CamShift Guided Particle Filter for Visual Tracking

Zhaowen Wang , Xiaokang Yang , Yi Xu [*] , Songyu Yu

Institute of Image Communication and Information Processing
Shanghai Jiao Tong University, Shanghai, PRC 200240

Abstract

In this article, a novel algorithm - CamShift guided particle filter (CAMSGPF) - is proposed for tracking object in video sequence. CamShift is incorporated into the probabilistic framework of particle filter as an optimization scheme for proposal distribution. Meanwhile, in the context of particle filter, the scale adaptation of CamShift is improved and the computation complexity is reduced. It is demonstrated through several real tracking tasks that the new method performs better than baseline trackers in both tracking robustness and computational efficiency.

Key words: CamShift guided particle filter, particle filter, CamShift, visual tracking algorithm

Classification codes: 5.013(Bayesian models and belief networks), 8.002(object and target tracking)
Abbreviations: PF: particle filter; SIR: sequential importance resampling; MSEPF: mean shift embedded particle filter; CAMSGPF: CamShift guided particle filter

1 Introduction

Visual object tracking is a fundamental topic in computer vision field and is widely applied to surveillance, robotics, human machine interface, object

[*] *Corresponding author:* Yi Xu, RM 409, No 5, DianYuan Building, Shanghai Jiao Tong University, Dongchuan Road 800, Shanghai, PRC 200240; xuyi@sjtu.edu.cn; Tel: 8621-13661784031; Fax: 8621-34204155

Email addresses: `whereaswill@sjtu.edu.cn` (Zhaowen Wang), `xkyang@sjtu.edu.cn` (Xiaokang Yang), `xuyi@sjtu.edu.cn` (Yi Xu), `syyu@sjtu.edu.cn` (Songyu Yu).

based video coding, etc. However, the task of robust tracking is very challenging regarding fast motion, occlusion, structural deformation, illumination variation, background clutters, real-time restriction, etc.

To handle these problems, many efforts have been paid to devise good tracking algorithms. One promising category is sequential Monte Carlo methods, also known as particle filers (PFs), which recursively estimate target posterior with discrete sample-weight pairs in a dynamic Bayesian framework. The basic idea was introduced by Hammersley & Morton (1954), and developed into various improved versions over the last decade (Carpenter et al., 1999; Gordon et al., 1993; Kanazawa et al., 1995; MacCormick & Blake, 1999). Due to particle filters' non-Gaussian non-linear assumption and multiple hypothesis property, they have been successfully applied to visual tracking (Isard & Blake, 1998; Perez et al., 2002), and show unique merit in cluttered environment. However, the inefficiency in sampling (due to the problem of degeneracy and impoverishment(Arulampalam et al., 2002)) and the huge computational complexity limit the usefulness of particle filter in on-line tracking.

Another popular tracking method is mean shift procedure, which finds the local maximum of probability distribution in the direction of gradient. Comaniciu & Ramesh (2000) gave a strict proof of the convergence of the algorithm, and Comaniciu et al. (2000) proposed a mean shift based tracking method. As a deterministic method, mean shift keeps single hypothesis and is thus computationally efficient. But it may run into trouble when similar objects are presented in background or when occlusion occurs.

Based on the pros and cons of particle filter and mean shift, Shan et al. (2004) proposed a new algorithm, Mean Shift Embedded Particle Filter (MSEPF). In MSEPF, mean shift is performed on each of the particles after they are propagated, so that the particles are "herded" to nearby local modes with large probability. Thus, the problem of degeneracy, where the weights of most particles become negligibly small after a few iterations, is tackled effectively. Moreover, as MSEPF can make better estimation of posterior even with a smaller set of samples, the computation cost is reduced proportionally. Similar efforts on combining mean shift with particle filter have also been reported in the work of Koichiro et al. (2004), Maggio & Cavallaro (2005), and Cai et al. (2006). However, for all the algorithms in this group, mean shift will inevitably make the samples too concentrated, aggravating another problem of particle filter - sample impoverishment.

In this paper, we further extend the idea of MSEPF and propose a novel algorithm, CamShift Guided Particle Filter (CAMSGPF). Here CamShift – an upgraded version of mean shift invented by Bradski (1998) – is used together with particle filter. Various mechanisms are devised to integrate the two kinds of basic algorithms so that they can benefit from each other. On one hand, CamShift optimizes the scale of each particle as well as its position, and thus plays an even stronger role than mean shift in improving the sampling efficiency of particle filter. On the other hand, the multiple hypothesis nature of particle filter enables CamShift to adjust scaling factor adaptively; and the redundancy between particles is explored so that CamShift can be applied in a simplified way on the whole particle set. Moreover, an ad-hoc scheme is employed in CAMSGPF to attack the problem of sample impoverishment. All these improvements are realized through a novel proposal density optimized by a modified CamShift, and the resultant filter remains valid from a Bayesian perspective.

Thanks to the features mentioned above, the proposed CAMSGPF algorithm is particularly competent for tracking video objects with great variation in scale, and has an advantage when tracking in cluttered background. The high efficiency of the algorithm is also appealing to applications where computational resources are limited. As verified by several real video sequences in our experiments, CAMSGPF outperforms standard particle filter and mean shift based trackers in terms of both robustness and efficiency.

The remainder of the paper is organized as follows. We first specify the system models in Section 2. The details on the integration of CamShift into particle filter are presented in Section 3. The ad-hoc scheme to resolve sample impoverishment is discussed in Section 4. Then experimental results are given in Section 5. In Section 6 we conclude the paper.

2 System Models

In our system, the target to be tracked in video sequence is modeled by a state vector:

$$\mathbf{x} = (x, y, l, h)^T \tag{1}$$

so that a rectangle box centered at coordinate (x, y) with length l and height h just covers the target in the image. For convenience, we denote

$$\mathbf{p} \equiv (x, y)^T \tag{2}$$

$$\mathbf{s} \equiv (l, h)^T \tag{3}$$

as the position and scale subspace of state vector.

