the proposed tracking system: (1) dominant color-spatial based object representation, (2) bin-ratio based similarity measure, (3) integral image of model parameters, (4) annealed PSO based searching process. We will give a detailed description of each component in the following sections.

The rest of the paper is organized as follows. The dominant color-spatial based object representation is described in Sect. 2. The bin-ratio based similarity measure is introduced in Sect. 3. The annealed PSO searching method and integral image of model parameters are presented in Sects. 4 and 5 respectively. Experimental results are shown in Sects. 6 and 7 is devoted to conclusion.

2 Dominant Color-Spatial Based Object Representation

In our work, the major color modes inside the object region are firstly obtained by a dominant-set based clustering algorithm, and then the spatial distribution of each color mode is extracted to enhance the discriminative ability of the color model.

2.1 Dominant-Set Based Clustering Algorithm

Suppose given an undirected edge-weighted graph $G = (V, E, A)$, where V is the vertex set, E is set of weighted edges that link different vertices, and A is a symmetric similarity matrix in which a_{ij} represents the similarities between vertex i and vertex j. The aim of dominant-set clustering is to cluster the vertices in V according to the similarity matrix A. Concepts and algorithms for dominant-set clustering are briefly introduced hereinafter for the convenience of the reader.

2.1.1 Concept of Dominant Set

A dominant set, as defined by Pavan and Pelillo (2003), is a combinatorial concept in graph theory that generalizes the notion of a maximal complete subgraph to edge-weighted graphs. The characteristics of dominant-set clustering rest with the definition of dominant sets.

Specifically, let $S \subseteq V$ be a nonempty subset of vertices. For any vertex $i \in S$, the average weighted degree of i relative to S is defined as

$$D_S(i) = \frac{1}{|S|} \sum_{j \in S} a_{ij} \tag{1}$$

where $|S|$ is the number of vertices in S. For a vertex $j \notin S$, the similarity $\phi_S(i, j)$ between vertices i and j relative to S is defined as $\phi_S(i, j) = a_{ij} - D_S(i)$. Then, the weight $w_S(i)$ of $i \in S$ relative to S is defined as

$$w_S(i) = \begin{cases} 1, & \textbf{if } |S| = 1 \\ \sum_{j \in S \setminus \{i\}} \phi_{S \setminus \{i\}}(j, i) w_{S \setminus \{i\}}(j), & \textbf{otherwise} \end{cases} \tag{2}$$

Intuitively, $w_S(i)$ measures the relative similarity between vertex i and the vertices of $S \setminus \{i\}$ with respect to the overall similarity among the vertices in $S \setminus \{i\}$. Based on the definition of $w_S(i)$, the total weight of S is defined as

$$W(S) = \sum_{i \in S} w_S(i) \tag{3}$$

The set S is defined as a dominant set if it satisfies the following conditions (Pavan and Pelillo 2003): (1) $\forall T \subset S$, $W(T) > 0$; (2) $\forall i \in S$, $w_S(i) > 0$; and (3) $\forall i \notin S$, $w_S(i) < 0$. Condition (1) indicates that vertices in each subset of S are closely and firmly united. Condition (2) indicates that S has large attraction to each vertex in S. Condition (3) indicates that S has no large attraction to any vertex outside S. Conditions (1) and (2) describe the internal homogeneity of S. Condition (3) describes the external heterogeneity of S. The definition of a dominant set simultaneously emphasizes internal homogeneity and external inhomogeneity, and thus is considered to be a general definition of "cluster".

2.1.2 Clustering Algorithm

Pavan and Pelillo (2003) establish an intriguing connection between the dominant set and a quadratic optimization problem as follows:

$$\begin{aligned} &\text{maximize} && g(\mathbf{z}) = \mathbf{z}^T A \mathbf{z} \\ &\text{subject to} && \mathbf{z} \in \Delta \end{aligned} \tag{4}$$

where $\mathbf{z}$ is the indictor vector of samples in the clustering process that satisfies

$$\Delta = \left\{ \mathbf{z} \in \mathbb{R}^n : z_i \geq 0 \text{ and } \sum_{i=1}^{n} z_i = 1 \right\}.$$

A is the similarity matrix of the input samples, and $g(\cdot)$ is the objective function which defines the cohesiveness of a cluster in a natural way. Let $\mathbf{z}^*$ denote a strict local solution of (4), and let $\sigma(\mathbf{z}^*)$ be the vertex support set of $\mathbf{z}^*$: $\sigma(\mathbf{z}^*) = \{i | z_i^* > 0\}$. It has been proved in Pavan and Pelillo (2003) that the vertex support set $\sigma(\mathbf{z}^*)$ corresponds to a dominant set in the graph represented by A. This allows us to formulate the pairwise clustering problem as the problem of finding a vector $\mathbf{z}^*$ that solves (4). The value $g(\mathbf{z}^*)$ at the local maximum indicates the "cohesiveness" of the dominant-set cluster specified by $\mathbf{z}^*$.

Pavan and Pelillo (2003) use the following iterative equation to solve (4)

$$z_i(t+1) = z_i(t) \frac{(A\mathbf{z}(t))_i}{\mathbf{z}(t)^T A \mathbf{z}(t)}, \quad i = 1, \ldots, n \tag{5}$$

where t indexes the number of iterations. Meanwhile, the authors prove Pavan and Pelillo (2003) that the trajectory $\mathbf{z}(t)$ generated by Eq. (5) converges to a strict local maximizer of program (4).

Table 1 Dominant-set clustering algorithm

Input: the similarity matrix A

1. Initialize $A^k, k = 1$ with A
2. Calculate the local solution of (4) by (5): $\mathbf{z}_k^*$ *and* $g(\mathbf{z}_k^*)$
3. Get the dominant set: $S_k = \sigma(\mathbf{z}_k^*)$
4. Split out S_k from A^k and get a smaller similarity matrix A^{k+1}
5. If A^{k+1} is not a null matrix, $A^k \leftarrow A^{k+1}$ and $k = k + 1$, then go to step 2; else exit

Output: $\cup_{l=1}^{k} \{S_l, \mathbf{z}_l^*, g(\mathbf{z}_l^*)\}$

Based on the above discussion, dominant set clustering can be conducted in a bipartition way. To cluster the samples, a dominant set of the weighted graph is found by Eqs. (4) and (5) and then removed from the graph. This process is iterated until the graph is empty. Each dominant set defines a cluster. Table 1 contains the algorithm for the clustering process. Therefore, dominant-set clustering has two advantages: (1) In contrast with many traditional clustering algorithms (k-means and mean shift), it can automatically determine the number of the clusters; (2) the computational demand is low compared with that of other graph-theoretic clustering algorithms that rely on an eigen analysis of A.

2.1.3 Fast Assignment Algorithm

To group any new samples obtained after the clustering process has taken place, Pavan and Pelillo (2005) propose a fast assignment algorithm which does not need to conduct a new dominant-set clustering. Let $S \subset V$ be a subset of vertices which is dominant in the original graph G and let i be a new data vertex. As stated in Sect. 2.1.1, the sign of $w_{S\cup\{i\}}(i)$ provides an indication as to whether i is tightly or loosely coupled with the vertices in S. However, if only the sign of $w_{S\cup\{i\}}(i)$ is considered, then the same point can be assigned to more than one class. This ambiguity is removed as follows. The degree of participation of i in $S \cup \{i\}$ is given by the ratio between $w_{S\cup\{i\}}(i)$ and $W(S \cup \{i\})$. Since computing the exact value of the ratio $w_{S\cup\{i\}}(i)/W(S \cup \{i\})$ is computationally expensive, a simple approximation formulas is provided in Pavan and Pelillo (2005).

$$\frac{w_{S\cup\{i\}}(i)}{W(S \cup \{i\})} \approx \frac{|S| - 1}{|S| + 1} \left(\frac{\boldsymbol{\alpha}^T \mathbf{z}_S^*}{g(\mathbf{z}_S^*)} - 1 \right) \tag{6}$$

where $\boldsymbol{\alpha}$ is an affinity vector containing the similarities between the new sample i and the original n samples, and $\mathbf{z}_S^*$ is the solution of problem (4) to obtain the dominant set S. The new sample i is assigned to the cluster with the largest value of $w_{S\cup\{i\}}(i)/W(S \cup \{i\})$. The detail of the fast assignment algorithm is shown in Table 2.

Table 2 Dominant-set fast assignment algorithm

Input: Affinity vector $\boldsymbol{\alpha} \in \mathbb{R}^n$, $\cup_{l=1}^{k}\{S_l, \mathbf{z}_l^*, g(\mathbf{z}_l^*)\}$

1. Compute $b_l = \frac{|S_l|-1}{|S_l|+1}\left(\frac{\boldsymbol{\alpha}^T\mathbf{z}_l^*}{g(\mathbf{z}_l^*)} - 1\right), l \in \{1, \ldots, k\}$
2. $l^* = \text{argmax}_l\, b_l$
3. If $b_{l^*} > 0$, assign $\boldsymbol{\alpha}$ to the cluster S_{l^*}; else $l^* = 0$, $\boldsymbol{\alpha}$ is consider as an outlier

Output: l^*

2.2 Dominant Color Mode Extraction

Given a image region, we first convert the *RGB* color space to the *rgI* space using the following formulas:

$$r = R/(R+G+B),\ g = G/(R+G+B),$$
$$I = (R+G+B)/3$$

Then we define the pixel-pairwise graph, where the weight a_{ij} on the edge connecting node i and node j is calculated by:[1]

$$a_{ij} = c\exp(-||\mathbf{f}_i - \mathbf{f}_j||^2) \tag{7}$$

where $\mathbf{f}_i = (r, g, I)$ is the intensity value of pixel i in the *rgI* color space, and c is the normalization factor. Then, we apply the dominant set based clustering algorithm to the constructed pixel-pairwise graph, and obtain the dominant color modes $\{S_l, \mathbf{z}_l\}_{l=1}^{k}$. In tracking applications, if all the pixels are clustered by a dominant set based clustering algorithm, then there will be many small clusters. Therefore the clustering is stopped if the number of the remaining pixels (vertices) is less than 5 % of the total pixel number inside the object region. This criterion has two advantages: (1) the outliers can be effectively filtered out; (2) the number of dominant colors is not strongly affected by the size of the object.

2.3 Spatial Layout of the Dominant Color

The above dominant color based representation captures the color distribution in the image region of interest. However, the spatial layout of pixels falling into the same color mode is ignored. In order to overcome this problem, the spatial mean $\boldsymbol{\mu}_l$ and the variance $\boldsymbol{\Sigma}_l$ of the lth color mode are extracted as follows.

$$\boldsymbol{\mu}_l = \frac{\sum_i \mathbf{p}_i\delta(L(i)-l)}{\sum_i \delta(L(i)-l)} \tag{8}$$

$$\boldsymbol{\Sigma}_l = \frac{\sum_i (\mathbf{p}_i - \boldsymbol{\mu}_l)(\mathbf{p}_i - \boldsymbol{\mu}_l)^T\delta(L(i)-l)}{\sum_i \delta(L(i)-l)} \tag{9}$$

[1] Here, the component I is normalized as $I/255$.

where $\mathbf{p}_i$ is the coordinate of the pixel i relative to the center position of the image region, and $L(i)$ is a label function to assign the pixel i to a given cluster. $\delta(\cdot)$ is the Kronecker function such that $\delta(L(i)-l) = 1$ if $L(i) = l$ and $\delta(L(i)-l) = 0$ otherwise.

As a result, the image region of interest can be represented by $\{\omega_l, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\}_{l=1}^{k}$, where ω_l is the weight of the lth dominant color mode and is calculated as follows:

$$\omega_l = \frac{\sum_i \delta(L(i)-l)}{\sum_{l=1}^{k}\sum_i \delta(L(i)-l)} \tag{10}$$

3 Similarity Measure for Matching

3.1 Bin-Ratio Based Color Distance Measure

3.1.1 Motivation

Let $\mathbf{u}^O = \{u_l^O\}_{l=1}^{k}$ be a k-bin color histogram that represents the color distribution of the pixels in the object template. The candidate histogram $\mathbf{u}^C = \{u_l^C\}_{l=1}^{k}$ is calculated as follows: first, the pixels inside the candidate image patch are assigned to the bins of the object histogram $\mathbf{u}^O$ using the fast assignment algorithm in Table 2, and then $\mathbf{u}^C$ is obtained according to the results of the assignment. Traditionally, the similarity between $\mathbf{u}^O = \{u_l^O\}_{l=1}^{k}$ and $\mathbf{u}^C = \{u_l^C\}_{l=1}^{k}$ is evaluated by the Bhattacharyya measure Comaniciu et al. (2003): $\rho(\mathbf{u}^O, \mathbf{u}^C) = \sum_{l=1}^{k}\sqrt{u_l^O u_l^C}$, or the histogram intersection measure Swain and Ballard (1991): $Q(\mathbf{u}^O, \mathbf{u}^C) = \sum_{l=1}^{k}\min(u_l^O, u_l^C)$. However, these measures are sensitive to illumination changes and image clutter. In tracking applications, as the object region is often represented by a rectangle, the histograms are often corrupted by background clutter that is irrelevant to the object. For example, as shown in Fig. 2, a large portion of pixels represented with blue

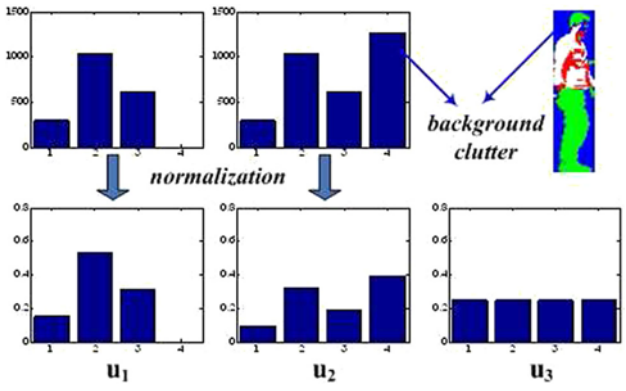


Fig. 2 An example of background clutter problem ($\mathbf{u}_1$: histogram without background pixels, $\mathbf{u}_2$: histogram with background pixels, $\mathbf{u}_3$: uniformly distributed histogram)

are background, which fall into the fourth bin of histogram. We can see that $\mathbf{u}_1$ and $\mathbf{u}_2$ look very different after normalization, which means pixels from the background introduce noise into the histogram which may cause inaccurate matching.

