Expert Systems with Applications 37 (2010) 8910–8914

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Short Communication

Particle filter based on Particle Swarm Optimization resampling for vision tracking

Jing Zhao *, Zhiyuan Li

College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China

ARTICLE INFO

Keywords:
Object tracking
Particle filter
Particle Swarm Optimization

ABSTRACT

Particle filter is a powerful tool for vision tracking based on Sequential Monte Carlo framework. The core of particle filter in vision tracking is how to allocate particles to a high posterior area. Particle Swarm Optimization (PSO) is applied to find high likelihood area in this paper. PSO algorithm can search the sample area around the last time object position depending on current observation. So, it can distribute the particles in high likelihood area even though the dynamic model of the object cannot be obtained. Our algorithm does not distribute the particles based on the weight of the particles last time like the sampling-importance resampling (SIR). SIR usually allures particles distributed in wrong likelihood area particularly tracking in cluttered scene. Since that some particles have larger weight maybe illusive. We first find the sample area by PSO algorithm, then we distribute the particles based on two different base points in order to achieve diversity and convergence. Experimental results in several real-tracking scenarios demonstrate that our algorithm surpasses the standard particle filter on both robustness and accuracy.

1. Introduction

In recent years, there has been a great deal of interest in applying particle filtering by Arulampalam, Maskell, Gordon, and Clapp (2002), also known as Condensation (Isard & Blake, 1998), to computer vision problems. Applications on rigid or non-rigid object tracking has demonstrated its usefulness (Choo & Fleet, 2001; Zhang & Pece, 2003).

The most appealing aspect of PF is to maintain multiple hypothesis of a state, which makes them more competent for heavily cluttered and complex scenes. However, PF are not always satisfactory, especially when the irregular and abrupt motion renders a weak dynamic model in real scenes (Chang & Ansari, 2005). There is a motion model in the transition equation. The motion model can help distribute the particles in the right sample area. But sometimes we do not know the motion model of the object. If we do not use the motion model, the particles may be distributed in the wrong sample area when the object is similar to background (Zhang & Pece, 2003).

The resampling scheme in Condensation is sampling particles in last time particle set according to their weight. The larger weight the particle has, the more times the particle will be chosen to generate the next time particles. But that some particles have large weight is illusive in clutter scene and when occlusion occurs. These "large" weight particles are far from the high likelihood area. So they will abduct the particles distributed in the wrong area piece by piece.

An improved strategy to overcome these problems is to design better proposal distributions. An auxiliary particle filter (APF) is one such example provided by Pitt and Shephard (1999), which generates particles from an importance distribution depending on a more recent observation. Its weakness is that it requires a large number of particles. When the state transition density is quite scattered and the likelihood varies significantly over the state transition distribution, APF is not always effective.

Another enhanced tactic for PF has been extensively studied since the pioneering work of Comaniciu, Ramesh, and Meer (2003), who were the first to introduce mean-shift analysis to visual tracking. Following their work, Maggio and Cavallaro (2005), Shan, Wei, Tan, and Ojardias (2004) subtly extended mean shift to the particle-filter framework. The central idea of their algorithms is to redistribute particles to their local mode of the posterior density by mean-shift analysis, thereby possibly using fewer particles to keep multiple hypotheses. The particles in this algorithm have lack of diversity. It performs ineffectively when occlusion occurs. Different from the work of Maggio and Cavallaro (2005), Shan et al. (2004), Chang and Ansari (2005) presented another method which approximated posterior density using kernel density estimate (KDE) and then estimated the gradient of posterior density by mean-shift analysis. Mean shift plays different roles in the above two kinds of methods in that it was used for maximizing the similarity function between the target candidate and the target model by Maggio and Cavallaro (2005), which was used as mode seeking of posterior density in Chang and Ansari (2005). De-

* Corresponding author.
E-mail address: zhaoj@hrbeu.edu.cn (J. Zhao).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.05.086

spite successful particle redistribution, no immediate observations are taken into consideration before sampling by Chang and Ansari (2005).