The state dynamics are modelled with the commonly used standard 2nd order autoregressive process:

$$p(\mathbf{x}_t|\mathbf{x}_{0:t-1}) = \mathcal{N}(\mathbf{x}_t; 2\mathbf{x}_{t-1} - \mathbf{x}_{t-2}, \mathbf{\Sigma}_D) \tag{4}$$

where $\mathbf{x}_{0:t-1}$ denotes the state sequence up to time $t-1$, and $\mathcal{N}(;,)$ stands for Gaussian distribution. The dynamic equation (4) in fact predicts the current state $\mathbf{x}_t$ as the sum of three terms: the previous state $\mathbf{x}_{t-1}$, the displacement of last transition $\mathbf{x}_{t-1} - \mathbf{x}_{t-2}$, and a zero-mean gaussian noise with covariance matrix $\mathbf{\Sigma}_D$. This simple model can well simulate most object motions, and plays a vital role in simplifying the overall algorithm as can be seen later in Section 3.2.

The observation model here is based on HSV color histogram, a popular approach first used by Perez et al. (2002). Given the current observation $\mathbf{z}_t$ (i.e. the current image frame), the candidate color histogram $\mathbf{H}(\mathbf{x}_t, \mathbf{z}_t)$ is evaluated in a region of $\mathbf{z}_t$ covered by $\mathbf{x}_t$. Then it is compared with the reference color histogram $\mathbf{H}^*$ using Bhattacharyya similarity metric $D[,]$. The likelihood distribution is defined to have an exponential form that favors small Bhattacharyya metric:

$$p(\mathbf{z}_t|\mathbf{x}_t) \propto e^{-\lambda D^2[\mathbf{H}^*, \mathbf{H}(\mathbf{x}_t, \mathbf{z}_t)]} \tag{5}$$

where parameter λ is a set to 20 by Perez et al. (2002).

With the dynamic and observation models of Eq. (4) and (5), we can recursively estimate the target posterior $p(\mathbf{x}_t|\mathbf{z}_{0:t})$ on each arrival of new observation by any Bayesian filter. In the following sections, we will show how this is done with the proposed CamShift guided particle filter.

3 CamShift Guided Particle Filter

The framework of CAMSGPF is based on the well-known Sequential Importance Resampling (SIR) particle filter (Gordon et al., 1993). In SIR particle filter, the probability distribution of target state is approximated via a finite set of N samples (particles) with state $\{\mathbf{x}^i\}_{i=1\ldots N}$ and the associated weight $\{w^i\}_{i=1\ldots N}$. Given the sample set at the previous time, $\{\mathbf{x}^i_{t-1}, w^i_{t-1}\}_{i=1\ldots N}$, the filtering starts by propagating each sample with a stochastic displacement according to some proposal distribution $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_t)$. Observation is then made at the resultant state $\{\mathbf{x}^i_t\}_{i=1\ldots N}$ and the new sample weight is updated by:

$$w^i_t \propto w^i_{t-1} \frac{p(\mathbf{z}_t|\mathbf{x}^i_t) p(\mathbf{x}^i_t|\mathbf{x}^i_{0:t-1})}{q(\mathbf{x}^i_t|\mathbf{x}^i_{0:t-1}, \mathbf{z}_t)} \tag{6}$$

Finally, a resample step will take place when necessary, so that samples with large weight are multiplied and samples with small weight are eliminated. Thus we get the sample set $\{\mathbf{x}^i_t, w^i_t\}_{i=1\ldots N}$ which corresponds to the target posterior at current time.

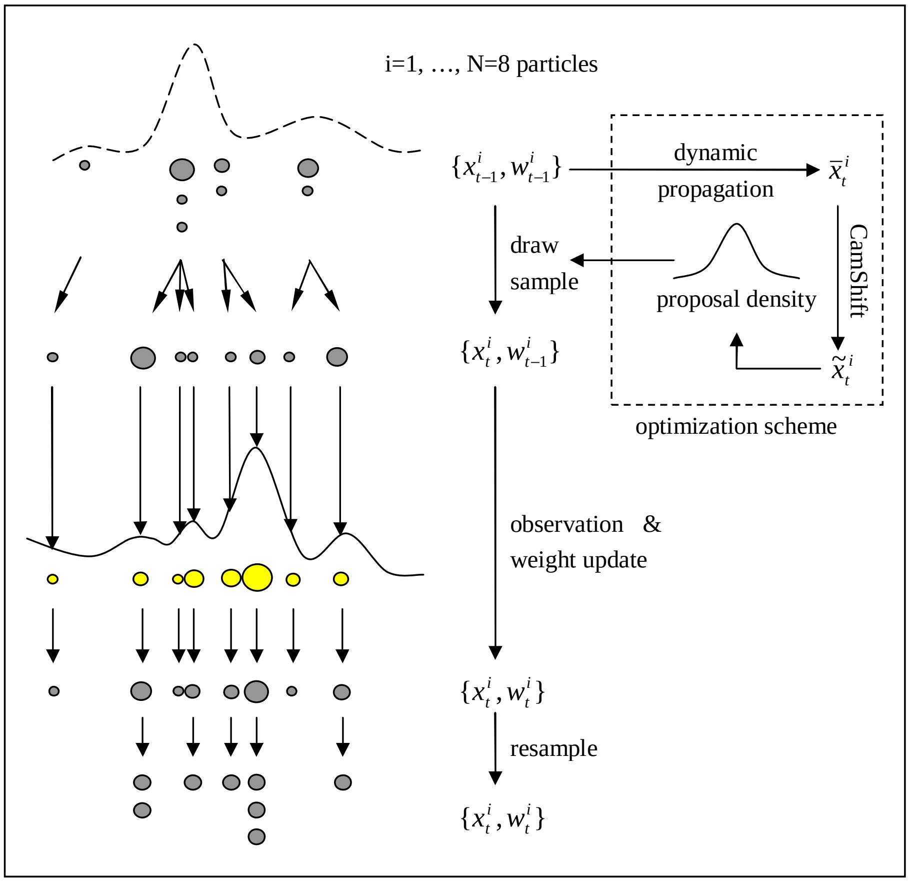


Fig. 1. The framework of CAMSGPF. Each blob's horizontal position and area represent the state and weight of a sample. The highlighted color indicates likelihood evaluated by observation model. The optimized proposal density is found in the dashed block.