To validate the above claim, we test the above two measures on this example. By introducing a uniformly distributed histogram $\mathbf{u}_3$ for reference, the following results are obtained: $\rho(\mathbf{u}_1, \mathbf{u}_2) = 0.24 < \rho(\mathbf{u}_1, \mathbf{u}_3) = 0.25$, $Q(\mathbf{u}_1, \mathbf{u}_2) = 0.6 < Q(\mathbf{u}_1, \mathbf{u}_3) = 0.65$. Both $\rho(\cdot, \cdot)$ and $Q(\cdot, \cdot)$ are similarity measures which means they can not find a correct match among $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$ in this case.

3.1.2 Proposed Distance Measure

To overcome these drawbacks, we introduce a bin-ratio based color distance measure, which was firstly proposed for category and scene classification in our previous work Xie et al. (2010). For a k-bin histogram $\mathbf{u} = \{u_l\}_{l=1}^k$, a $k*k$ ratio matrix $\mathbf{B}_r$ is defined using the ratios of paired bin values. Each element in the matrix has the form (u_i/u_j) which measures the ratio of bin i and j. The ratio matrix is shown as follows:

$$\mathbf{B}_r = \left(\frac{u_i}{u_j}\right)_{i,j} = \begin{bmatrix} \frac{u_1}{u_1} & \frac{u_2}{u_1} & \cdots & \frac{u_k}{u_1} \\ \frac{u_1}{u_2} & \frac{u_2}{u_2} & \cdots & \frac{u_k}{u_2} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{u_1}{u_k} & \frac{u_2}{u_k} & \cdots & \frac{u_k}{u_k} \end{bmatrix} \tag{11}$$

With the definition of the ratio matrix, we compare the vth bin between two histograms $\mathbf{u}^O$ and $\mathbf{u}^C$ using the distance measure M_v, which is defined as the sum of squared difference between the vth rows of the corresponding ratio matrices of $\mathbf{u}^O$ and $\mathbf{u}^C$: $\sum_{l=1}^k \left(\frac{u_l^O}{u_v^O} - \frac{u_l^C}{u_v^C}\right)^2$. The ratio matrix $\mathbf{B}_r$ suffers from the instability problem when its entries are close to zero. To avoid this problem, we include the normalization part $\frac{1}{\mathbf{u}_v^O} + \frac{1}{\mathbf{u}_v^C}$ and define the following distance measure,

$$\begin{aligned} M_v(\mathbf{u}^O, \mathbf{u}^C) &= \sum_{l=1}^k \left(\left(\frac{u_l^O}{u_v^O} - \frac{u_l^C}{u_v^C} \right) / \left(\frac{1}{u_v^O} + \frac{1}{u_v^C} \right) \right)^2 \\ &= \sum_{l=1}^k \left(\frac{u_l^O u_v^C - u_v^O u_l^C}{u_v^O + u_v^C} \right)^2 \end{aligned} \tag{12}$$

As shown in Eq. (12), the numerator $u_l^O u_v^C - u_v^O u_l^C$ can still represent the difference of ratios. The denominator $u_v^O + u_v^C$ is similar to the normalization part in the $\mathcal{X}^2$ distance Puzicha et al. (1997), included to make the distance measure more stable. Based on the $\mathbb{L}_2$ normalization $\sum_{l=1}^k u_l^2 = 1$, the above distance measure can be simplified as follows.

$$\begin{aligned} M_v(\mathbf{u}^O, \mathbf{u}^C) &= \sum_{l=1}^k \left(\frac{u_l^O u_v^C - u_v^O u_l^C}{u_v^O + u_v^C} \right)^2 \\ &= \frac{\sum_{l=1}^k ((u_v^C u_l^O)^2 + (u_v^O u_l^C)^2 - 2u_v^O u_v^C u_l^O u_l^C)}{(u_v^O + u_v^C)^2} \\ &= \frac{(u_v^C)^2 \sum_{l=1}^k (u_l^O)^2 + (u_v^O)^2 \sum_{l=1}^k (u_l^C)^2 - 2u_v^O u_v^C \sum_{l=1}^k (u_l^O u_l^C)}{(u_v^O + u_v^C)^2} \\ &= \frac{(u_v^C)^2 + (u_v^O)^2 - 2u_v^O u_v^C \sum_{l=1}^k (u_l^O u_l^C)}{(u_v^O + u_v^C)^2} \\ &= \frac{(u_v^C + u_v^O)^2 - u_v^O u_v^C (2 + \sum_{l=1}^k 2u_l^O u_l^C)}{(u_v^O + u_v^C)^2} \\ &= \frac{(u_v^C + u_v^O)^2 - u_v^O u_v^C \sum_{l=1}^k ((u_l^O)^2 + (u_l^C)^2 + 2u_l^O u_l^C)}{(u_v^O + u_v^C)^2} \\ &= \frac{(u_v^C + u_v^O)^2 - u_v^O u_v^C \sum_{l=1}^k (u_l^O + u_l^C)^2}{(u_v^O + u_v^C)^2} \\ &= 1 - \frac{u_v^O u_v^C}{(u_v^O + u_v^C)^2} ||\mathbf{u}^O + \mathbf{u}^C||_2^2 \end{aligned}$$

where $||\mathbf{u}^O + \mathbf{u}^C||_2$ is the $\mathbb{L}_2$ norm of $\mathbf{u}^O + \mathbf{u}^C$ and can be obtained before the distance calculation. Therefore, this distance measure has a linear computational complexity in the bin number, as do the Bhattacharyya distance and histogram intersection distance.

The bin-ratio distance between color histograms $\mathbf{u}^O$ and $\mathbf{u}^C$ is formulated as follows:

$$\begin{aligned} \Psi_c(\mathbf{u}^O, \mathbf{u}^C) &= \sum_{v=1}^k M_v(\mathbf{u}^O, \mathbf{u}^C) \\ &= \sum_{v=1}^k \left(1 - \frac{u_v^O u_v^C}{(u_v^O + u_v^C)^2} ||\mathbf{u}^O + \mathbf{u}^C||_2^2 \right) \end{aligned} \tag{13}$$

The above bin-ratio distance measure is robust under 1) illumination changes, 2) partial occlusion, 3) contamination by background pixels. For the example in Fig. 2, the result of the bin-ratio distance measure is $\Psi_c(\mathbf{u}_1, \mathbf{u}_2) = 1.4 < \Psi_c(\mathbf{u}_1, \mathbf{u}_3) = 1.5$ which is correct. These claims are further supported by experiments on both synthetic data and real data as reported in Sect. 6.

3.2 Spatial Distance Measure

As stated in Sect. 1.1.1, the spatial layout information is not included in the color based distance measure. Therefore, we extract the spatial layout information of each dominant color mode, and introduce a spatial distance measure to enhance the robustness of the color based distance measure.

For the object image and a candidate region, the spatial mean and the covariance matrix of each color mode are computed as in Eqs. (8) and (9) accordingly: $\{\boldsymbol{\mu}_l^O, \boldsymbol{\Sigma}_l^O\}_{l=1}^k$, $\{\boldsymbol{\mu}_l^C, \boldsymbol{\Sigma}_l^C\}_{l=1}^k$. Together with the weight of the corresponding color mode, the spatial distance measure $\Psi_s(\mathbf{u}^O, \mathbf{u}^C)$ is defined as follows:

$$\Psi_s(\mathbf{u}^O, \mathbf{u}^C) = \sum_{l=1}^{k} \omega_l \exp\left\{-\frac{1}{2}(\boldsymbol{\mu}_l^O - \boldsymbol{\mu}_l^C)^T(\boldsymbol{\Sigma}_l^O + \boldsymbol{\Sigma}_l^C)^{-1}(\boldsymbol{\mu}_l^O - \boldsymbol{\mu}_l^C)\right\} \tag{14}$$

where the weight ω_l is calculated using the target template as in (10), and updated in the parameter updating process.

As a result, the dominant color-spatial based similarity measure can be defined as follows.

$$\Psi_{cs}(\mathbf{u}^O, \mathbf{u}^C) = \exp\left\{\frac{1}{2\sigma_1^2}(1 - \Psi_s(\mathbf{u}^O, \mathbf{u}^C))\right\} * \exp\left\{-\frac{1}{2\sigma_2^2}\Psi_c(\mathbf{u}^O, \mathbf{u}^C)\right\} \tag{15}$$

where σ_1^2, σ_2^2 are the parameters which balance the influence of spatial distance and bin-ratio distance.

3.3 Parameter Updating

In most tracking applications, the tracker must deal with changes in the appearance of the object, so it is necessary to design an updating scheme for the parameters employed in the appearance model.

The updating process is described in detail as follows.

1. Suppose the tracking result for the current frame has been obtained. Based on the label information and the positions of the pixels inside the image region obtained by tracking results, the parameters $\{\omega_l, \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l\}_{l=1}^k$ of the object model are updated as in Eqs. (8), (9) and (10).
2. There may be outlier pixels left inside the image region obtained by tracking results when applying the dominant-set fast assignment algorithm to the image region. If the number of the outlier pixels exceeds 1/6 of the total number of pixels inside the tracked image region and their cohesiveness value evaluated by $g(\cdot)$ exceeds the cohesiveness value of the smallest cluster in the target template, then these pixels are grouped as a new cluster which is added to the object model; otherwise these pixels are filtered out as noise.

In this way, the parameter updating process makes a trade-off between robustness and adaptation. As a result, it not only effectively captures the variations of the target, but also reliably prevents drifting during tracking.

4 Swarm Intelligence Based Searching Method

The abrupt motions in LFR image sequences cause sample impoverishment when they are tracked using a particle filter. We first investigate the reason for this sample impoverishment and then propose a swarm intelligence based searching method: annealed particle swarm optimization, which can overcome the sample impoverishment problem both in theory and practice.

4.1 Sample Impoverishment Problem of Particle Filter

The particle filter is an on-line Bayesian inference process for estimating an unknown state $\mathbf{s}_t$ at time t from sequential observations $\mathbf{o}_{1:t}$ perturbed by noise. The Bayesian inference process is based on

$$p(\mathbf{s}_t|\mathbf{o}_{1:t}) \propto p(\mathbf{o}_t|\mathbf{s}_t)p(\mathbf{s}_t|\mathbf{o}_{1:t-1}) \tag{16}$$

where the prior $p(\mathbf{s}_t|\mathbf{o}_{1:t-1})$ is the propagation of the previous posterior density along the temporal axis,

$$p(\mathbf{s}_t|\mathbf{o}_{1:t-1}) = \int p(\mathbf{s}_t|\mathbf{s}_{t-1})p(\mathbf{s}_{t-1}|\mathbf{o}_{1:t-1})d\mathbf{s}_{t-1} \tag{17}$$

When the state transition distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ and observation distribution $p(\mathbf{o}_t|\mathbf{s}_t)$ are non-Gaussian, the above integration is intractable and one has to resort to numerical approximations such as particle filters. The basic idea of particle filter is to use a number of particles $\{\mathbf{s}_t^i\}_{i=1}^N$, sampled from the state space, to approximate the posterior distribution by $p(\mathbf{s}_t|\mathbf{o}_{1:t}) = \frac{1}{N}\sum_{i=1}^N \delta(\mathbf{s}_t - \mathbf{s}_t^i)$, where $\delta(\cdot)$ is the Dirac function. Since it is usually impossible to sample from the true posterior, an easy-to-implement distribution, the so-called *proposal distribution* denoted by $q(\cdot)$ is employed, hence $\mathbf{s}_t^i \sim q(\mathbf{s}_t|\mathbf{s}_{t-1}^i, o_{1:t})$, $(i = 1, \ldots, N)$, and each particle's weight is set to

$$\eta_t^i \propto \frac{p(\mathbf{o}_t|\mathbf{s}_t^i)p(\mathbf{s}_t^i|\mathbf{s}_{t-1}^i)}{q(\mathbf{s}_t|\mathbf{s}_{t-1}^i, \mathbf{o}_{1:t})}. \tag{18}$$

Finally, the posterior probability distribution is approximated by $p(\mathbf{s}_t|\mathbf{o}_{1:t}) = \sum_{i=1}^N \eta_t^i \delta(\mathbf{s}_t - \mathbf{s}_t^i)$.

The proposal distribution $q(\cdot)$ is critically important for a successful particle filter because it concerns putting the sampling particles in the useful areas where the posterior is significant. In tracking applications, the state transition distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ is usually taken as the proposal distribution because of its simplicity. To cope with unpredictable motion,

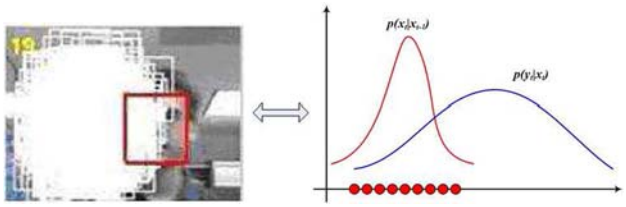


Fig. 3 Sample impoverishment problem

most state transition models use a naive random walk around the previous system state. As shown in Fig. 3, when the object undergoes abrupt motion, the performance of the particle filter is very poor because most particles have low weights, thereby leading to the well-known sample impoverishment problem.

As proved in Doucet et al. (2000), it is shown that the 'optimal' importance proposal distribution is $p(\mathbf{s}_t|\mathbf{s}_{t-1}^i, \mathbf{o}_t)$ in the sense of minimizing the variance of the importance weights. However, in practice, it is impossible to use $p(\mathbf{s}_t|\mathbf{s}_{t-1}^i, \mathbf{o}_t)$ as the proposal distribution in the non-linear and non-Gaussian cases, because it is difficult to sample from $p(\mathbf{s}_t|\mathbf{s}_{t-1}^i, \mathbf{o}_t)$ and to evaluate $p(\mathbf{o}_t|\mathbf{s}_{t-1}^i) = \int p(\mathbf{o}_t|\mathbf{s}_t)\ p(\mathbf{s}_t|\mathbf{s}_{t-1}^i)d\mathbf{s}_t$. So the question is, how to incorporate the current observation $\mathbf{o}_t$ into the transition distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1})$ to form an effective proposal distribution at a reasonable computation cost?

4.2 Annealed Particle Swarm Optimization

In this part, we propose an annealed particle swarm optimization (APSO) algorithm to effectively search for the object motion parameters. From the particle filtering point of view, the searching process in APSO is an approximation to the optimal importance sampling.