In this paper, we present a new method to distribution particles. The particles will simulate the posteriori density accurately only when particles distribute in the likelihood area. Our work is as follows, firstly, we utilize some particles to search the distribution area based on Particle Swarm Optimization (PSO) by Kennedy and Eberhart (2001). Thus, we do not need to know the motion model of the object. Secondly, we distribute particles in this area based on two base points to simulate the posteriori density. The search space in PSO is a little area around the object position. Because the position of the object in the continuous frames is so close. The particles are used to search avoid the illusive weight. This distribution method can combine diversity and convergence. We called our algorithm PSO-PF. The framework of our algorithm is shown in Fig. 1.

The remainder of the paper is organized as follows. In Section 2, particle-filter algorithm is first reviewed. Our algorithm is outlined in Section 3. The observation model of the object is presented in Section 4. In Section 5, we illustrate some experimental results. Finally, we conclude the paper and point out future work.

2. Particle filter

Generic Bayesian filtering is used to estimate the state of a nonlinear dynamic system sequentially in time. However, some difficulties exist, namely, intractable integrations in estimating the posterior density. To solve these difficulties, probability densities in Bayesian filtering are represented by means of point-mass, this technique is called particle filtering. A continuous state vector of a target object at time step t is denoted by $x_t \in R^{N_x}$ with its history $X_{1\ldots t} = \{x_1, \ldots, x_t\}$, and a measurement vector is $z_t \in R^{N_x}$ with its history $Z_{1\ldots t} = \{z_1, \ldots, z_t\}$. The target dynamics is assumed to be represented as a temporal Markov chain:

$$p(x_t|x_{1:t-1}) = p(x_t|x_{t-1}) \tag{1}$$

$$p(Z_{1\ldots t}|X_{1\ldots t}) = \prod_{i=1}^{t} p(z_i|x_i) \tag{2}$$

According to the Bayes rule, the posterior density is then given by

$$p(x_t|z_{1:t}) = k_t p(z_t|x_t) p(x_t|z_{1:t-1}) \tag{3}$$

where k_t is the normalization term and

$$p(x_t|z_{1:t-1}) = \int p(x_t|x_{t-1}) p(x_{t-1}|z_{1:t-1}) \tag{4}$$

is the prediction density as the prior. This recursion seems simple, but the computation should include intractable integrations. In particle filtering, by using a set of samples and the corresponding weights at time step t, the posterior is approximated as $\left\{(x_t^{(n)}, w_t^{(n)}),\ n = 1, \ldots, N\right\}$

$$p(x_t|z_{1:t}) \approx \sum_{n=1}^{N} w_t^{(n)} \delta\left(x_t - x_t^{(n)}\right) \tag{5}$$

where $\delta\left(x_t - x_t^{(n)}\right)$ is Dirac's delta function. Then, the prior is approximated as

$$p(x_t|z_{1:t-1}) \approx \sum_{n=1}^{N} w_t^{(n)} p\left(x_t|x_{t-1}^{(n)}\right) \tag{6}$$

The weight $w_{t-n}^{(n)}$ is determined such that $w_{t-n}^{(n)} \propto p\left(z_{t-1}|x_{t-1}^{(n)}\right)$. $\sum_{n=1}^{N} w_{t-n}^{(n)} = 1$.

If large number of particles can be used to sample, the posteriori density would be accurate. But it is unpractical in real-time tracking. So, the limited particles been sampled in accurate area is important in vision tracking.

3. Particle Swarm Optimization resampling

3.1. Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a population based stochastic optimization technique developed by Dr. Eberhart and Dr. Kennedy in 1995, inspired by social behavior of bird flocking or fish schooling.

PSO learned from the scenario and used it to solve the optimization problems. In PSO, each single solution is a "bird" in the search space. We call it "particle". All of the particles have fitness values which are evaluated by the fitness function to be optimized, and have velocities which direct the flying of the particles. The particles fly through the problem space by following the current optimum particles.