The critical issue in SIR filter is the design of proposal density $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_t)$. Technically, any *pdf* supporting the target posterior can be considered. For example, the dynamic distribution $p(\mathbf{x}_t|\mathbf{x}_{0:t-1})$ is often used as proposal density for implementation convenience. However, a careless choice of proposal density may lead to poor sampling efficiency, generating a set of samples most of which stay far away from posterior modes – a problem known as degeneracy(Arulampalam et al., 2002).

In the proposed CAMSGPF algorithm, we introduce a new proposal density whose probabilistic parameter is optimized in the light of current observation before samples are drawn from it. For each particle $\mathbf{x}_{t-1}^i$, we first search a local mode $\tilde{\mathbf{x}}_t^i$ near its dynamic propagation center using CamShift method:

$$\tilde{\mathbf{x}}_t^i = CamShift\left(\bar{\mathbf{x}}_t^i, \mathbf{z}_t\right) \tag{7}$$

where $\bar{\mathbf{x}}_t^i = \mathrm{E}_{p(\mathbf{x}_t|\mathbf{x}_{0:t-1}^i)}[\mathbf{x}_t] = 2\mathbf{x}_{t-1}^i - \mathbf{x}_{t-2}^i$; $CamShift(\mathbf{x}, \mathbf{z})$ returns the CamShift result with initial value $\mathbf{x}$ and observation $\mathbf{z}$. Then the proposal density is defined as a Gaussian distribution[1] centered at $\tilde{\mathbf{x}}_t^i$:

$$q(\mathbf{x}_t^i|\mathbf{x}_{0:t-1}^i, \mathbf{z}_t) = \mathcal{N}\left(\mathbf{x}_t^i; \tilde{\mathbf{x}}_t^i, \mathbf{\Sigma}_{CAM}\right) \tag{8}$$

where $\mathbf{\Sigma}_{CAM}$ is a fixed covariance matrix capturing the potential error in CamShift. Eq. (8) is reminiscent of the Gaussian proposal density in unscented particle filter (UPF) proposed by Van der Merwe et al. (2000). UPF determines the parameters of Gaussian proposal via unscented transformation, whereas our method optimizes them by CamShift algorithm.

The proposal density of Eq. (8) takes into account current observation as well as state dynamics, so that particles drawn form it are implicitly driven towards posterior modes. In this way, fewer samples are required to approximate the posterior distribution and the problem of degeneracy is greatly alleviated. The framework of CAMSGPF is outlined in Fig. 1.

The idea presented above shows how CamShift can help particle filter draw a good sample set. On the other hand, we have found the performance of CamShift can also be improved in the context of particle filter: the optimization in scale subspace becomes more accurate and the computational complexity is substantially reduced. In the sequel, we shall discuss these issues in detail.

3.1 CamShift on One Sample

As implied by Eq. (7) and (8), in CAMSGPF, the Gaussian center of proposal density is obtained by applying CamShift procedure on the dynamic transition center $\bar{\mathbf{x}}_t^i$ of each sample. Let us denote the position and scale substates of $\bar{\mathbf{x}}_t^i$ as $\bar{\mathbf{p}}_t^i$ and $\bar{\mathbf{s}}_t^i$. They are adjusted iteratively to local mode $\tilde{\mathbf{p}}_t^i$ and $\tilde{\mathbf{s}}_t^i$, which constitute the Gaussian center $\tilde{\mathbf{x}}_t^i$.

The shift of position $\bar{\mathbf{p}}_t^i$ in one iteration is directed by mean shift vector (Comaniciu et al., 2000):

$$\mathbf{M}_{r,g}(\bar{\mathbf{p}}_t^i) = \frac{\sum_{j=1}^{M} \mathbf{p}_j m(\mathbf{p}_j) g\left(\left\|\frac{\bar{\mathbf{p}}_t^i - \mathbf{p}_j}{r}\right\|^2\right)}{\sum_{j=1}^{M} m(\mathbf{p}_j) g\left(\left\|\frac{\bar{\mathbf{p}}_t^i - \mathbf{p}_j}{r}\right\|^2\right)} - \bar{\mathbf{p}}_t^i \tag{9}$$

where $\{\mathbf{p}_j\}_{j=1\ldots M}$ are pixel coordinates within the rectangle area specified by state $\bar{\mathbf{x}}_t^i$; $m(\mathbf{p}_j)$ is the likelihood weight for pixel value at $\mathbf{p}_j$, which is the square root of corresponding histogram bin ratio between $\mathbf{H}^*$ and $\mathbf{H}(\bar{\mathbf{x}}_t^i, \mathbf{z}_t)$;

[1] In fact, the form of proposal density is not restricted to Gaussian. We use Gaussian here just for simplicity.

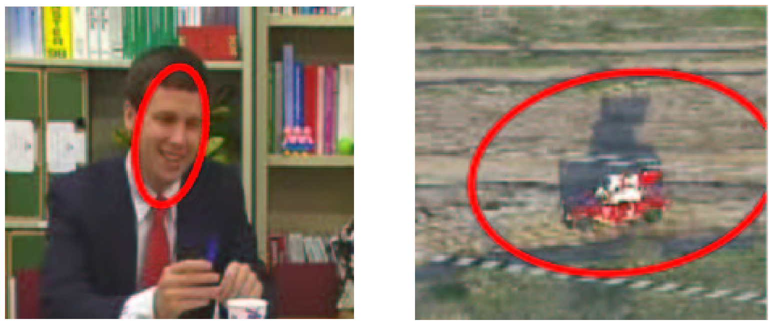

(a) (b)

Fig. 2. With the moment-scale relationship of Eq. (11), CamShift can successfully track a man's face (a); while fails to adjust the scale properly when tracking a red car in field (b).

$g()$ is a kernel profile function; and r is the window radius for normalization. With the above mean shift vector, the position $\bar{\mathbf{p}}_t^i$ is updated by

$$\bar{\mathbf{p}}_t^i \leftarrow \bar{\mathbf{p}}_t^i + \mathbf{M}_{r,g}(\bar{\mathbf{p}}_t^i) \tag{10}$$

As Eq. (9) and (10) are evaluated repeatedly, $\bar{\mathbf{p}}_t^i$ is guaranteed to converge. The converged value is used as the local mode $\tilde{\mathbf{p}}_t^i$.