4.2.1 Iteration Process

Particle swarm optimization Kennedy and Eberhart (1995), is a population-based stochastic optimization technique, which is inspired by the social behavior of birds flocking. In detail, a PSO algorithm is initialized with a group of random particles $\{\mathbf{s}^{i,0}\}_{i=1}^N$ (N is the number of particles). Each particle $\mathbf{s}^{i,0}$ has a corresponding fitness value $f(\mathbf{s}^{i,0})$ and a relevant velocity $\mathbf{v}^{i,0}$, which is a function of the best state found by that particle ($\mathbf{p}^i$, for individual best), and of the best state found so far among all particles ($\mathbf{g}$, for global best). Given these two best values, the particle updates its velocity and state with the following equations in the nth iteration,

$$\mathbf{v}^{i,n+1} = \xi^n \mathbf{v}^{i,n} + \varphi_1\rho_1(\mathbf{p}^i - \mathbf{s}^{i,n}) + \varphi_2\rho_2(\mathbf{g} - \mathbf{s}^{i,n}) \quad (19)$$

$$\mathbf{s}^{i,n+1} = \mathbf{s}^{i,n} + \mathbf{v}^{i,n+1} \quad (20)$$

where ξ^n is the inertial weight, the φ_1, φ_2 are acceleration constants, and $\rho_1, \rho_2 \in (0, 1)$ are uniformly distributed random numbers. The inertial weight ξ^n is usually a monotonically decreasing function of the iteration number n. For example, given a user-specified maximum weight ξ_{max} , a minimum weight ξ_{min} and the initialization of $\xi^0 = \xi_{max}$, one way to update ξ^n is as follows:

$$\xi^{n+1} = \xi^n - d\xi,\ d\xi = (\xi_{max} - \xi_{min})/n_{max} \quad (21)$$

where n_{max} is the maximum iteration number.

In order to reduce the number of parameters in the above PSO iteration process, we propose an annealed Gaussian version of PSO algorithm (APSO), in which the iteration process is modified as follows:

$$\mathbf{v}^{i,n+1} = |r_1|(\mathbf{p}^i - \mathbf{s}^{i,n}) + |r_2|(\mathbf{g} - \mathbf{s}^{i,n}) + \boldsymbol{\epsilon} \quad (22)$$

$$\mathbf{s}^{i,n+1} = \mathbf{s}^{i,n} + \mathbf{v}^{i,n+1} \quad (23)$$

where $|r_1|$ and $|r_2|$ are the absolute values of the independent samples from the Gaussian probability distribution $\mathcal{N}(0, 1)$, and $\boldsymbol{\epsilon}$ is a zero-mean Gaussian perturbation noise which prevents the particles from becoming trapped in local optima. The covariance matrix of $\boldsymbol{\epsilon}$ is changed in an adaptive simulated annealing way Ingber (1993):

$$\boldsymbol{\Pi}_{\boldsymbol{\epsilon}} = \boldsymbol{\Pi} e^{-cn} \quad (24)$$

where $\boldsymbol{\Pi}$ is the covariance matrix of the predefined transition distribution, c is an annealing constant, and n is the iteration number. The elements in $\boldsymbol{\Pi}_{\boldsymbol{\epsilon}}$ decrease rapidly as the iteration number n increases which enables a fast convergence rate.

Then the fitness value of each particle is evaluated using the observation model $p(\mathbf{o}^{i,n+1}|\mathbf{s}^{i,n+1})$ as follows.

$$f(\mathbf{s}^{i,n+1}) = p(\mathbf{o}^{i,n+1}|\mathbf{s}^{i,n+1}) \quad (25)$$

Here, the observation model $p(\mathbf{o}^{i,n+1}|\mathbf{s}^{i,n+1})$ reflects the similarity between the candidate image observation $\mathbf{o}^{i,n+1}$ and the target model which is defined in Eq. (15).

After the fitness value of each particle $f(\mathbf{s}^{i,n+1})$ is evaluated, the individual best and the global best of particles are updated in the following equations:

$$\mathbf{p}^i = \begin{cases} \mathbf{s}^{i,n+1}, & \textbf{if } f(\mathbf{s}^{i,n+1}) > f(\mathbf{p}^i) \\ \mathbf{p}^i, & \textbf{else} \end{cases} \quad (26)$$

$$\mathbf{g} = \arg\max_{\mathbf{p}^i} f(\mathbf{p}^i) \quad (27)$$

In analogy with the foraging behavior of the bird flocks, here the optimal state of $f(\cdot)$ corresponds to food, and the particles in state space correspond to birds.

As a result, the particles interact locally with one another and with their environment in analogy with the 'cognitive' and 'social' aspects of animal populations, and eventually cluster in the regions where the local optima of $f(\cdot)$ are located. In Eq. (22), the three terms on the right hand side represent *cognitive effect*, *social effect* and noise part respectively, where *cognitive effect* refers to the evolution of the particle according to its own observations, and *social effect* refers to the evolution of the particle according to the cooperation between all particles.

4.2.2 Approximation to Optimal Importance Sampling

When applied to the tracking applications, the sequential information should be incorporated into the annealed PSO algorithm. In particular, the initial particles in the annealed PSO algorithm are firstly sampled from the transition distribution as follows:

$$\mathbf{s}_t^{i,0} \sim \mathcal{N}(\mathbf{p}_{t-1}^i, \boldsymbol{\Sigma}) \tag{28}$$

In our tracking process, resampling is not needed because the individual best of the particle set $\{\mathbf{p}_{t-1}^i\}_{i=1}^N$ converged at time $t-1$ provides a compact sample set for time propagation, and then the annealed PSO iterations are carried out until convergence.

From the particle filtering point of view, the above process is a two-stage sampling strategy to generate samples that approximate to samples from the 'optimal' proposal distribution $p(\mathbf{s}_t|\mathbf{s}_{t-1}^i, \mathbf{o}_t)$: first, in the coarse importance sampling stage, the particles are sampled from the state transition distribution as in (28) to enhance their diversity. In the fine importance sampling stage, the particles evolve through APSO iterations, and are updated according to the newest observations. In fact, this is essentially a latent multi-layer importance sampling process with an implicit proposal distribution. Let $\mathbf{s}_t \in \mathbb{R}^d$ be a d-dimensional state. Let's focus on one APSO iteration in Eqs. (22) and (23). The distribution of the lth element in the vector $|r_1|(\mathbf{p}_t^i - \mathbf{s}_t^{i,n})$ is as follows:

$$|r_1|(p_t^i - s_t^{i,n})_l, \sim \begin{cases} 2\mathcal{N}(0, (p_t^i - s_t^{i,n})_l^2)\ [0, +\infty), & \textbf{if}\ \ (p_t^i - s_t^{i,n})_l \geq 0 \\ 2\mathcal{N}(0, (p_t^i - s_t^{i,n})_l^2)\ (-\infty, 0), & \textbf{otherwise} \end{cases}$$

where $l = 1, \ldots, d$. As shown in Fig. 4, the distribution is defined on half domain of a Gaussian distribution but with two times the usual Gaussian amplitude. Therefore, the distribution of $|r_1|(\mathbf{p}_t^i - \mathbf{s}_t^{i,n})$ is $|r_1|(\mathbf{p}_t^i - \mathbf{s}_t^{i,n}) \sim \mathbf{R}_1$, where $\mathbf{R}_1$ is defined on half domain of a d-dimensional Gaussian distribution with a doubled amplitude as follows.

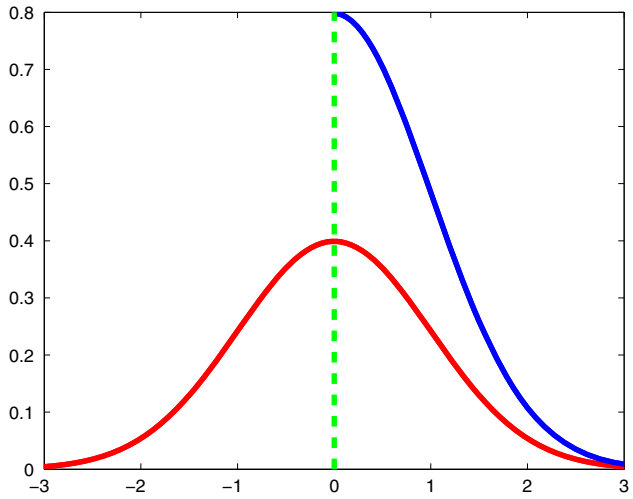


Fig. 4 Pdf for a Gaussian distribution: $\mathcal{N}(0, 1)$ (red) and for a Gaussian distribution conditioned to have support in $[0, +\infty)$: $2\mathcal{N}(0, 1)$ (blue)

$$2\mathcal{N}(0, \boldsymbol{\Sigma}_p),\ \boldsymbol{\Sigma}_p = \begin{pmatrix} (\mathbf{p}_t^i - \mathbf{s}_t^{i,n})_1^2 & \mathbf{0} \\ & \ddots \\ \mathbf{0} & (\mathbf{p}_t^i - \mathbf{s}_t^{i,n})_d^2 \end{pmatrix}$$

Similarly, $|r_2|(\mathbf{g}_t - \mathbf{s}_t^{i,n})$ is distributed on the half domain of the following distribution $\mathbf{R}_2$

$$2\mathcal{N}(0, \boldsymbol{\Sigma}_g),\ \boldsymbol{\Sigma}_g = \begin{pmatrix} (\boldsymbol{g}_t - \boldsymbol{s}_t^{i,n})_1^2 & \mathbf{0} \\ & \ddots \\ \mathbf{0} & (\boldsymbol{g}_t - \boldsymbol{s}_t^{i,n})_d^2 \end{pmatrix}$$

Together with $\boldsymbol{\epsilon} \sim \mathbf{R}_3 = \mathcal{N}(0, \boldsymbol{\Sigma}_\epsilon)$, the implicit proposal distribution behind an APSO iteration is $\mathbf{R} = \mathbf{R}_1 * \mathbf{R}_2 * \mathbf{R}_3$[2] with a $\mathbf{s}_t^{i,n}$ translation. Here $*$ stands for the convolution operator.

In this way, the PSO iterations can naturally take the current observation $\mathbf{o}_t$ into consideration, since $\{\mathbf{p}_t^i\}_{i=1}^N$ and $\mathbf{g}_t$ are updated to their observations. Therefore, with coarse importance sampling from the state transition distribution $p(\mathbf{s}_t|\mathbf{p}_{t-1}^i)$, the hierarchical sampling process can approximate to the optimal sampling from $p(\mathbf{s}_t|\mathbf{s}_{t-1}^i, \mathbf{o}_t)$.

As shown in Fig. 5, when the transition distribution is situated in the tail of the observation likelihood, the particles directly drawn from this distribution do not cover a significant region of the likelihood, and thus the importance weights of most particles are low, leading to unfavorable performance. In contrast, through hierarchial sampling process in our algorithm, the particles are moved towards the region where the likelihood of observation has larger values, and are finally

[2] Since the analytical form of $\boldsymbol{R}$ is not available, we called it latent sampling process.

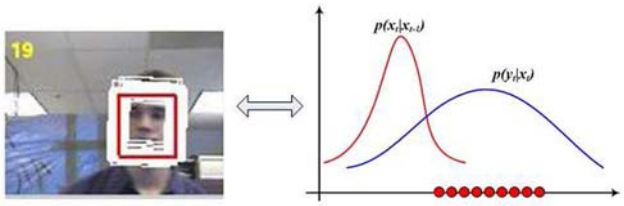


Fig. 5 Sampling results after PSO iterations

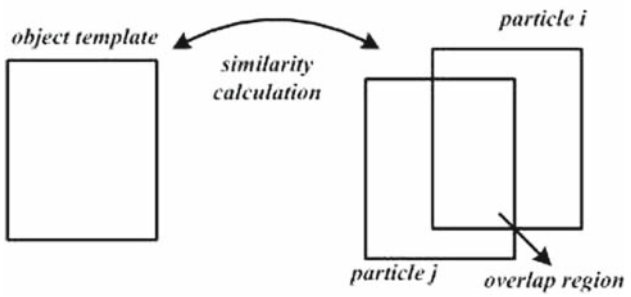


Fig. 6 An illustration example

relocated to the dominant modes of the likelihood, demonstrating the effectiveness of our sampling strategy.

4.3 Differences to Previous Work

Image observations have been utilized for particle guidance in previous work Sullivan and Rittscher (2001), Choo and Fleet (2001). In Sullivan and Rittscher (2001), a momentum based on the differences between consecutive frames is used to determine the search mechanism and the size of the particle set. Motion prediction based on the differences between consecutive frames is unreliable, especially in the LFR video. Besides, the tracking framework in Sullivan and Rittscher (2001) is a particle filter which is not effective in dealing with the abrupt motion in LFR videos. In Choo and Fleet (2001), Choo and Fleet propose a hybrid Monte Carlo technique. Multiple Markov chains are employed to rapidly explore the state space using posterior gradients. The target posterior is approximated by a linear mixture of transition densities and measurement density. However, it is impossible to define a proper transition density, because the object motion is unpredictable in the LFR video, thus leading to an inefficient sampling in the LFR video tracking.

5 Integral Image of Model Parameters

When evaluating the the fitness value of each particle in the APSO iteration process, we need to calculate the parameters of the candidate image region to obtain the similarity measure in Eq. (15). As shown in Fig. 6, the candidate image regions corresponding to the particles may have many overlapped areas, and the image pixels inside the overlapping region will be used repeatedly in calculations. This involves large and unnecessary computational load.

5.1 Integral Image Calculation

The motivation is to avoid the unnecessary computation. The idea is to: first, estimate the maximal coverage region of the particle set, and then calculate the label and position features for each pixel. Finally, construct the integral image of these features. The details are described as follows.