PSO is initialized with a group of random particles (solutions) and then searches for optima by updating generations. In every iteration, each particle is updated by following two "best" values. The first one is the best solution (fitness) it has achieved so far, and the fitness value is also stored, this value is called *pbest*. Another "best" value that is tracked by the particle swarm optimizer is the best value, obtained so far by any particle in the population. This best value is a global best called *gbest*. When a particle takes part of the population as its topological neighbors, the best value is a local best called *lbest*.

After finding the two best values, the particle updates its velocity and positions with following equation (*a*) and (*b*).

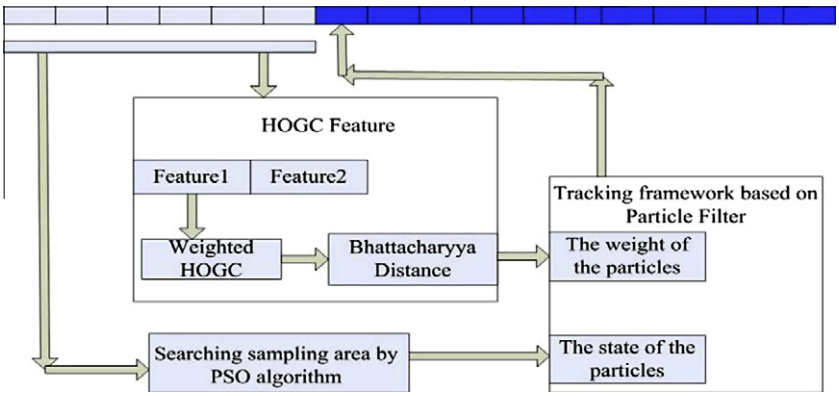

Fig. 1. The framework of our algorithm.

```
For each particle
    Initialize particle
END
Do
    For each particle
        Calculate fitness value
        If the fitness value is better than the best fitness value (pBest) in history
            set current value as the new pBest
    End
    Choose the particle with the best fitness value of all the particles as the gBest
    For each particle
        Calculate particle velocity according equation (a)
        Update particle position according equation (b)
    End
While maximum iterations or minimum error criteria is not attained
```

Fig. 2. The pseudo code of PSO algorithm.

$$V[] = V[] + C1 * rand() * (pbest[] - present[]) + C2 * rand() * (gbest[] - present[]) \quad (a)$$

$$present[] = present[] + v[] \quad (b)$$

$V[]$ is the particle velocity, $present[]$ is the current particle (solution). $pbest[]$ and $gbest[]$ are defined as stated before. $rand()$ is a random number between $(0,1)$. $C1$, $C2$ are learning factors. Usually $C1 = C2 = 2$.

The pseudo code of the procedure is as shown in Fig. 2.

3.2. PSO resampling

Firstly, we need to find the sample area at time step t depending on current observation. In the usual vision, object center position at t is around the one at $t-1$. But we do not know the object motion model sometimes, we cannot know the object position yet. In SIR, sampling at time step t is depending on these large weight particles at time step $t-1$. Some large weight particles is illusive. SIR allures particles sampling in the wrong area especially in the clutter scene. We search the right sample area first of all.

We randomly sample 10 particles from the object center position at time step $t-1$. These particles are used to search likelihood area. The fitness of the particle is the similar measure between the candidate model and the target model. We set maximum iterations is five times. Because PSO is an iteration algorithm in search space, it will cost much time. But PSO in our algorithm does not need iterate so many times. The first reason is that searching space is a small area around the object position at times step $t-1$ and initialized particles are also around the object position. The second reason is that we do not need to find the idiographic position where the object is at next time. We only need to find the right sample area to help particle filter perform well. The experiments demonstrate that five times is appropriate.

When the iteration ends, the *gBest* and *pBest* are calculated. At this step, we distribute particles according to *gBest* and *pBest.* In order to improve the diversity, we distribute a subset of particles from *gBest*. The remainder of particles are distributed from *pBest.*

4. Observation model

In order to improve the robustness of the algorithm we present, we combine the color histogram (HC) and histogram of orientations gradient (HOGC), called HOGC as our observation model. The color histogram is unaware to rotation while the gradient one is unaware to color. It considers the color representation and contour representation of an object respectively.