The adjustment of the scale substate $\bar{\mathbf{s}}_t^i = (\bar{l}_t^i, \bar{h}_t^i)^T$ is also based on pixel likelihood weight $m(\mathbf{p}_j)$, which is the byproduct in evaluating mean shift vector according to Eq. (9). Since a denser pixel weight distribution (larger zeroth moment of the likelihood distribution image) implies a larger target size, the scale $\bar{\mathbf{s}}_t^i$ is adjusted by Bradski (1998) in proportion to the zeroth moment:

$$\bar{\mathbf{s}}_t^i \leftarrow k\sqrt{\frac{M_{00}(\bar{\mathbf{x}}_t^i)}{256 \times \bar{l}_t^i \times \bar{h}_t^i}} \cdot \bar{\mathbf{s}}_t^i \tag{11}$$

where k is a constant determined empirically, and the zero moment $M_{00}(\bar{\mathbf{x}}_t^i)$ is calculated by

$$M_{00}(\bar{\mathbf{x}}_t^i) = \sum_{j=1}^{M} m(\mathbf{p}_j) \tag{12}$$

where the pixel set $\{\mathbf{p}_j\}_{j=1\ldots M}$ is defined as in Eq. (9). Note the moment should be normalized by the maximum value of probability distribution (256 in the 8-bit case).

Since the relationship between zeroth moment and target size in Eq. (11) is found empirically, it can only work properly in limited situations. For example, with the parameter set by Bradski (1998), CamShift can track human face successfully, but may fail to adjust the target size in more general applications (Fig. 2).

This shortcoming is easily tackled in CAMSGPF with the multiple hypothesis nature of particle filter. When applying CamShift to each $\bar{\mathbf{x}}_t^i$, we will try moment-size estimation functions with different constant k^i, so that the resultant state $\tilde{\mathbf{x}}_t^i$ is granted the opportunities of both scaling up and scaling down. Here k^i is drawn randomly from two predefined candidates, k_1 and k_2 ($k_1 > k_2$), which enlarges and diminishes the size respectively:

$$k^i = \begin{cases} k_1 & \text{if } u < \frac{1}{2} \\ k_2 & \text{if } u \geq \frac{1}{2} \end{cases} \tag{13}$$

where variable u is randomly drawn from uniform distribution $\mathrm{U}[0, 1)$. The selected k^i is substituted into Eq. (11) when we update the target scale in each CamShift iteration. As different scales are tried on all proposal centers $\tilde{\mathbf{x}}_t^i$, samples generated from the proposal density can either increase or decrease in scale subspace. The well-scaled samples will gain a larger weight through observation model, while the badly-scaled samples will be eliminated in the process of resampling. With this multi-hypothesis and test paradigm, we can properly adjust the target size without knowing the precise moment-size relationship.

3.2 Efficiency Improvement

The time consumption of CAMSGPF is greatly reduced with the proposal density optimized by CamShift, because fewer particles are required to accomplish target state tracking. At the same time, however, this improvement in efficiency is partly cancelled out by the introduction of CamShift, since CamShift takes extra time besides particle filter in the overall algorithm. What's worse, a traditional CamShift (or mean shift) procedure is more complicated than a generic particle filter recursion cycle. So it is expected that the computation spent in finding proposal density by CamShift will dominate in CAMSGPF algorithm, which becomes the bottleneck to further improve the algorithm efficiency.

In fact, when integrated with particle filter, the CamShift procedure can be applied in a much simpler manner than it is used alone. The simplification lies in several aspects:

1) At each time step, we do not need to apply CamShift on all dynamic transition center $\bar{\mathbf{x}}_t^i$'s. Instead, only one CamShift procedure is needed for a group of $\bar{\mathbf{x}}_t^i$'s that are adjacent in state space, since the states within a neighborhood will be shifted to the same position and share approximately the same zero moment density:

$$\begin{cases} \tilde{\mathbf{p}}_t^j \approx \tilde{\mathbf{p}}_t^i \\ m_{00}(\bar{\mathbf{x}}_t^j) \approx m_{00}(\bar{\mathbf{x}}_t^i) \end{cases} \quad \text{for } \bar{\mathbf{x}}_t^j \in \Omega(\bar{\mathbf{x}}_t^i) \tag{14}$$

where $\Omega(\bar{\mathbf{x}}_t^i)$ is a neighborhood centered at $\bar{\mathbf{x}}_t^i$, and the zero moment density $m_{00}(\bar{\mathbf{x}}_t^i)$ is defined as

$$m_{00}(\bar{\mathbf{x}}_t^i) \equiv \frac{M_{00}\left(\bar{\mathbf{x}}_t^i\right)}{256 \times \bar{l}_t^i \times \bar{h}_t^i} \tag{15}$$

Insert Eq. (14), (15) into Eq. (11), then the scale substate after I iterations of CamShift can be approximated by:

$$\tilde{\mathbf{s}}_t^j \approx (k^j)^I \sqrt{m_{00}(\bar{\mathbf{x}}_t^i, I)} \cdot \bar{\mathbf{s}}_t^j \quad \text{for } \bar{\mathbf{x}}_t^j \in \Omega(\bar{\mathbf{x}}_t^i) \tag{16}$$

where $m_{00}(\bar{\mathbf{x}}_t^i, I)$ stands for the accumulated product of $m_{00}(\bar{\mathbf{x}}_t^i)$ in I iterations.

However, clustering dynamic transition centers into groups according to their proximity in state space is itself a time-demanding task, usually with time complexity quadratic to the number of center points. In CAMSGPF, a suboptimal but linear-time-complexity method is used to group the centers $\{\bar{\mathbf{x}}_t^i\}_{i=1\ldots N}$. Remember $\bar{\mathbf{x}}_t^i$'s are generated by propagating particles $\{\mathbf{x}_{t-1}^i\}_{i=1\ldots N}$ from the last epoch. As duplicated samples in $\{\mathbf{x}_{t-1}^i\}_{i=1\ldots N}$ are indexed sequentially in the resampling step[2], it will be very likely that two particles with consecutive indices are within a neighborhood:

$$p\left(\mathbf{x}_{t-1}^{i+1} \in \Omega(\mathbf{x}_{t-1}^i)\right) \gg 0 \tag{17}$$

Since $\bar{\mathbf{x}}_t^i$ is related to $\mathbf{x}_{t-1}^i$ through a common dynamic equation for any particle, it naturally leads to an analogy to Eq. (17) for $\bar{\mathbf{x}}_t^i$:

$$p\left(\bar{\mathbf{x}}_t^{i+1} \in \Omega(\bar{\mathbf{x}}_t^i)\right) \gg 0 \tag{18}$$

With the above knowledge, we are able to quickly cluster $\bar{\mathbf{x}}_t^i$'s into adjacent neighborhoods in a linear search. We inspect for each dynamic transition center $\bar{\mathbf{x}}_t^i$ to see whether it is close enough to the previous center $\bar{\mathbf{x}}_t^{i-1}$. If that is the case, the previous CamShift result can be reused as suggested by Eq. (14) and Eq. (16); otherwise, a new CamShift is calculated for its own. It will be shown in Section 5.2 that this mechanism can reduce the number of necessary CamShifts considerably.