- Estimate the maximal coverage region Ω of the particle set, that is a region which includes all the possible particles. To determine this set, we require prior information about the object motion. However, in LFR video, it is difficult to estimate the abrupt motion. In our work, we introduce a variable $\boldsymbol{\kappa}_t^{max}$ which is the absolute value of the maximum velocity in a second up to time t. The maximal coverage region Ω for time $t+1$ is set to the image area corresponding to $[\mathbf{g}_t - 1.2\boldsymbol{\kappa}_t^{max}, \mathbf{g}_t + 1.2\boldsymbol{\kappa}_t^{max}]$, where $\mathbf{g}_t$ is the tracking result of the object state at time t. Here, the maximal coverage region is heuristically selected by utilizing the motion information in the previous tracking process, and thus provides a reasonable bound on the movement of the particles and a certain capability to take account of their acceleration.
- Label each pixel in Ω using the fast assignment algorithm in Table 2, and then construct a 5D feature for each pixel $\mathbf{q}_i = \{\delta(L(i)-l), (\mathbf{p}_i)_x, (\mathbf{p}_i)_y, (\mathbf{p}_i)_x^2, (\mathbf{p}_i)_y^2\}$, where $\delta(L(i)-l)$ is the Kronecker function such that $\delta(L(i)-l)=1$ if $L(i)=l$ and $\delta(L(i)-l)=0$ otherwise. $(\mathbf{p}_i)_x$, $(\mathbf{p}_i)_y$ are, respectively, the x and y coordinate values of the pixel i.
- Construct the integral image of the 5D feature for each color mode. For example, given a position $\mathbf{p}_i$, the corresponding value of the integral image for the lth color mode is

$$H_i^l = \sum_{\mathbf{p}_k \in \Omega, \mathbf{p}_k \leq \mathbf{p}_i, L(k)=l} \mathbf{q}_k$$

- Calculate the parameters of the candidate image region. Suppose the corresponding image region of a given particle is D, as shown in in Fig. 7(b). Similar to Viola and Jones (2004), the integral image of this region for the lth color mode can be easily obtained by four table lookup operations $H_4^l + H_1^l - H_2^l - H_3^l$. As a result, the parameters of this particle can be obtained from Eqs. (8), (9) and (10).

A similar idea is used in previous work Wang et al. (2007). Our method is different from Wang et al. (2007) in that: (1) In

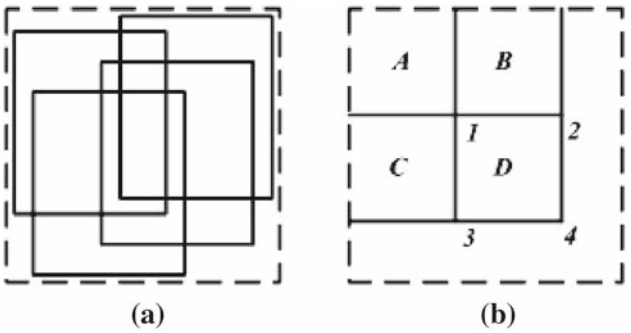


Fig. 7 **a** *Solid line*: Particle set, *dashed line*: the coverage region. **b** An illustration of calculation the *rectangle* integral image

Wang et al. (2007), the particle filtering framework is used and thus it is easy to obtain the coverage region Ω of particle set after the particle generation process. The particles change in the iterations of particle swarm optimization and thus it is more difficult to estimate the coverage region Ω of all possible particles. (2) Pixels inside Ω are labeled by a nearest neighbor classifier in Wang et al. (2007). In our work, labelling is realized by the dominant-set fast assignment algorithm.

5.2 Computational Complexity Analysis

In summary, the features of each pixel need to be calculated only once even though the APSO algorithm is iterative. The complexity is analyzed as follows: In the APSO based searching process, there are two parameters which affect the computational complexity: the number of particles N and the iteration number M. Let Y be the computational complexity for evaluating one particle without the integral image, let y be the computational complexity for evaluating one particle with the integral image, and let C be the computational complexity for building the integral image. The total computational complexity without the integral image is NMY. While with the integral image built in this section, the computational complexity is $NMy + C$. In more detail, y includes the cost of a set of table lookup operations and the direct computation in Eq. (15), and satisfies $y \ll Y$. The value of C depends on the experimental data, and we have $C/Y < 10$ in our experiments. Assuming $C/Y = 10$ and ignoring y, the integral image strategy can achieve $\frac{NMY}{NMy+C} = \frac{NM}{10}$ times speed up. The relationships among Y, y, C are validated in our experiments.

6 Experimental Results

In this section, we first carry out two experiments with synthetic data to analyze and validate the claimed advantages of the bin-ratio based distance measure and the annealed particle swarm optimization based searching method. Then we test the proposed tracking system in the following aspects on real data: different object representations, different similarity measures, different searching methods, different frame rates and average running time. All the experiments are conducted with Matlab on a platform with Pentium IV 3.2GHz CPU and 512M memory. The initial object positions are manually labeled.

6.1 Synthetic Analysis of Bin-Ratio Based Distance Measure

To validate the claimed advantages of the bin-ratio based distance measure, we compare it with the histogram intersection measure (Swain and Ballard 1991) in the presence of illumination changes and image clutters.

6.1.1 Robustness to Illumination Changes

In this part, we assume that the target region is represented by a 20×20 patch in which the pixel values are drawn from a uniform distribution over $\{0, 1, \ldots, 255\}$. The corresponding histogram $\mathbf{u}_1$ is obtained by uniformly grouping the pixels into 16 bins.[3] When the illumination changes, we assume that the intensities of a ratio $\varsigma \in [0, 100\,\%]$ of pixels within the patch change by a factor m in the range $[1/5, 5]$ (the intensities of pixels are set to 255 if their values are bigger than 255 after illumination changes), and the corresponding histogram is $\mathbf{u}_2$. For comparison, we generate another histogram $\mathbf{u}_3$ in a similar way to $\mathbf{u}_1$, for reference. For the bin-ratio based distance, since it is a distance measure, $\Psi_c(\mathbf{u}_1, \mathbf{u}_2) < \Psi_c(\mathbf{u}_1, \mathbf{u}_3)$ indicates the robustness to illumination changes. While the histogram intersection distance $Q(\cdot, \cdot)$ is a similarity measure, $Q(\mathbf{u}_1, \mathbf{u}_2) > Q(\mathbf{u}_1, \mathbf{u}_3)$ shows its robustness to illumination changes. We repeat this process for 10,000 times and calculate the frequency that these inequalities are correct for various m and ς. The experimental results are reported in Table 3.

The results in Table 3 are analyzed in two aspects as follows.

1. Let us first focus on ς. Both measures have poor performance when $\varsigma > 50\,\%$. The reason is that: when the intensities of more than 50 % pixels increase or decrease by the same factor (simulating for the illumination changes), and then these pixel values shift to the bins of hight intensity value or low intensity value. As a result, the corresponding histogram $\mathbf{u}_2$ is very different from the uniform histogram $\mathbf{u}_1$. In addition, $\mathbf{u}_1$ and $\mathbf{u}_3$ are both generated in the same way, and they are all nearly

[3] Since the patch is generated by a uniform distribution, the dominant set clustering can not achieve better performance than uniform binning.

Table 3 The accuracies of different measures with different values of ς and m

ς	Method	$m=\frac{1}{5}$	$m=\frac{1}{4}$	$m=\frac{1}{3}$	$m=\frac{1}{2}$	$m=2$	$m=3$	$m=4$	$m=5$
$\varsigma=10\,\%$	$\Psi_c(\cdot,\cdot)$	0.982	0.988	0.987	0.984	0.983	0.983	0.977	0.968
	$Q(\cdot,\cdot)$	**0.996**	**0.999**	**0.999**	**0.999**	**0.998**	**0.994**	**0.99**	**0.983**
$\varsigma=20\,\%$	$\Psi_c(\cdot,\cdot)$	**0.844**	**0.887**	**0.956**	**0.967**	**0.88**	**0.758**	**0.633**	**0.508**
	$Q(\cdot,\cdot)$	0.675	0.801	0.898	0.957	0.817	0.547	0.353	0.256
$\varsigma=30\,\%$	$\Psi_c(\cdot,\cdot)$	**0.337**	**0.457**	**0.656**	**0.817**	**0.549**	**0.202**	**0.106**	**0.062**
	$Q(\cdot,\cdot)$	0.081	0.15	0.402	0.728	0.318	0.05	0.017	0.005
$\varsigma=40\,\%$	$\Psi_c(\cdot,\cdot)$	**0.034**	**0.07**	**0.207**	**0.504**	**0.181**	**0.025**	**0.007**	**0.002**
	$Q(\cdot,\cdot)$	0.002	0.008	0.05	0.334	0.06	0	0	0
$\varsigma=50\,\%$	$\Psi_c(\cdot,\cdot)$	**0.001**	**0.005**	**0.022**	**0.155**	**0.034**	**0.001**	0	0
	$Q(\cdot,\cdot)$	0	0	0.002	0.089	0.007	0	0	0
$\varsigma=60\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	**0.023**	**0.003**	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0.016	0	0	0	0
$\varsigma=70\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0
$\varsigma=80\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0
$\varsigma=90\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0

Better performance values are given in bold

uniform distributions. Therefore, finding a correct match among $\mathbf{u}_1$, $\mathbf{u}_2$ and $\mathbf{u}_3$ is really a challenging task. If $\mathbf{u}_3$ is generated from a Gaussian distributed patch, then the performance of two measures will significantly improve and $\Psi_c(\cdot,\cdot)$ still outperforms $Q(\cdot,\cdot)$ in most cases.

2. $\Psi_c(\cdot,\cdot)$ performs better than $Q(\cdot,\cdot)$ in most cases except $\varsigma=10\,\%$. In this case, only 10 % of pixels are affected by illumination changes, and the bins of the histogram are influenced only slightly after normalization. Therefore the histogram intersection distance $Q(\cdot,\cdot)$ changes only slightly. However, when $\varsigma>20\,\%$, more pixels are contaminated which cause the imbalance between different bins and the bin-ratios can deal the imbalance more better.

6.1.2 Robustness to Image Clutter

In this part, we assume the target is randomly corrupted by image clutter, e.g. background noise and partial occlusion. As before, $\mathbf{u}_1$ represents the histogram of the target. The image clutter is simulated as follows. We randomly select a ratio $\varsigma\in[0,100\,\%]$ pixels within the patch and replace the intensity values of these pixels by a certain value $m*255$. The resulting histogram $\mathbf{u}_2$ represents the target under image clutter.

The parameters in this experiment are set as follows: $\varsigma\in[0,100\,\%]$ is the proportion of bins influenced by the image clutter, and m is sampled uniformly from [0, 1]. In this experiment, $\mathbf{u}_3$ is generated from a Gaussian distributed patch with mean 128 and standard deviation 32.[4] The experiment is run for 10,000 times, and the proportion of distance comparisons that correctly match $\mathbf{u}_1$ and $\mathbf{u}_2$ is recorded. The experimental results are reported in Table 4.

The results in Table 4 are analyzed as follows. First, $\Psi_c(\cdot,\cdot)$ outperforms $Q(\cdot,\cdot)$ in all cases. This shows that the bin-ratio used for evaluation is more robust to the image clutter. Second, we find that both $\Psi_c(\cdot,\cdot)$ and $Q(\cdot,\cdot)$ have a poor performance when $\varsigma>50\,\%$. This reasonable because more than 50 % pixels are background or occluded which causes incorrect matching.

6.2 Synthetic Analysis of Annealed Particle Swarm Optimization

In this part, APSO is tested on a popular non-linear state estimation problem, which is described as a benchmark in many papers (Merwe et al. 2000). Consider the following nonlinear state transition model given by

$$s_t=1+\sin(\varpi\pi(t-1))+\phi_1 s_{t-1}+\gamma_{t-1},\quad s_t\in\mathbb{R} \tag{29}$$

where γ_{t-1} is a Gamma $\mathcal{G}a(3,2)$ random variable modeling the process noise, and $\varpi=4e-2$ and $\phi_1=0.5$ are scalar parameters. A non-stationary observation model is as follows

[4] Pixels are replaced to simulate image clutter which is more challenging than scaling by a factor in case of illumination changes. For this reason the entries of $\mathbf{u}_3$ are not uniform in size. For more detail, please refer to analysis in Sect. 6.1.1.

Table 4 The accuracies of different measures with different values of ς and m

ς	Method	$m=0$	$m=\frac{1}{7}$	$m=\frac{2}{7}$	$m=\frac{3}{7}$	$m=\frac{4}{7}$	$m=\frac{5}{7}$	$m=\frac{6}{7}$	$m=1$
ς =10 %	$\Psi_c(\cdot,\cdot)$	1	1	1	1	1	1	1	1
	$Q(\cdot,\cdot)$	1	1	1	1	1	1	1	1
$\varsigma=20\,\%$	$\Psi_c(\cdot,\cdot)$	1	1	1	1	1	1	1	1
	$Q(\cdot,\cdot)$	1	1	1	1	1	1	1	1
$\varsigma=30\,\%$	$\Psi_c(\cdot,\cdot)$	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	$Q(\cdot,\cdot)$	0.936	0.941	0.946	0.924	0.937	0.945	0.928	0.941
$\varsigma=40\,\%$	$\Psi_c(\cdot,\cdot)$	**0.999**	**0.999**	**1**	**1**	**1**	**1**	**1**	**1**
	$Q(\cdot,\cdot)$	0.003	0.001	0.005	0	0.001	0.003	0.002	0.001
$\varsigma=50\,\%$	$\Psi_c(\cdot,\cdot)$	**0.387**	**0.387**	**0.372**	**0.372**	**0.381**	**0.358**	**0.398**	**0.326**
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0
$\varsigma=60\,\%$	$\Psi_c(\cdot,\cdot)$	**0.001**	**0.001**	**0.001**	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0
$\varsigma=70\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0
$\varsigma=80\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0
$\varsigma=90\,\%$	$\Psi_c(\cdot,\cdot)$	0	0	0	0	0	0	0	0
	$Q(\cdot,\cdot)$	0	0	0	0	0	0	0	0

$$o_t = \begin{cases} \phi_2 x_t^2 + n_t, & t \le 30 \\ \phi_3 x_t - 2 + n_t, & t > 30 \end{cases} \tag{30}$$

where $\phi_2 = 0.2, \phi_3 = 0.5$, and the observation noise n_t is drawn from a Gaussian distribution $\mathcal{N}(0, 0.00001)$. Given only the noisy observation o_t, several filters are used to estimate the underlying state sequence s_t for $t = 1 \ldots 60$. Here, we compare APSO with conventional particle filter (Arulampalam et al. 2002), extended Kalman based particle filter (Freitas et al. 2000), unscented particle filter (Merwe et al. 2000), auxiliary particle filter (Pitt and Shephard 1999), and conventional particle swarm optimization Kennedy and Eberhart (1995).[5] For each algorithm, a proposal distribution is chosen as shown in Table 5. The parameters in APSO and PSO are set as follows: $\Sigma = 0.8, c = 2$, $\varphi_1 = \varphi_2 = 1, \xi_{max} = 0.8, \xi_{min} = 0.1, T = 20$. Figure 8 gives an illustration of the estimates generated from a single run of the different filters. Compared with other nonlinear filters, APSO is more robust to the outliers, when the observation is severely contaminated by the noise. Since the result of a single run is a random variable, the experiment is repeated 100 times with re-initialization to generate statistical averages. Table 5 summarizes the performance of all the different filters in the following aspects: the means, variances of the mean-square-error (MSE) of the state estimates and the average execution time for one run. It is obvious that the average accuracy of APSO is better than generic PF, EKPF, APF and

[5] We call these filters APSO, PF, EKPF, UPF, APF, PSO respectively for short in the following parts.