We define a color histogram (HC) of 48 dimensions for both the object and its background. In each color component (R, G and B in RGB color space), 16 dimensions of histogram features are extracted. Color Histogram is described in some paper by Comaniciu et al. (2003).

We extract HOG features on gray value image windows. On each window, a histogram of 72 dimensions is extracted to descript the gradient orientation of an object. HOG feature is the evolvement of edge orientation histograms by Geronimc, Lopez, Ponsa, and Sappa (2007) and SIFT descriptors by Lowe (2004). Fig. 3 indicates the ***HOG extraction***, and steps are as follows:

1. resizing the object rectangle region into an image window of fixed size, $32 * 32$.
2. divide the image window into small spatial regions (cell) with the size of $8 * 8$, a $2 * 2$ group of cells is a sliding widow (9 windows).
3. the orientation is 8-bin.
4. calculate the orientation $ori(h,w)$ of each pixel in the sliding window according to Eq. (7).

$$\begin{aligned} I &= G(\sigma, 0) * I_0 \\ dy &= I(h+1, w) - I(h-1, w) \\ dx &= I(h, w+1) - I(h, w-1) \\ ori(h, w) &= atan2(dy, dx) \quad ori \in [-\pi, \pi] \end{aligned} \quad (7)$$

Then we calculate the histogram obtained $72(8 * 9)$ dimensions.

To make the HOG features handle objection rotation, we use the method in SIFT by Lowe (2004). An orientation histogram is formed from the gradient orientations within a region around the center of the image window. That is to say, the center point of the image window is a SIFT keypoint. The orientation histogram has 18 bins covering the 360° range of orientations in our proposed approach. Peaks in the orientation histogram correspond to dominant directions of local gradients. The highest peak in the histogram is detected, and then we rotate the image window to the model we initialed at the beginning of the tracking process. Therefore, the HOG of the image window is independent to the orientation here.

The Bhattacharyya distance is used to measure the similarity distance between the object model and candidate model.

5. Experiments

In this section, Particle Filter based on PSO is tested on public tests to compare performance with the traditional Particle Filter on robustness and accuracy.

In our experiments, the sate of the particles is the object position. The measure is the HOGC feature. The transition equation is

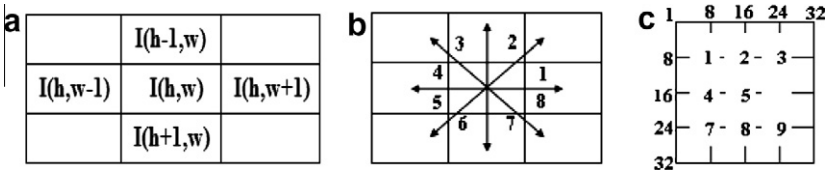

Fig. 3. HOG extraction. (a) Mask for pixel gradient calculation, (b) Orientation bin for voting, (c) An image window partitioned into 9 blocks with the same size.

$$x_{k+1} = f(k, x_k) + w_k \tag{8}$$

where w_k is zero mean Gaussian noise. We do not use the motion equation $f(k,x_k)$. In real life situations, especially with variational motion and interacting with other objects of the scene, it is very difficult to know the object dynamics beforehand. Different object will have different dynamics. In our algorithm, we do not need to know the motion model. PSO is used to searching new sample area.

We compare robustness between PF and PSO-PF in Figs. 4–6. We compare accuracy in Fig. 7. We calculate the error on X axes and Y axes according the equation

$$\text{Error on } X = \frac{|X_t \cdot x - \widehat{X}_t \cdot x|}{W} \tag{9}$$

$$\text{Error on } Y = \frac{|X_t \cdot y - \widehat{X}_t \cdot y|}{H} \tag{10}$$

Fig. 4 shows the result of tracking a white car interacted by other cars. Fig. 4(a) shows that PSO-PF algorithm performs well. Firstly HOGC feature describes the object from the color and configuration. Although HOGC feature is discriminative, the tracking may fail because of the illusive large weight particles. PSO can search the sample center around the object center depending on current observation. Because the next object position must be around the upper time object position. PSO-PF can always track

Fig. 4. (a) PSO-PF tracking. (b) PF tracking.