2) In CAMSGPF, CamShift is terminated after a small and fixed number of iterations, as opposed to the traditional procedure, which goes on until convergence. Even with a limited number of iterations, particles drawn from

[2] This is the case for most resample schemes, like systematic resampling.

the sub-optimized proposal density can still get closer to local mode at each time. This halfway drift to local mode is built into the displacement term of the 2nd order autoregressive dynamic model in Eq. (4), and will be carried to future time steps. Therefore, even if a sample is not shifted to the local mode at current time step, it can still transit in that direction "inertially", as long as the posterior distribution does not change drastically.

3) The validity of mean shift vector is no longer verified in our simplified CamShift. Because of the complicated likelihood distribution, the mean shift vector found by Eq. (9) will sometimes point away from local maximum. Thanks to particle filter, these improper shifts can be well handled in our algorithm. When a sample is drawn from a proposal whose center $\tilde{\mathbf{x}}_t^i$ has been shifted or scaled apart from the true target state, it will be assigned a low weight by particle filter and eventually eliminated in resampling stage. So in CAMSGPF, the role of CamShift is to give particles some crude implications on propagation direction (through proposal density), while the judgment and feedback are left to particle filter. It is because of this interactive relationship that we name this algorithm CamShift *guided* particle filter.

With the above simplifications, substantial improvement in efficiency is guaranteed for our algorithm, since the computational load of CAMSGPF is proportional to the number of CamShift's for each proposal density optimization, and the computation for each CamShift is proportional to the number of iterations and verifications. The notation $CamShift()$ throughout this paper refers to this simplified CamShift algorithm, unless otherwise stated. The algorithm is summarized in Table 1, with parameter I as the fixed number of iterations. The input arguments include initial state set $\{\bar{\mathbf{x}}_t^i\}_{i=1...N}$ and current observation $\mathbf{z}_t$; the output is the optimized proposal density centers $\{\tilde{\mathbf{x}}_t^i\}_{i=1...N}$.

4 Resolving Impoverishment Problem

With CamShift integrated in CAMSGPF, all particles tend to transit in the same direction (to local modes). After a few iterations, most of them will collide together and sample diversity is severely lost. This gives rise to the problem of sample impoverishment (Arulampalam et al., 2002), which is fatal to the performance of tracker when proposal center returned by CamShift is trapped in false local mode. In order to maintain the balance between sampling effectiveness and diversity, we amend the original proposal density in Eq. (8) by superimposing a term of the dynamic model in Eq. (4):

$$q'(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_t) = \alpha p(\mathbf{x}_t|\mathbf{x}_{0:t-1}) + (1-\alpha) q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_t) \qquad (19)$$

Table 1
Algorithm of the Simplified CamShift in CAMSGPF

```
$\{\tilde{\mathbf{x}}_t^i\}_{i=1...N} = CamShift(\{\bar{\mathbf{x}}_t^i\}_{i=1...N}, \mathbf{z}_t)$

• for $(i = 1 : N)$
  ∘ Draw $u \sim \mathrm{U}[0, 1)$
  ∘ if $(u < \frac{1}{2})$
        Set $k^i = k_1$
  ∘ else
        Set $k^i = k_2$
  ∘ endif
  ∘ if $(i == 1$ or $\bar{\mathbf{x}}_t^i \notin \Omega(\bar{\mathbf{x}}_t^{i-1}))$
        Set $\mathbf{x}^{ref} \equiv (\mathbf{p}^{ref}, \mathbf{s}^{ref})^T = \bar{\mathbf{x}}_t^i, \quad m_{00}(\mathbf{x}^{ref}, 0) = 1$
        for $(j = 1 : I)$
            Find mean shift vector $\mathbf{M}_{r,g}(\mathbf{p}^{ref})$ by Eq. (9)
            Shift position: $\mathbf{p}^{ref} = \mathbf{p}^{ref} + \mathbf{M}_{r,g}(\mathbf{p}^{ref})$
            Find zero moment density $m_{00}(\mathbf{x}^{ref})$ by Eq. (15)
            Accumulate moment: $m_{00}(\mathbf{x}^{ref}, j) = m_{00}(\mathbf{x}^{ref}, j-1) \times m_{00}(\mathbf{x}^{ref})$
        endfor
  ∘ endif
  ∘ Set $\tilde{\mathbf{x}}_t^i = \left(\mathbf{p}^{ref}, (k^i)^I \sqrt{m_{00}\left(\mathbf{x}^{ref}, I\right)} \cdot \bar{\mathbf{s}}_t^i\right)^T$
• endfor
```

where α is a constant controlling the relative importance of two linear components. In the final form of our proposed algorithm, samples will be drawn directly from $q'()$, and the weight is updated accordingly. We shall see in Section 5.1 that this ad-hoc amendment to proposal density indeed makes our CAMSGPF more robust in challenging tracking task.

5 Experimental Results

In this section, the performance of CAMSGPF is compared with other trackers from a number of aspects. In the experiments, the parameters of CAMSGPF are set as follows: $k_1 = 1.2$; $k_2 = 0.9$; $I = 2$; $\alpha = 0.5$; $\Omega(\bar{\mathbf{x}}_t^i) = \{\mathbf{x} \mid \|\mathbf{x} - \bar{\mathbf{x}}_t^i\|_2 < 2\}$; $\mathbf{\Sigma}_D = diag\{1.0, 0.25, (5, 4; 4, 5) \times 10^{-4}\}$; $\mathbf{\Sigma}_{CAM} = diag\{0.25, 0.25, 2 \times 10^{-4}, 2 \times 10^{-4}\}$. All the tests are carried out on 320×240-pixel sequences with a Pentium IV 2.8G PC. When we present tracking results in the following, the particle states are visualized on pictures as blue rectangles, while their average (the final output) is represented by a red bold rectangle.