Table 5 Experimental results of state estimation

Algorithm	Proposal	MSE mean	MSE var	Time(s)
PF	$p(x_t\|x_{t-1})$	0.42225	0.045589	3.6939
EKPF	$N(\bar{x}_t, \bar{P}_t)$	0.31129	0.015167	13.014
UPF	$N(\bar{x}_t, \bar{P}_t)$	0.06977	0.024894	26.2815
APF	$p(x_t\|x_{t-1})$	0.55196	0.037047	7.1835
PSO	$p(x_t\|x_{t-1})$	0.13019	0.044086	10.2087
APSO	$p(x_t\|x_{t-1})$	0.060502	0.06852	6.8005

comparable to that of UPF. However, the real-time performance of our algorithm is much better than UPF as Table 5 shows. Meanwhile, we can see that APSO can achieve a much faster convergence rate than conventional PSO. This is because the velocity part employed in Eq. (19) carries little information, while the annealing part in APSO iterations enhances the diversity of the particle set and its adaptive effect enables a fast convergence rate. In summary, the total performance of APSO prevails over that of other nonlinear filters.

In order to further evaluate the sampling effectiveness of the hierarchical sampling process in APSO, we consider a special case of the state space model as follows:

$$s_t = f(s_{t-1}) + \gamma_{t-1}, \quad \gamma_{t-1} \sim \mathcal{N}(0, 10) \tag{31}$$

$$o_t = \frac{s_t}{20} + n_t, \quad n_t \sim \mathcal{N}(0, 1) \tag{32}$$

where $f(s_{t-1}) = \frac{s_{t-1}}{2} + \frac{25 s_{t-1}}{1+s_{t-1}^2} + 8\cos(1.2(t-1))$. As proved in Doucet et al. (2000), when applying Bayesian filtering

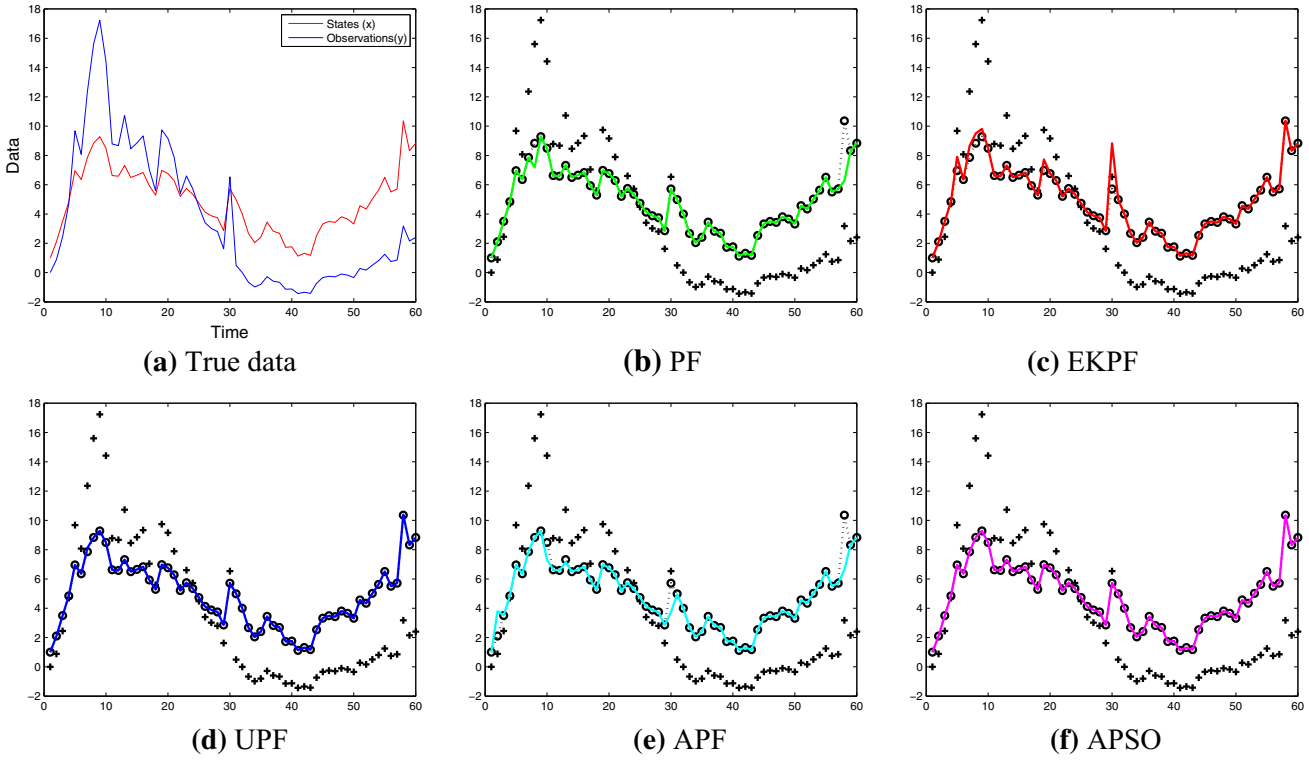


Fig. 8 An illustration of a single run of different filters. **a** True data. **b** PF. **c** EKPF. **d** UPF. **e** APF. **f** APSO

methods to the above nonlinear Gaussian filtering problem, there is an analytic form of the optimal proposal distribution, which is formulated as follows.

$$p(s_t|s_{t-1}, o_t) = \mathcal{N}\left(\frac{40}{41}f(s_{t-1}) + \frac{20}{41}o_t, \frac{400}{41}\right) \tag{33}$$

Based on the optimal proposal distribution, we conduct an experiment to test the sampling effectiveness of the above mentioned nonlinear filters. In each nonlinear filter, after the importance sampling process, we obtain a set of samples $\{s_t^i\}_{i=1}^N$ ($N = 100$), and the effectiveness of the sampling is evaluated by calculating the log-likelihood of the samples on the optimal proposal distribution in the following way.

$$E(\{s_t^i\}_{i=1}^N) = \log\left(\prod_{i=1}^{N}\mathcal{N}(s_t^i, \frac{40}{41}f(s_{t-1}) + \frac{20}{41}o_t, \frac{400}{41})\right) \tag{34}$$

where $\mathcal{N}(s_t^i, \frac{40}{41}f(s_{t-1}) + \frac{20}{41}o_t, \frac{400}{41})$ is the value of s_t^i on the proposal distribution in Eq. (33). If all the sample points are located on the mode of this distribution, the value of $E(\cdot)$ is -89.4. Values close to -89.9 demonstrates that the samples are close to the mode of the proposal distribution. The average value of $E(\cdot)$ for each nonlinear filter is shown in Table 6,

Table 6 Sampling effectiveness of all the nonlinear filters

	PF	EKPF	UPF	APF	APSO
$\bar{E}(\cdot)$	-206	-150.6	-122.6	-189.7	-112.8

which shows that APSO is the most effective in sampling among all the nonlinear filters.

6.3 Tracking Applications

6.3.1 Description of Testing Videos

The proposed tracking system is tested with 9 video sequences and their original frame rate is 15. Most of them are from public datasets, and have public groundtruth. Detailed information on the video sequences, including the frame rate number, derivation, total length and whether groundtruth is available, are shown in Table 7. For sequence 6 and sequence 8, the object in these two sequences has large motion, so they are not down sampled in the frame rate.

6.3.2 Different Object Representations

To show the influence of the object representation on the tracking performance, we conduct experimental comparisons of the dominant color-spatial based representation,

Table 7 Dataset description

Data	Frame rate	Derivation	Length	Groundtruth
Sequence 1	5	http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm	210	Yes
Sequence 2	3	http://homepages.inf.ed.ac.uk/rbf/CAVIAR/	310	Yes
Sequence 3	3	http://whluo.net/category/code_software/	238	No
Sequence 4	5	http://staff.science.uva.nl/~smeulder/research.html	392	No
Sequence 5	3	http://www.cs.technion.ac.il/~amita/fragtrack/fragtrack.htm	250	Yes
Sequence 6	15	http://vision.stanford.edu/~birch/headtracker/	31	No
Sequence 7	1	http://homepages.inf.ed.ac.uk/rbf/CAVIAR/	900	Yes
Sequence 8	15	http://cv.snu.ac.kr/research/~vtd/	71	Yes
Sequence 9	1–5	http://homepages.inf.ed.ac.uk/rbf/CAVIAR/	200	Yes

(a) color histogram based representation

(b) 4x4 block color histogram based representation

(c) dominant color-spatial based representation

Fig. 9 Tracking performances of sequence 1 for frame #1,34,70,100,127,154,205. **a** Color histogram based representation. **b** 4×4 block color histogram based representation. **c** Dominant color-spatial based representation

the color histogram based representation, 4×4 block color histogram based representation, the subspace based representation (Ross et al. 2008), and the region covariance based representation (Hu et al. 2012).[6]

As shown in Fig. 9, for the color histogram based object representation, the tracking window starts to move away from the object position at frame 70 when the background is cluttered and the illumination changes. The track is completely lost when the object is partially occluded by the car. For the 4×4 block color histogram based representation, it deals with the background clutter and illumination changes better than the color histogram based object representation, but the track still fails under partial occlusion. The reason is twofold: (1) the color histogram is sensitive to background distractions

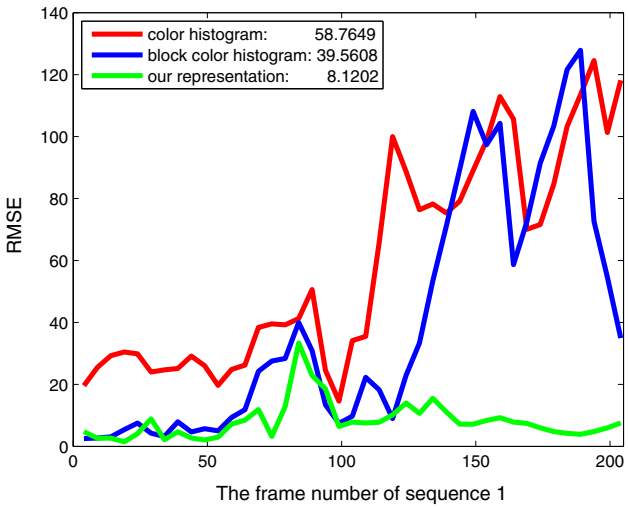


Fig. 10 The RMSE curve of tracking results of sequence 1

[6] The original source code can be found at http://whluo.net/code-for-tracker-based-on-riemannian-subspace-learning/.

Table 8 Quantitative results (RMSE: root mean square error, STR: successful tracking rate)

	Method	Average RMSE	STR (%)
Sequence 1	Color histogram based representation	58.7649	31.7
	4×4 block color histogram based representation	39.5608	58.6
	Dominant color-spatial based representation	8.1202	100
Sequence 5	Histogram intersection based measure	18.0977	74
	Mutual information based measure	15.5154	98
	Bin-ratio based measure	6.1326	100
Sequence 7	Particle filter	19.5267	81.7
	Annealed particle filter	6.0856	100
	Annealed PSO	4.9474	100
Sequence 8	MIL tracker	49.4837	15.5
	SSOB tracker	15.2096	84.5
	Tuzel's tracker	14.054	97.1
	Our tracker	8.3869	100

and illumination changes; (2) the color histogram and the 4×4 block color histogram are not robust enough to the partial occlusion. Because the relative positions of the object pixels are ignored in the color histogram, even divided into 4×4 block, it contains very little information about the spatial layout of the dominant colors. In contrast, the dominant color-spatial based representation overcomes the above two limitations by using the dominant color modes and extracting their spatial information. As a result, the object is tracked successfully throughout the video sequence As shown in Fig. 10 and Table 8, the tracking accuracy is also quantitatively analyzed by calculating the root mean square error (RMSE) between the tracking results and the groundtruth and successful tracking rate (STR). When the overlap area of the tracking results and the groundtruth is less than 30 % of the area of the groundtruth, we consider this is a failure to track. STR is successful tracking rate throughout the whole sequence. We can see the average RMSE for the color histogram representation is 58.7649 pixel distance, the average RMSE for the 4×4 block color histogram representation is 39.5068 pixel distance and the average RMSE for our representation is 8.1202 pixel distance. Correspondingly, the STR for the three different appearance models are 31.7, 58.6, 100 %.

The comparison between dominant color-spatial based representation and subspace based representation is illustrated in Fig. 11, from which we can see that the subspace model drifts away from the object when the object turns around at frame 251 and loses the object completely at frame 341 when the object bows. The reason is the large appearance changes and variations in the object's pose caused by the low frame rate. Thus the correspondence of pixels between the object and the subspace is not accurate, leading to the tracking failure. The dominant color-spatial based representation finds a good balance between adaptability and robustness, and thus can adapt well to these appearance changes. As a result, it can successfully track the object throughout the sequence.

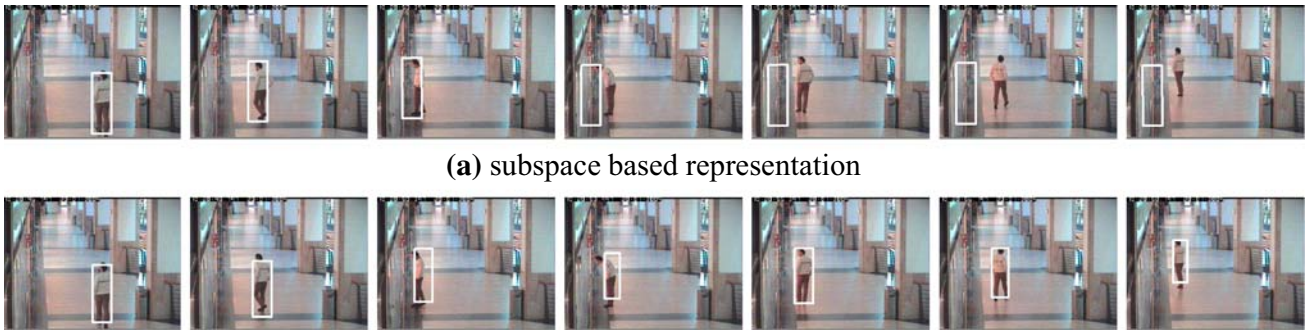
Furthermore, the comparison between dominant color-spatial based representation and the region covariance based representation is shown in Fig. 12. The tracker based on the region covariance features fails when the man bows down to reach the ground at frame 176. The reason is that the region covariance feature is calculated based on the coordinates, intensity values and intensity derivatives of the image pixels. When the man suddenly bows down in the LFR video, there is a sudden change in the covariance feature. The appearance model employed in our work simply captures the dominant colors. This ensures a larger tolerance to the pose changes, and thus a better tracking performance.