Fig. 5. (a) PSO-PF tracking. (b) PF tracking.

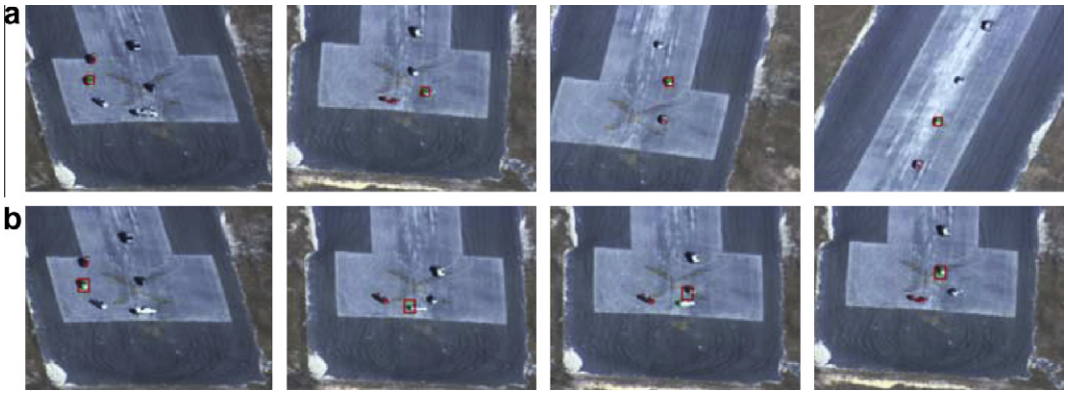

Fig. 6. (a) PSO-PF tracking. (b) PF tracking.

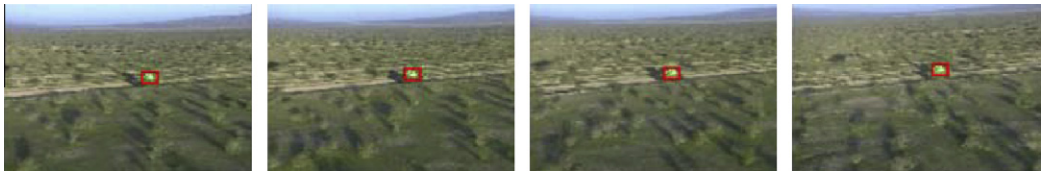

Fig. 7. PSO-PF tracking.

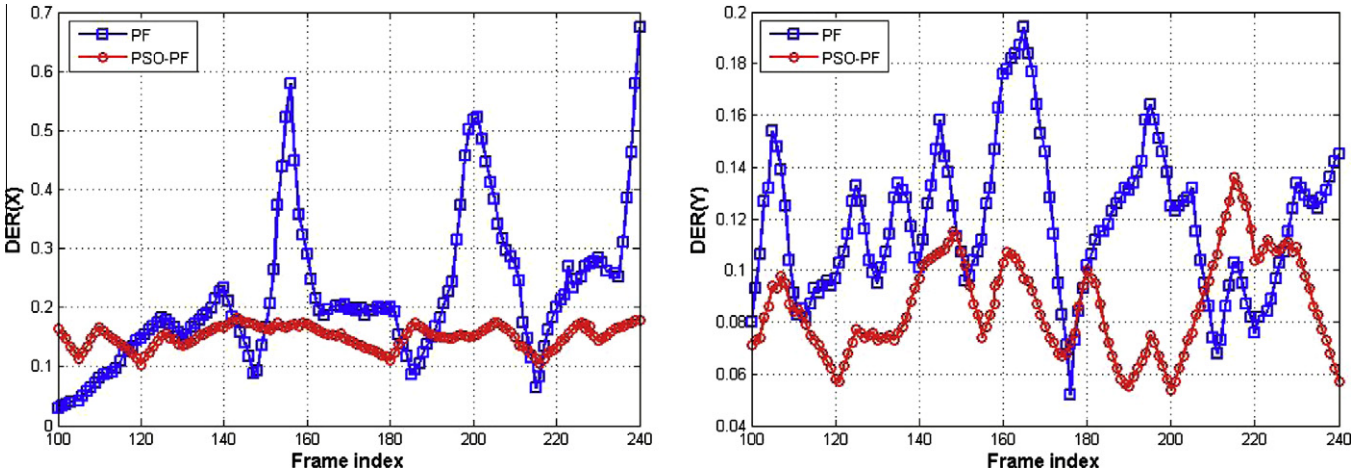

Fig. 8. The tracking error of PF and PSO-PF on Fig. 7.