5.1 Tracking Robustness

We first demonstrate the robustness of the proposed algorithm under various tracking conditions. Fig. 3 shows the CAMSGPF tracker is able to track a fast moving hockey player with only 10 particles. The same sequence is also tracked by a traditional particle filter (PF), with 100 particles and 10 particles respectively. PF with 100 particles is observed to track the target correctly throughout the sequence, and its output is used as benchmark here. The tracking results of the three trackers are compared in Fig. 4, which shows clearly that CAMSGPF with 10 particles can track almost as well as PF with 100 particles; while PF with 10 particles fails completely.

In Fig. 5, we compare several trackers' adaptation to substantial change of target scale in the "soccer" sequence. Some of the tracking results by CAMSGPF are listed in the third column, in comparison with those tracked by MSEPF and CamShift in the first and second column. It is observed that our method has the best adaptation to the change of target scale, as its output rectangle fits the target most closely.

Fig. 6 shows some tracking results carried out on "F1", a sequence featured with fast object motion and complete occlusion (in short time). With the

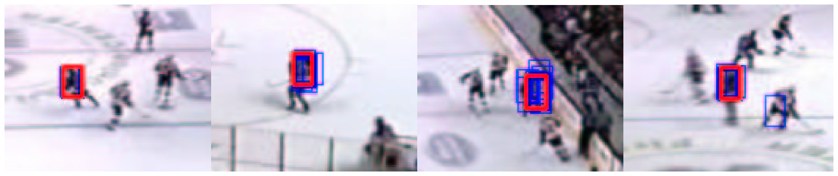
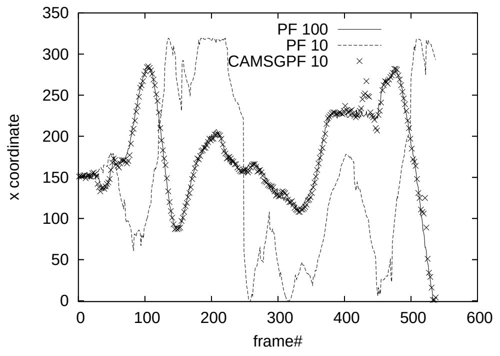
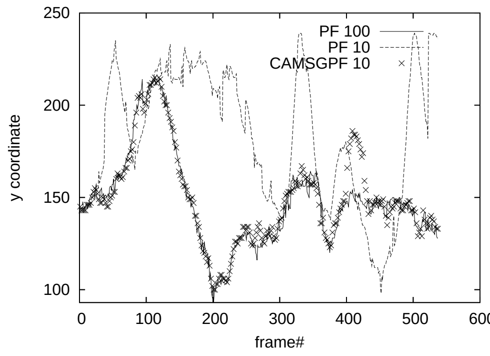


Fig. 3. "Hockey" sequence successfully tracked by CAMSGPF with 10 particles (frame 56, 211, 427, 516)



(a)



(b)

Fig. 4. Tracking results on "hockey" sequence by particle filter with 100 particles (solid line), 10 particles (dashed line), and CAMSGPF with 10 particles (crossed point). (a) results of x coordinate. (b) results of y coordinate.

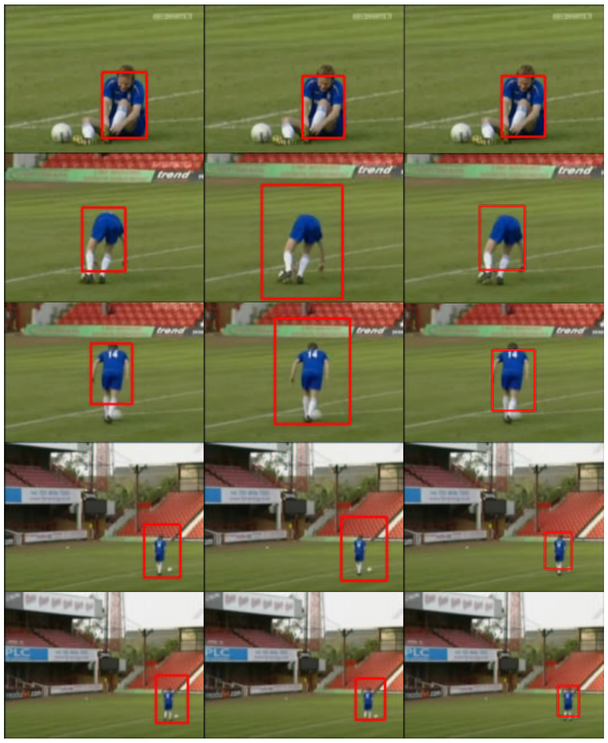


Fig. 5. “Soccer” sequence tracked by MSEPF, CamShift and CAMSGPF (in the columns from left to right). (frame 1, 114, 123, 176, 193)