6.3.3 Different Measures

In this part, an experimental comparison among the bin-ratio based measure, the histogram intersection based measure Swain and Ballard (1991) and the mutual information based measure[7] is carried out.

As shown in Fig. 13, sequence 4 undergoes three kinds of illumination changes. The top row and the middle row show the similar tracking results using the histogram intersection based measure and the mutual information based measure. It is clear that the trackers gradually drift away from the true position, because the match between the object and the template is not accurate after the illumination changes. In contrast, the proposed measure is not directly based on the number of pixels contributing to a particular color bin. The use of bin-ratios reduces the influence of the illumination changes. Therefore, the bin-ratio based measure achieves sat-

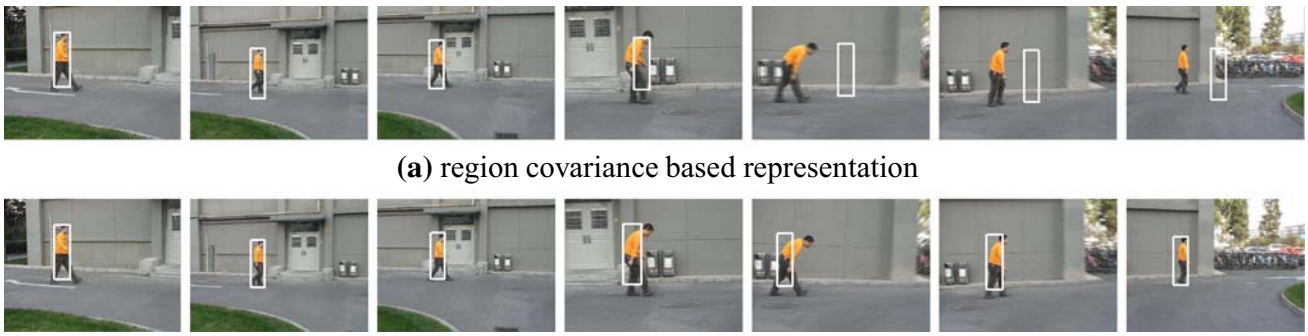
[7] http://en.wikipedia.org/wiki/Mutual_information



(a) subspace based representation

(b) dominant color-spatial based representation

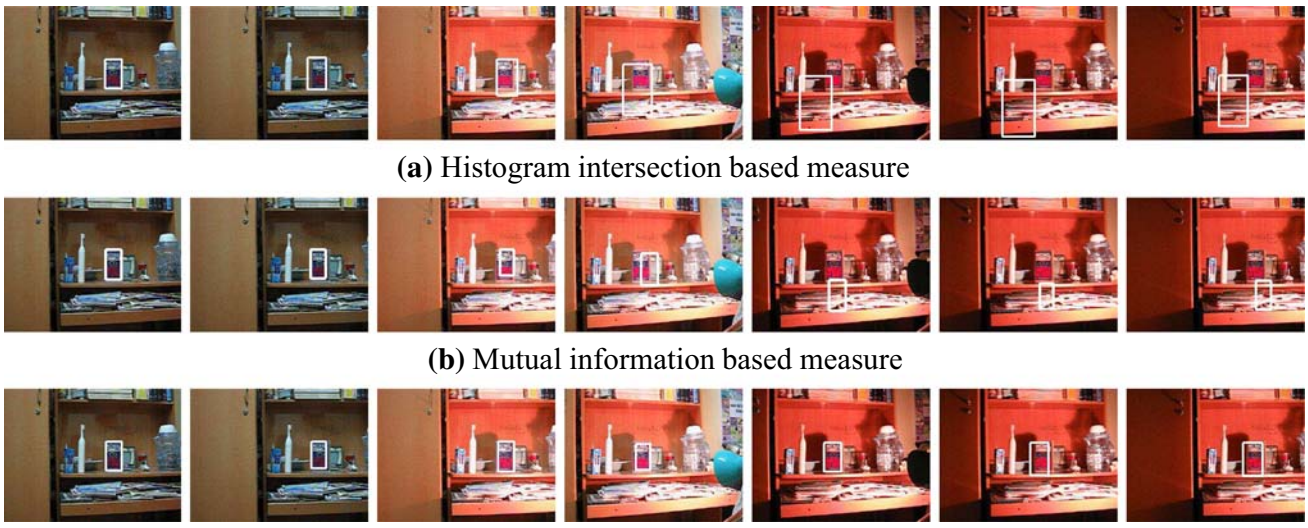
Fig. 11 Tracking performances of sequence 2 for frame #161,201,251,341,376,396,446. **a** Subspace based representation. **b** Dominant color-spatial based representation



(a) region covariance based representation

(b) dominant color-spatial based representation

Fig. 12 Tracking performances of sequence 3 for frame #1,46,76,126,176,191,221. **a** Region covariance based representation. **b** Dominant color-spatial based representation

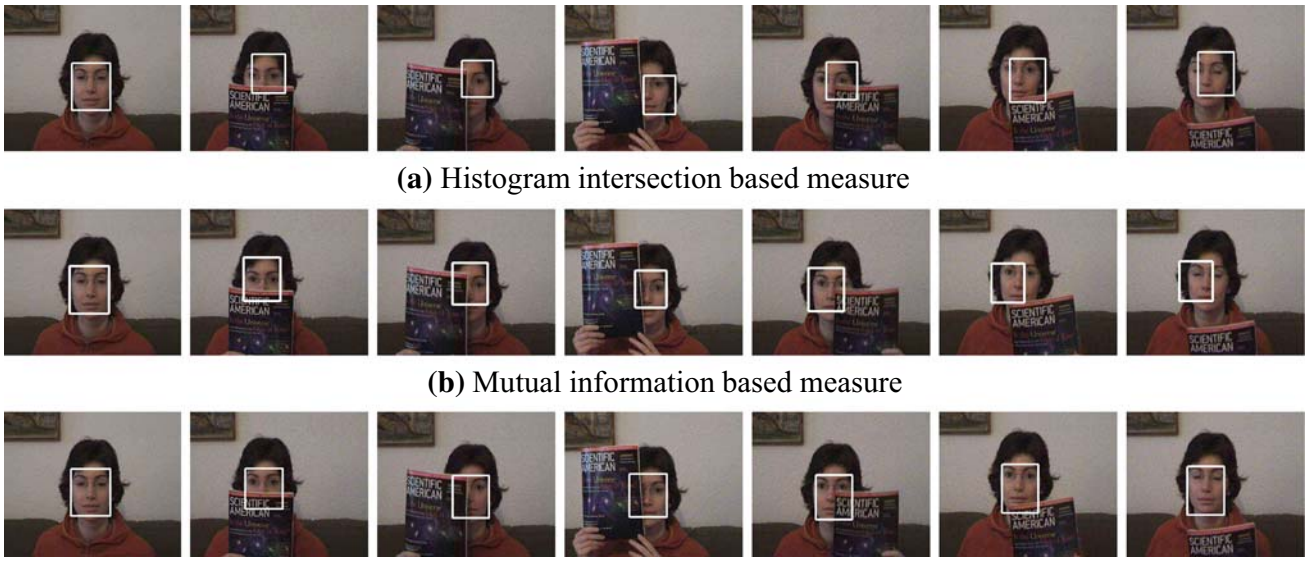


(a) Histogram intersection based measure

(b) Mutual information based measure

(c) Bin-ratio based measure

Fig. 13 Tracking performances of sequence 4 for frame #262,280,289,340,355,370,391. **a** Histogram intersection based measure. **b** Mutual information based measure. **c** Bin-ratio based measure



(a) Histogram intersection based measure

(b) Mutual information based measure

(c) Bin-ratio based similarity

Fig. 14 Tracking performances of sequence 5 for frame #2,37,87,122,172,237,242. **a** Histogram intersection based measure. **b** Mutual information based measure. **c** Bin-ratio based similarity

isfactory tracking results in sequence 3 (see the bottom row of Fig. 13).

We also test these three measures on a video sequence with partial occlusion. As shown in Fig. 14, for the histogram intersection based measure and the mutual information based measure, the occluded pixels are discarded, since these pixels can not provide positive information for matching, leading to inaccurate localization. While in the proposed similarity measure in Eq. (13), the similarity between corresponding color bins is a summation of the ratios of all the color bins. This alleviates the influence of the occlusion, and achieves a more accurate localization. In addition, the RMSE between the tracking results and the groundtruth is calculated and presented in Fig. 15 and the STR is presented in Table 8. From these quantitative results, we find that bin-ratio measure achieves the best performance.

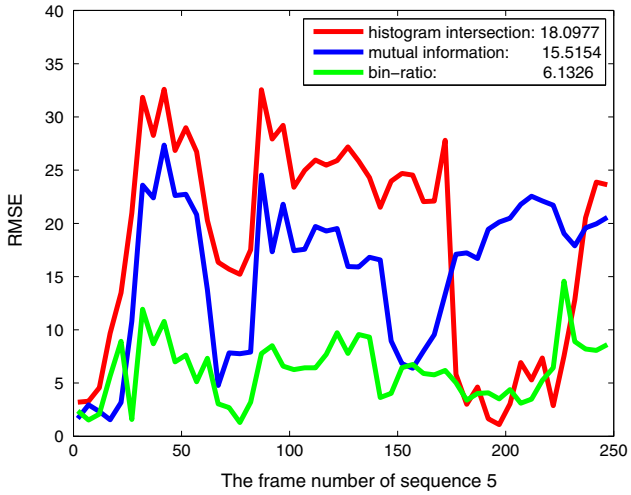


Fig. 15 The RMSE curve of tracking results of sequence 5

6.3.4 Different Searching Methods

To provide experimental validation of the analysis in Sect. 4, we compare the APSO based searching method with the particle filtering based searching method and the annealed particle filtering based searching method Deutscher et al. (2000). To make the comparison more convincing, the annealing is only applied to the diffusion variance. The parameters employed in the three searching methods are set as follows: (1) For sequence 6, the number of particles used in the particle filter is 600, and the number of particles used in APSO and annealed particle filter is 60 with 10 annealed iterations. (2) For sequence 7, the number of particles used in the particle filter is 200, and the number of particles used in APSO and annealed particle filter is 20 with 10 annealed iterations. All the searching methods are initialized with the same diffusion variances that correspond to the searching region in the state space.

As shown in Fig. 16, the particle filtering based searching method fails to track the rapid motion of the object, because it can not catch the rapid motion of the object. More particles and an enlargement of the searching region may improve its performance, but this will lead to a high computational cost for searching and involves more noise. The annealed parti-

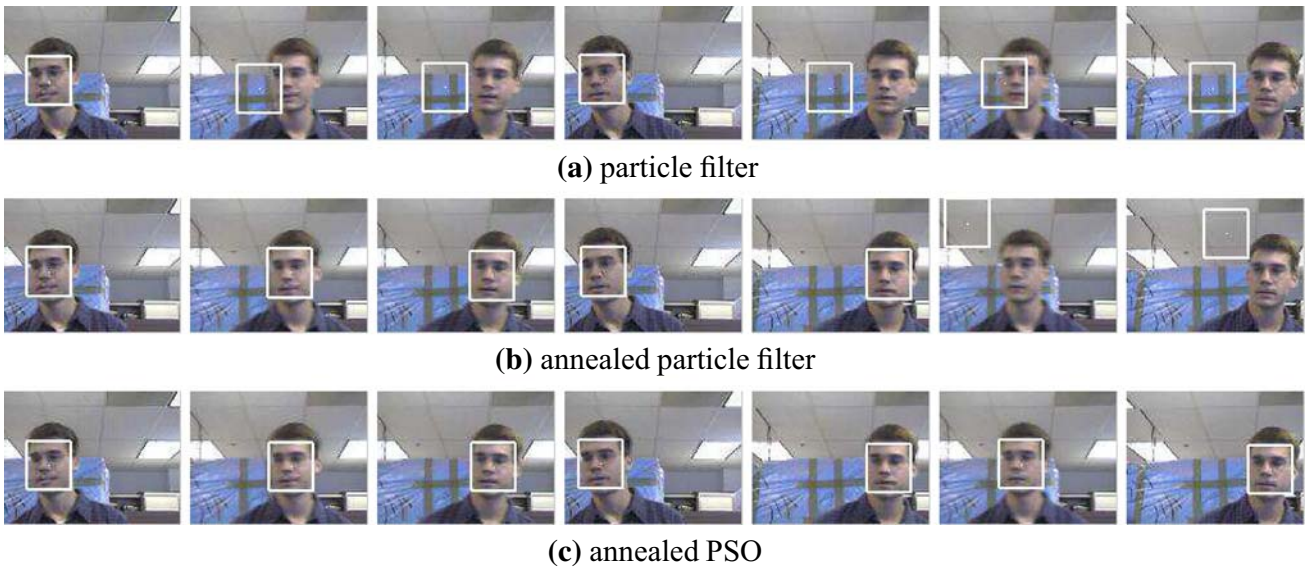


Fig. 16 Tracking performances of sequence 6 for frame #1,6,11,16,22,28,31. **a** Particle filter. **b** Annealed particle filter. **c** Annealed PSO