the object especially when the car interact with the other three cars. In Fig. 4(b), when the tracked car encounters the first white car, the particles distribute dispersedly. The PF tracking failed.

Fig. 5 shows the result of tracking a car occluded by trees. Tracking an occluded object is difficult, because the observation is wrong when occlusion occurs. The particle Filter is a prediction model. So it can predict the object position when brief occlusion occurs. When the right observation cannot be obtained, SIR leads the particle distribution dispersedly. The PSO has searched the sample area around the object position (It may be not accurate at this time, but the approximate orientation is right). We utilize the two sample base points. The diversity and the convergence are all maintained. Fig. 5(a) shows the result of PSO-PF tracking. It can successfully track the car after came out from the tree. In Fig. 5(b), PF tracking is failed when occlusion occurs.

Fig. 6 shows the result of object tracking with illumination, configuration change and weak dynamic model. In the beginning, we only see the front and top of the car. Then illumination changes when the car veers. Afterwards we can see the top and the back of the car. We do not know the motion model yet. Fig. 6(a) is the result of the PSO-PF tracking, It performs well. Fig. 6(b) is the result of the PF tracking. The tracking fails. As the weight of some particles is illusive, it abducts the particles to form the wrong distribution in SIR. So the particles in PF cannot be convergence.

Fig. 7 shows the result of a tank tracked in the grassland. The performance is well by both PSO-PF tracking and PF tracking. The difference between two methods is accuracy. From the Fig. 8, we know that the accuracy in PSO-PF tracking is higher than the PF tracking both on X axes and Y axes.

6. Conclusion and future works

In this paper, we present an efficient resampling method based on Particle Swarm Optimization. The experiments demonstrate that our algorithm performs well in some complex scene. In the future, we need to enhance the efficiency PSO algorithm in PF. How to guide the particles to distribute reasonably when occlusion occurs is our next research.

Acknowledgements

This work is supported by the National Nature Science Foundation of China (60873036) and Ph.D. Programs Foundation of Ministry of Education of China under Grant No. 20070217051.

References

Arulampalam, S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for on-line nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing, 50*(2), 174–188.

Chang, C., & Ansari, R. (2005). Kernel particle filter for visual tracking. *IEEE Signal Processing Letters, 12*(3), 242–245.

Choo, K., & Fleet, D. J. (2001). People tracking using hybrid Monte Carlo filtering. In *Proceedings of international conference on computer vision. Vancouver, Canada.*

Comaniciu, D., Ramesh, V., & Meer, P. (2003). Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 25*(5), 564–577.

Geronimc, D., Lopez, A., Ponsa, D., & Sappa, A. D. (2007). Haar wavelets and edge orientation histograms for on board pedestrian detection. *Pattern Recognition and Image Analysis. Girona, Spain* (pp. 418–425).

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

Kennedy, J., & Eberhart, R. C. (2001). *Swarm intelligence*. Morgan Kaufmann.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *Journal of Computer Vision., 60*(2), 91–110.

Maggio, E., & Cavallaro, A. (2005). Hybrid particle filter and mean shift tracker with adaptive transition model. In *Proceedings of IEEE signal processing society international conference on acoustics, speech, and signal processing (ICASSP), Philadelphia, PA, USA.*

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

Shan, C., Wei, Y., Tan, T., & Ojardias, F. (2004). Real time hand tracking by combining particle filtering and mean shift. In *Sixth international conference on automatic face and gesture recognition.*

Zhang, L. P. T., & Pece, A. E. C. (2003). Visual contour tracking based on particle filters. *Image and Vision Computing, 21*, 111–123.