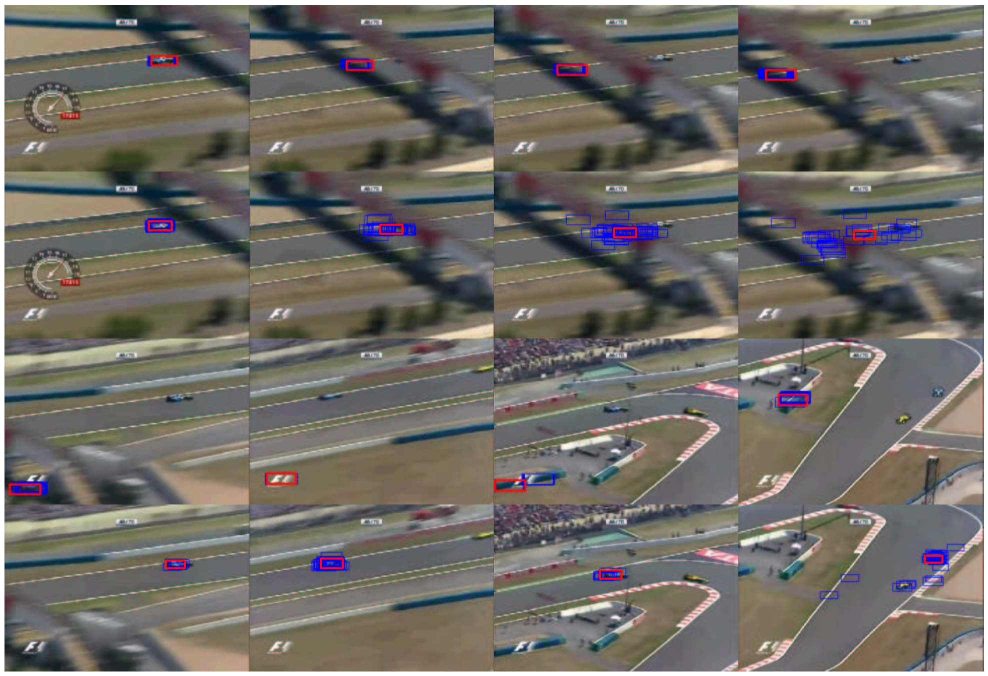


Fig. 6. “F1” sequence tracked by MSEPF (odd rows) and CAMSGPF (even rows); CAMSGPF deals with the temporal occlusion well. (frame 12, 17, 19, 21, 28, 66, 109, 177)

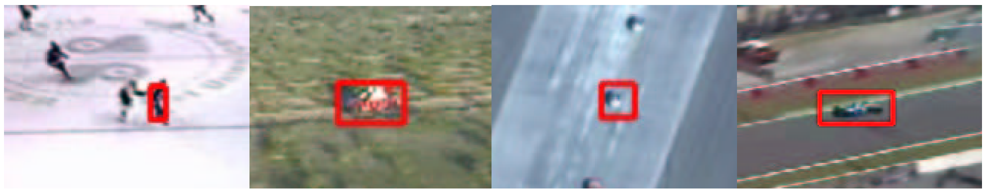


Fig. 7. Tracking results on "hockey", "redteam", "egtest01" and "F1" sequence.

method of MSEPF (the odd number rows), particles suffer seriously from impoverishment and overlap with each other most of the time. So when the occlusion occurs, they may get trapped to some false local maximum, and cannot recover from failure even when the target reappears. On the contrary, CAMSGPF can deal with the temporal occlusion well and tracks successfully throughout the sequence (the even number rows). Due to the dynamic propagation part in Eq. (19), the particles in CAMSGPF will spread more widely during the occlusion, thus granting the tracker a good chance to regain the target state when it shows up again.

5.2 Tracking Efficiency

To verify the efficiency achieved by adopting the simplified version of CamShift, we compare the time consumption of CAMSGPF with MSEPF on several sequences, where both of them can track correctly, as illustrated in Fig. 7. Each sequence has been tracked by the two algorithms with a set of particle numbers, and the average time consumptions per frame are plotted in Fig. 8 (a)-(d). For all the sequences, the time consumptions of both algorithms grow linearly with the number of particles, but CAMSGPF is consistently faster than MSEPF.

This improvement is mainly attributed to the reduced number of CamShifts. As can be seen in Fig. 9, the actual number of CamShifts required to meet the approximation condition of Eq. (14) is only about half of the total particle number. Without executing a lot of redundant CamShifts, we can thus reduce the computational complexity of CAMSGPF tremendously.

Table 2 summarizes the performance of MSEPF and CAMSGPF on four sequences quantitatively. The average time consumption per particle, namely, the slope of the lines in Fig. 8, is listed in the left columns. We find that it takes CAMSGPF 30% to 50% less time than MSEPF to process each particle. In the right columns of Table 2, the average Bhattacharyya similarity metric between the tracking results and the target template is shown to indicate the tracking accuracy. The sacrifice in accuracy is trivial with respect to the time saved for all the sequences; and the great advantage of CAMSGPF over MSEPF in "F1" further confirms the result in Fig. 6.

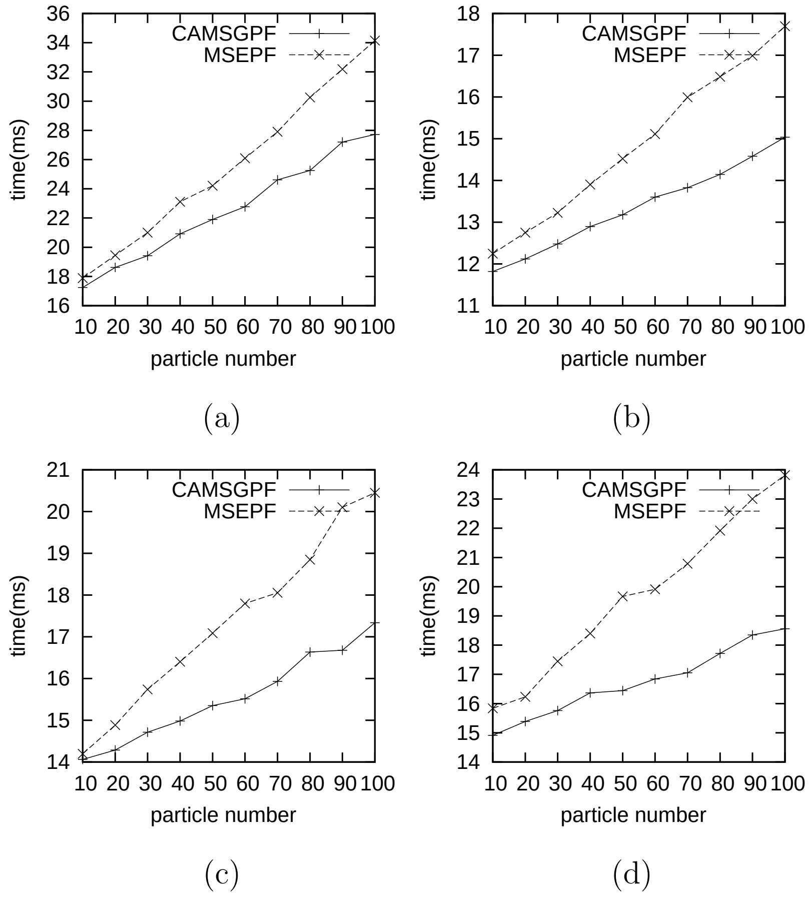


Fig. 8. Time consumption versus particle number, for CAMSGPF and MSEPF. (a) “redteam”; (b) “hockey”; (c) “egtest01”; (d) “F1”