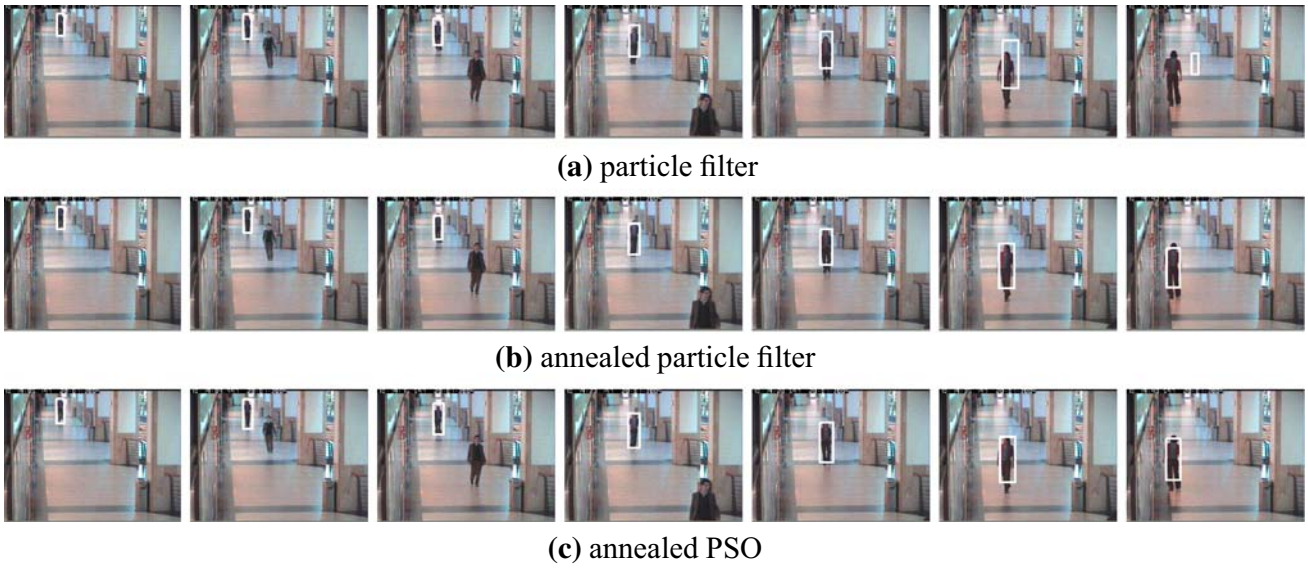


Fig. 17 Tracking performances of sequence 7 for frame #401, 521, 686, 851, 986, 1151, 1286. **a** Particle filter. **b** Annealed particle filter. **c** Annealed PSO

cle filter can achieve a better performance than the particle filter, because the annealed iterations can enlarge the searching space. However, there is no guidance information for the evolution of particles in the annealed particle filter. In the APSO based searching method, the particles evolve according to individual and environmental information in the search space, and thus never lose track of the object even under abrupt motion. Meanwhile, the random perturbations of the particles ensure that the search does not become trapped in local optima and the annealing factor enables a much faster convergence rate.

Figure 17 shows some key frames of the tracking results in sequence 7, from which we can also find that APSO and the annealed particle filter achieve similar tracking results, because the motion generated by LFR in this sequence is not very large. The particle filter fails to track the man when he suddenly turns around, which validates the analysis of the sample impoverishment problem in Sect. 4.1. Furthermore,

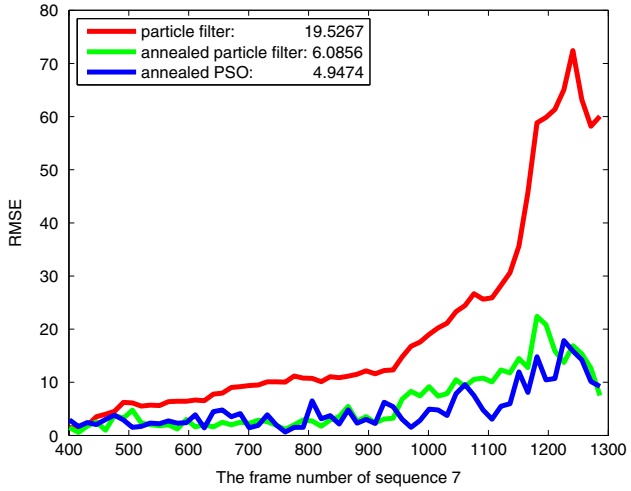


Fig. 18 The RMSE curve of tracking results of sequence 7

we have conducted a quantitative evaluation of these two searching methods, and have a comparison in MSE between the tracking results and the groundtruth. Figure 18 shows the RMSE curve of the tracking results for sequence 7. The average RMSE for the particle filter is 19.5267 pixel distance and the average RMSE for APSO is 4.9474 pixel distance.

6.3.5 Comparison with State-of-the-Art Trackers

To make the experiments more convincing, we conduct a comparison experiment between our algorithm and three state-of- the-art discriminative tracking algorithms (Grabner et al. 2008; Babenko et al. 2011; Tuzel et al. 2008). Specifically, these three competing trackers are referred as semi-supervised on-line boosting (SSOB) (Grabner et al. 2008), multiple instance learning (MIL) (Babenko et al. 2011),[8] and Tuzel's tracker Tuzel et al. (2008). All the algorithms use the same searching region in the tracking process, and a greedy exhaustive search strategy is adopted by SSOB and MIL.

The eighth sequence contains a running buck, with a cluttered background. As shown in Fig. 19, we can see that MIL fails to track the object at frame 12 and can not recover the track in the remaining part of the sequence. The reason is that the Harr feature is not discriminative enough when the appearance of the object changes. The SSOB tracker loses track of the object at frame 24 when some water splashes over the buck and blurs the image. However, the tracker is recovered because SSOB has a detection component. If the detection component is intrigued, it can help relocalize the object. According to the STR results in Table 8, both Tuzel's tracker and our tracker can successfully track the object throughout most of the sequence with the same searching region. We also present quantitative evaluations of these three tracking algorithms. As shown in Fig. 20, the RMSE of the two points (left top and right bottom) in the bounding box is calculated between the tracking results and the groundtruth. From the RMSE curve, we can see that our tracker outperforms the MIL tracker, the SSOB tracker and Tuzel's tracker in accuracy.

[8] The source codes are downloaded from the authors' webpage.

6.3.6 Performance with Different Parameters

The effect of the frame rate and the number of dominant colors on performance are key issues of the proposed tracking system. In this part, we will investigate the the sensitivity of the tracking system to the changes in the frame rate and the number of dominant colors.

(1) *Different Frame Rates* Figure 21 shows the tracking results of sequence 9 with frame rate 1–5. It is not easy to find which frame rate has the best tracking performance from Fig. 21, so we calculate the RMSE between the tracking results and the groundtruth of sequence 9 with different frame rates. As shown in Fig. 22, from left to right, the results are corresponding to frame rate 1, 2, 3, 4, 5 respectively. From Fig. 22, we can see that the average RMSE is 3.3402, 3.1354, 2.8757, 2.9571, 2.8978 for frame rate 1, 2, 3, 4, 5, which means that the tracking performance is similar under different frame rates. The reason is that the PSO based searching framework can handle the abrupt motion well even when the frame rate is decreased to 1.

(2) *Different Number of Dominant Colors* Figure 23 shows the RMSE curves of tracking results on sequence 9 when the number of dominant colors is 6–10. We can see that the tracking performance is similar when the number of dominant colors is 6–8, and these results are better than the results in case of 9 and 10 dominant colors. The reason is that there are about 6–8 dominant colors inside the target region. If we set the dominant colors to 9 or 10, the extra color modes will be noise. The average RMSE in case of 9 and 10 dominant colors are 5.2564 and 5.4917 respectively, which are tolerable in tracking applications. The reason is that: since the dominant set become smaller in the dominant set clustering process, the pixel number within the last two dominant colors are very small which takes a little effect on the proposed similarity measure. In summary, the proposed algorithm is not very sensitive when the number of dominant colors changes.

6.3.7 Average Running Time

To validate the efficiency of our tracking system, the following two aspects of tracking time are analyzed: the comparison of average running time for the proposed tracker with the integral image in Sect. 5 and the proposed tracker without the

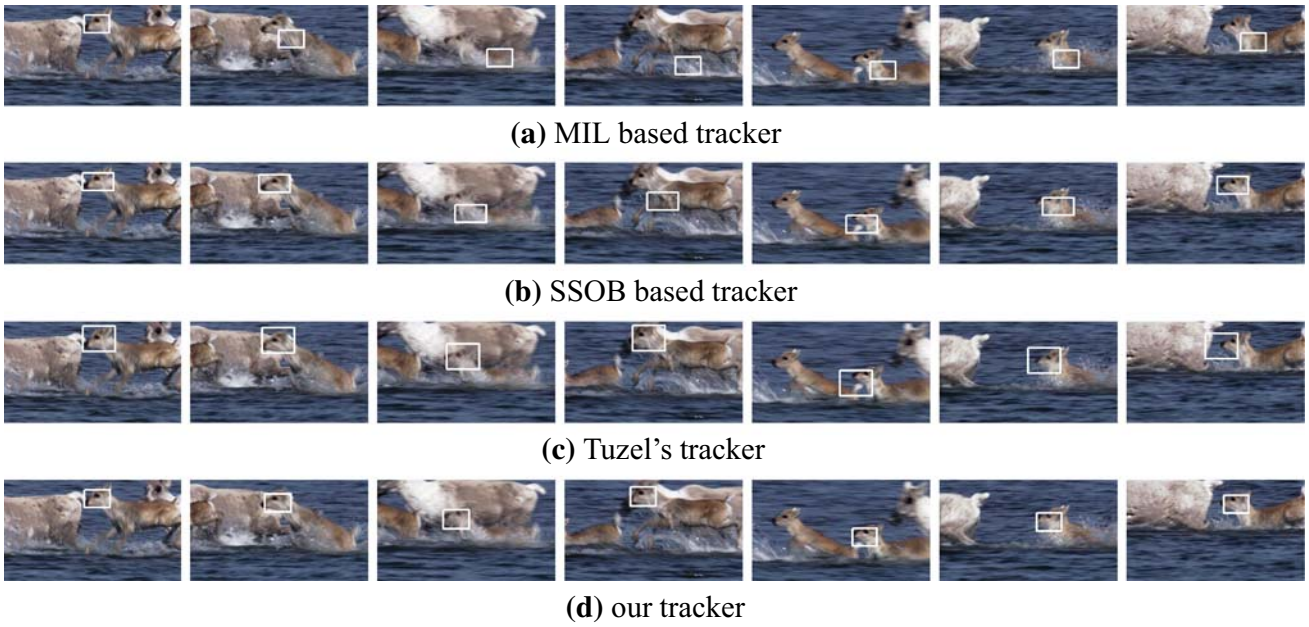


Fig. 19 Tracking performances of sequence 8 for frame #3,12,24,33,40,58,71. **a** MIL based tracker. **b** SSOB based tracker. **c** Tuzel's tracker. **d** Our tracker

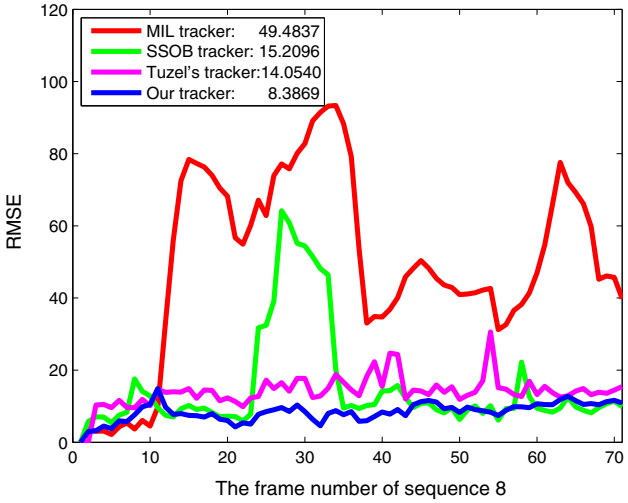


Fig. 20 The RMSE curve of tracking results of sequence 8

integral image in Sect. 5, and also the comparison of average running time under different frame rates. For the first comparison, we take tracking results on sequence 6 as an example. The average running time of the proposed tracker with the integral image is 0.2578 seconds per frame, while the average running time of the proposed tracker without the integral image is 12.65 seconds per frame. The result demonstrates that the use of the integral image can achieve about 50 times speed-up. The reason is analyzed as follows: the number of particles and iterations N, M are set to 60, 10 respectively, and the average C/Y is about 8. Taking the y into consideration, the analysis in Sect. 5.2 is consistent with the above experimental result. Table 9 shows the average running time of sequence 7 under different frame rates. Without any optimization, the average running time gradually increases when the frame rate decreases. The reason is that when the frame rate decreases, the motion of the object between frames is larger, so the maximal coverage region Ω employed in Sect. 5 is larger and thus more time is needed to calculate the integral image of the region. The analysis of the average running time strictly supports the content described in Sect. 5.

Moreover, we will investigate the computational complexity when the number of color modes changes. When the size of the histogram changes from 6 to 10, the corresponding average running times are 0.2197, 0.2284, 0.2355, 0.2432, 0.2498 (see Table 10). The average running time shows only a slow increase. The reason is as follows. There are two major steps in the integral image calculation process: (1) Construction of a 5D feature for each pixel; (2) Calculation of the integral image of the 5D feature for each color mode. The computational complexity of the first step remains the same when the histogram size grows. While both computational complexity and memory grow linearly according to the histogram size. Fortunately, the main computational complexity arises from the constructing of a 5D feature for each pixel, so the average running time varies little when the number of color mode changes.

(a) Frame rate 1: #502, 547, 592, 637, 682, 742, 787

(b) Frame rate 2: #502, 558, 600, 635, 698, 740, 796

(c) Frame rate 3: #502, 567, 592, 622, 667, 722, 797

(d) Frame rate 4: #502, 566, 590, 630, 678, 718, 798

(e) Frame rate 5: #502, 538, 586, 616, 676, 745, 796

Fig. 21 Tracking performances of sequence 9 with frame rate 1–5

6.3.8 Summary

We summarize the reasons why our tracking system achieves satisfactory results in the following cases:

- For changes in illumination: (1) we employ the normalized *rgI* color space , which is more robust to illumination changes than the original *RGB* color space; (2) the proposed similarity measure does not depend directly on the number of pixels in a particular color bin, but on the bin-ratio. These factors reduce the influence of illumination changes.
- For changes in the state of objects (including position, shape, size): (1) the dominant color based appearance model provides global statistical information about the target region, so the model is robust against shape/size changes; (2) the APSO based framework searches the state space associated with each target effectively.
- For the changes in the local background: the spatial layout information of the dominant colors are used in the appearance model to enhance its ability to distinguish the target from the background.

7 Conclusion

This paper has proposed a tracking system for LFR videos. This system includes a new appearance model and a new search method. In the appearance model, the object is represented by the dominant color-spatial modes and a bin-ratio based similarity measure is used for matching. The space of object motions is searched by the annealed PSO based method, and the model parameters are obtained from a look-up table which is constructed using an integral image. Experimental results demonstrate the effectiveness of the tracking system.