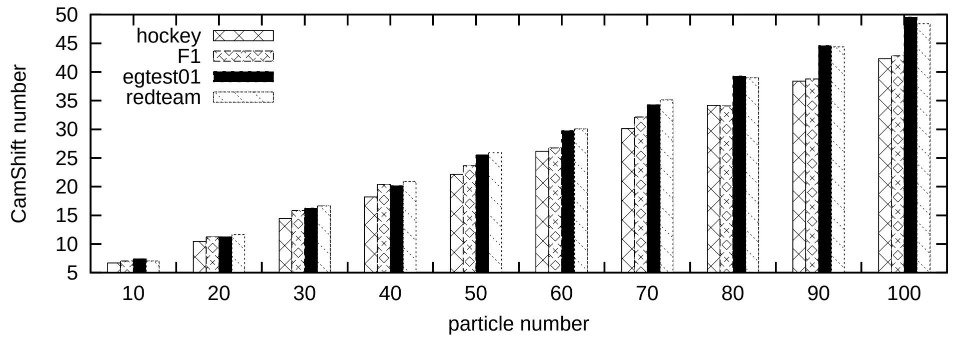


Fig. 9. Number of CamShifts applied in CAMSGPF with respect to different particle numbers.

6 Conclusions

A novel tracking algorithm, CAMSGPF, has been proposed by exploring the interaction between particle filtering and CamShift. CamShift helps improve the sampling efficiency of particle filter in both position and scale space; and meanwhile, in the context of particle filter, CamShift achieves better scale

Table 2
Comparison of efficiency and accuracy between MSEPF and CAMSGPF

Sequence	Time/Particle(ms)		Time	Similarity metric*		Similarity
	MSEPF	CAMSGPF	saved	MSEPF	CAMSGPF	difference
hockey	0.0618	0.0350	43.40%	0.0903	0.0985	9.10%
redteam	0.1807	0.1185	34.43%	0.1150	0.1296	12.74%
egtest01	0.0691	0.0357	48.40%	0.1213	0.1231	1.52%
F1	0.0903	0.0399	55.84%	0.2695	0.2030	-24.66%

* a smaller similarity metric indicates a better match between two objects

adaption and can be applied in a simplified way without much loss in performance. All these improvements, together with an anti-impoverishment scheme, are wrapped up in a proposal density function with parameter optimized by a modified CamShift, so that the overall Bayesian filter remains valid. The experimental results demonstrate that CAMSGPF outperforms CamShift and MSEPF in both tracking robustness and efficiency.

Acknowledgment

We thank Prof. Ercan E. Kuruoğlu from ISTI, Area Della Ricerca CNR for the helpful discussion, and the anonymous reviewers for the valuable comments that improved the paper substantially. This work was supported in part by NSFC (60502034), 863 Program (2006AA01Z124), Shanghai Postdoctoral Foundation (06R214138) and the 111 Project.

References

Arulampalam, M.S., Maskell, S., Gordon, N., & Clapp, T., 2002. A Tutorial on Particle Filters for On-line Nonlinear/ Non-Gaussian Bayesian Tracking, IEEE Transactions of Signal Processing, vol. 50(2), pages 174-188.

Bradski, G.R., 1998. Computer vision face tracking as a component of a perceptual user interface, the Workshop on Applications of Computer Vision, pages 214-219, Princeton, NJ.

Cai, Y., Freitas, N., Little, J., 2006 Robust Visual Tracking for Multiple Targets, European Conference on Computer Vision, vol. 4, pp. 107-118.

Carpenter, J., Clifford, P., & Fearnhead, P., 1999. Improved particle filter for nonlinear problems, IEE Proceedings of the Radar, Sonar, Navig., vol. 146(1):2-7.

Comaniciu, D., & Ramesh, V., 2000. Mean Shift and Optimal Prediction for

Efficient Object Tracking, IEEE International Conference on Image Processing, Vancouver, Canada, pp. 70-73.

Comaniciu, D., Ramesh, V., & Meer, P., 2000. Real-time tracking of non-rigid objects using mean shift, IEEE Conference on Computer Vision and Pattern Recognition, pages II: 142-149, Hilton Head, SC.

Gordon, N., Salmond, D., & Smith, A. F. M., 1993. Novel approach to nonlinear and non-Gaussian Bayesian state estimation, IEE Processinds-F, vol. 140, pp. 107-113.

Hammersley, J.M., & Morton, K.M., 1954. Poor man's Monte Carlo, Journal of the Royal Statistical Society B, 16, 23-38

Isard, M., & Blake, A., 1998. Condensation-conditional density propagation for visual tracking, International Journal on Computer Vision, 29(1), pp. 5-28.

Kanazawa, K., Koller, D., &Russell, S. J., 1995. Stochastic simulation algorithms for dynamic probabilistic networks, Proc. Eleventh Annu. Conf. Uncertainty AI, pp. 346-351.

Koichiro, D., Oki, K., & Takayuki, O., 2004. Object tracking by the mean-shift of regional color distribution combined with the particle-filter algorithm, proceedings of the 17th International Conference on Pattern Recognition, 3:506-509.

MacCormick, J., & Blake, A., 1999. A probabilistic exclusion principle for tracking multiple objects, the Proceedings of the Seventh IEEE International Conference on Computer Vision, vol.1, pp. 572-578.

Maggio, E., & Cavallaro, A., 2005 Hybrid particle filter and mean shift tracker with adaptive transition model, Proc. of IEEE Signal Processing Society International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 2, pp. 221-224.

Perez, P., Hue, C., Vermaak J. & Gangnet, M., 2002. Color-Based Probabilistic Tracking, European Conference on Computer Vision, pp. 661-675.

Shan, C., Wei, Y., Tan, T., & Ojardias, F., 2004. Real time hand tracking by combining particle filtering and mean shift, Sixth IEEE International Conference on Automatic Face and Gesture Recognition.

Van der Merwe, R., Doucet, A., de Freitas, J.F.G. & Wan, E., 2000 The Unscented Particle Filter, Technical Report CUED/F-INFENG/TR 380, Cambridge University Engineering Department.