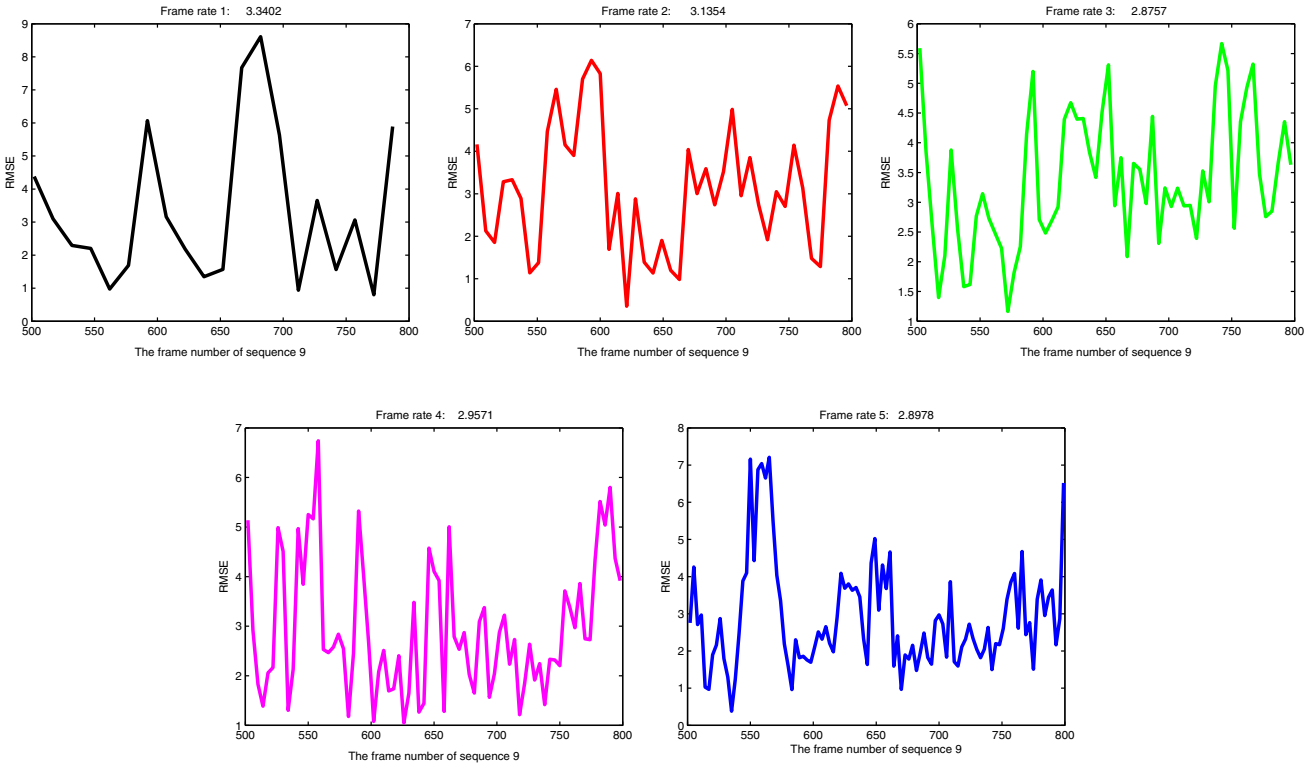


Fig. 22 The RMSE curve of the tracking results with frame rate 1–5

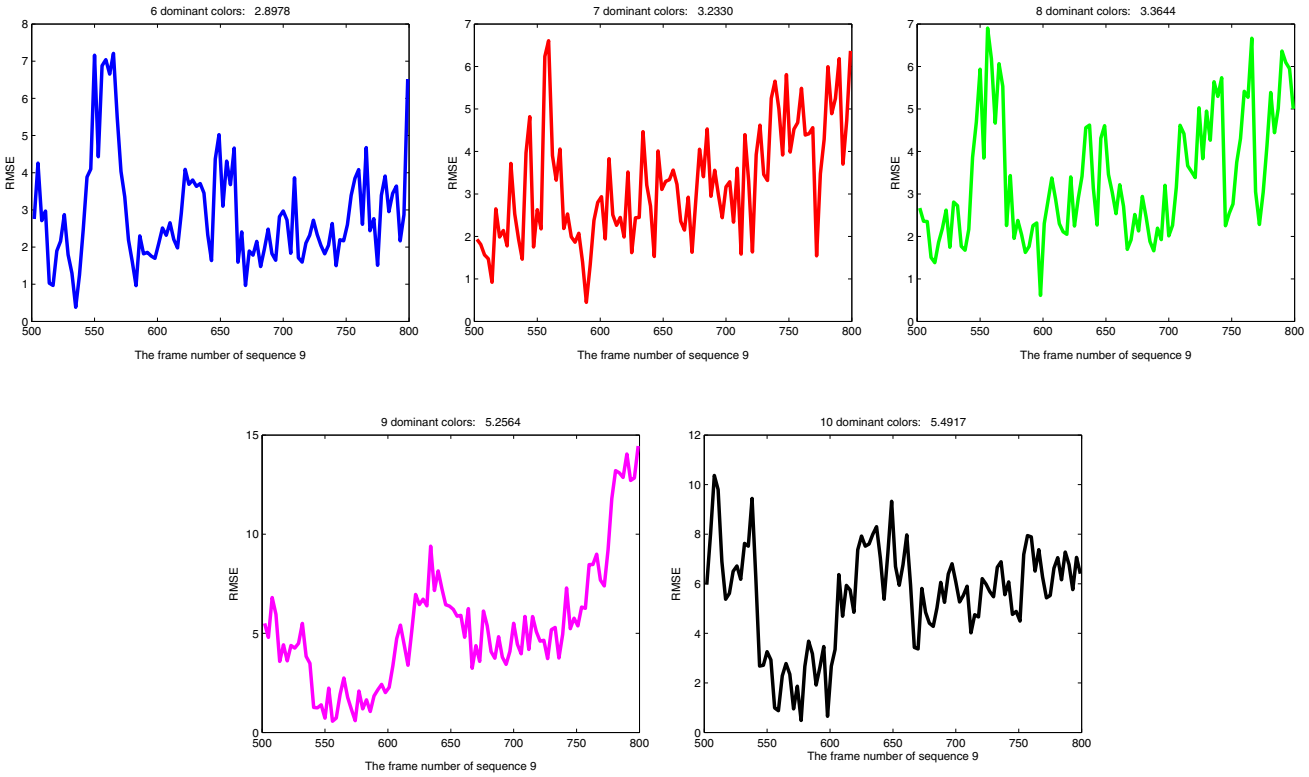


Fig. 23 The RMSE curve of the tracking results with 6–10 dominant colors

Table 9 Average running time under different frame rates

Frame rate number	Average running time (seconds per frame)
5	0.2197
4	0.2218
3	0.2316
2	0.2411
1	0.2695

Table 10 Average running time under different number of color mode

Number of color mode	Average running time (seconds per frame)
6	0.2197
7	0.2284
8	0.2355
9	0.2432
10	0.2498

References

Arulampalam, M., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particles filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, *50*(2), 174–188.

Avidan, S. (2004). Support vector tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(8), 1064–1072.

Avidan, S. (2007). Ensemble tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(2), 261C271.

Babenko, B., Yang, M., & Belongie, S. (2011). Robust object tracking with online multiple instance learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *33*(8), 1619C1632.

Black, M. J., & Jepson, A. D. (2004). EigenTracking: Robust matching and tracking of articulated objects using a view-based representation. *International Journal of Computer Vision*, *26*(1), 63–84.

Bray, M., Meier, E. K., Schraudolph, N. N., & Gool, L. V. (2007). Fast stochastic optimization for articulated structure tracking. *Image and Vision Computing*, *25*(3), 352–364.

Carrano, C. J. (2009). Ultra-scale vehicle tracking in low spatial resolution and low frame-rate overhead video. *Proceedings of SPIE*, *7445*, 744504.

Choo, K., & Fleet, D. (2001). People Tracking Using Hybrid Monte Carlo Filtering, In: *Proceeding of International Conference on Computer Vision*.

Collins, R., Liu, Y., & Leordeanu, M. (2005). Online selection of discriminative tracking features. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(10), 1631C1643.

Comaniciu, D., Ramesh, V., & Meer, P. (2003). Kernel-based object tracking. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, *25*(5), 234–240.

Deutscher, J., Blake, A., & Reid, I. (2000). Articulated body motion capture by annealed particle filtering. In: *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 126–133.

Doucet, A., Godsill, S., & Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statist. Comput.*, *10*(3), 197–208.

Freitas, D., Niranjan, M. A., Gee, A. H., & Doucet, A. (2000). Sequential Monte Carlo methods to train neural network models. *Neural Computation*, *12*(4), 955–993.

Fukunaga, K., & Hostetler, L. (1975). The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, *21*(1), 32–40.

Grabner, H., Grabner, M., & Bischof, H. (2006). Real-time tracking via on-line boosting. In: *Proceedings of British Machine Vision Conference*, vol. 1, pp. 46–56.

Grabner, H., Grabner, M., & Bischof, H. (2008). Semi-supervised on-line boosting for robust tracking. In: *Proceedings of European Conference on Computer Vision*, pp. 234–247.

Hager, G. D., & Belhumeur, P. N. (1998). Efficient region tracking with parametric models of geometry and illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*(10), 1025–1039.

Hu, W., Li, X., Luo, W., Zhang, X., Maybank, S., & Zhang, Z. (2012). Single and multiple object tracking using log-euclidean riemannian subspace and block-division appearance model. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(12), 2420–2440.

Ingber, L. (1993). Simulated annealing: Practice versus theory. *Journal of Mathematical and Computer Modeling*, *18*(11), 29–57.

Isard, M., & Blake, A. (1998). Condensation: Conditional density propagation for visual tracking. *International Journal of Computer Vision*, *29*(1), 5–28.

Jepson, A. D., Fleet, D. J., & El-Maraghi, T. F. (2003). Robust online appearance models for visual tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *25*(10), 1296–1311.

Kass, M., Witkin, A., & Terzopoulos, D. (1988). Snakes: Active contour models. *International Journal of Computer Vision*, *1*(4), 321–332.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In: *Proceeding of International Conference on Neural Networks*, pp. 1942–1948.

Leung, A.P., & Gong, S.G. (2007). Optimizing distribution-based matching by random subsampling. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Levy, A., & Lindenbaum, M. (2000). Sequential Karhunen–Loeve basis extraction and its application to images. *IEEE Transactions on Image Processing*, *9*(8), 1371–1374.

Li, Y., Ai, H., Yamashita, T., Lao, S., & Kawade, M. (2008). Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(10), 1728–1740.

Lim, J., Ross, D., Lin, R. S., & Yang, M. H. (2004). Incremental learning for visual tracking. *In Advances in Neural Information Processing Systems*, pp. 793–800.

Lin, R. S., Ross, D., Lim, J., & Yang, M. H. (2004). Adaptive discriminative generative model and its applications. *In Advances in Neural Information Processing Systems* pp. 801–808

Merwe, R., Doucet, A., Freitas, N., & Wan, E. (2000). The unscented particle filter. *Technical Report CUED/F-INFENG/TR 380*. Cambridge: Cambridge University Engineering Department.

Nummiaro, K., Meier, E., & Gool, L. (2003). An adaptive color-based particle filter. *Image and Vision Computing*, *21*(1), 99–110.

Pavan, M., & Pelillo, M. (2003). A new graph-theoretic approach to clustering and segmentation. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3895–3900.

Pavan, M., & Pelillo, M. (2005). Efficient out-of-sample extension of dominant-set clusters. *Advances in Neural Information Processing Systems*, *17*, 1057–1064.

Pitt, M. K., & Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, *94*(446), 590–599.

Porikli, F. & Tuzel, O. (2006). *Object tracking in low-frame-rate video*. US Patent Application Publication, US20060222205A1.

Porikli, F., Tuzel, O., & Meer, P. (2006). Covariance tracking using model update based on lie algebra. In: *Proceedings of International Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. 728–735.

Porikli, F., & Tuzel, O. (2005). Object tracking in low-frame-rate video. *SPIE Image and Video Communications and Processing*, *5685*, 72–79.

Puzicha, J., Hofmann, T., & Buhmann, J. (1997). Non-parametric similarity measures for unsupervised texture segmentation and image retrieval. In: *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*.

Ross, D., Lim, J., Lin, R., & Yang, M. (2008). Incremental learning for robust visual tracking. *International Journal of Computer Vision*, *77*(1), 125–141.

Saffari, A., Leistner, C., Santner, J., Godec, M., & Bischof, H. (2010). On-line random forests. In: *Proceedings of European Conference on Computer Vision Workshops*, pp. 1393–1400.

Stauffer, C., & Grimson, W. (1999). Adaptive background mixture models for real-time tracking. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Sullivan, J., & Rittscher, J. (2001). Guiding random particles by deterministic search. In: *Proceeding of International Conference on Computer Vision*.

Swain, M., & Ballard, D. (1991). Color indexing. *International Journal of Computer Vision*, *7*(1), 11–32.

Tomasi, C. & Kanade, T. (1991). Detection and tracking of point features. *Technical Report CMU-CS-91-132*, Carnegie Mellon University.

Tuzel, O., Porikli, F., & Meer, P. (2008). Learning on lie groups for invariant detection and tracking. In: *Proceedings of International Conference on Computer Vision and Pattern Recognition*, pp. 1–8.

Viola, P., & Jones, M. (2004). Robust real-time face detection. *International Journal of Computer Vision*, *52*(2), 137–154.

Wang, H., Suter, D., Schindler, K., & Shen, C. (2007). Adaptive object tracking based on an effective appearance filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *29*(9), 1661–1667.

Xie, N., Ling, H., Hu, W., & Zhang, X. (2010). Use bin-ratio information for category and scene classification. In: *Proceeding of IEEE Conference on Computer Vision and Pattern Recognition*.

Zhang, T., Fei, S., Lu, H., & Li, X. (2009). Modified particle filter for object tracking in low frame rate video. In: *Proceeding of IEEE Conference on Decision and Control* (pp. 2552–2557).

Zhou, S. K., Chellappa, R., & Moghaddam, B. (2004). Visual tracking and recognition using appearance-adaptive models in particle filters. *IEEE Transactions on Image Processing*, *13*, 1491–1506.