Video Target Tracking Based on a New Adaptive Particle Swarm Optimization Particle Filter

Feng Liu[1], Shi-bin Xuan[1,2], and Xiang-pin Liu[1]

[1] College of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China
[2] Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Nanning 530006, China
sbinxuan@gxun.cn

Abstract. To improve accuracy and robustness of video target tracking, a tracking algorithm based on a new adaptive particle swarm optimization particle filter (NAPSOPF) is proposed. A novel inertia weight generating strategy is proposed to balance adaptively the global and local searching ability of the algorithm. This strategy can adjust the particle search range to adapt to different motion levels. The possible position of moving target in the first frame image is predicted by particle filter. Then the proposed NAPSO is utilized to search the smallest Bhattacharyya distance which is most similar to the target template. As a result, the algorithm can reduce the search for matching and improve real-time performance. Experimental results show that the proposed algorithm has a good tracking accuracy and real-time in case of occlusions and fast moving target in video target tracking.

Keywords: Adaptive, Particle Swarm Optimization, Particle Filter, Video Target Tracking, Occlusions, Fast Moving Target, Bhattacharyya Distance.

1 Introduction

Video target tracking is a key problem in computer vision, which has a wide range of applications in human-computer interaction, visual surveillance, intelligent transportation and military guidance etc [1]. In spite of the substantial research effort expended to tackle this challenge, developing a robust and efficient tracking algorithm still remains unsolved due to the inherent difficulty of the tracking problem.

Recently, particle filters (PF) have been extensively used in video target tracking field [2-4], which have been proved to be a robust method of tracking due to the ability of solving non-Gaussian and non-linear problems [5]. Nevertheless, PF may confront with the problem of weight degradation which if solved by re-sampling method may result unavoidable particle impoverishment [6]. As a result, the accuracy and robustness of target tracking are influenced.

To address that problem, a tracking algorithm based on a new adaptive particle swarm optimization particle filter (NAPSOPF) is proposed. Bhattacharyya distance is

D.-S. Huang et al. (Eds.): ICIC 2013, LNAI 7996, pp. 163–171, 2013.
© Springer-Verlag Berlin Heidelberg 2013

utilized as the measurement of similarity between models of target template and candidate region in the algorithm based on color histogram [7]. The possible position of moving target in the first frame image is predicted by particle filter, and matching the target template and candidate regions with the color histogram statistical characteristics in order to ensure the tracking accuracy. Then the proposed NAPSO is utilized to search the smallest Bhattacharyya distance which is most similar. As a result, the algorithm can estimate a more realistic state and reduce the re-sampling frequency of particle filter, so that the computational cost of particle filter is effectively reduced.

The experimental results prove that the proposed algorithm has a good tracking accuracy and real-time in case of occlusions and fast moving target in target tracking.

2 PSO-PF Algorithm

2.1 Basic PSO Algorithm

PSO is a stochastic, population-based optimization algorithm. It was originally proposed by Eberhart [8]. PSO algorithm can be expressed as follows: to randomly initialize a particle swarm whose number is $\boldsymbol{m}$ and dimension is $\boldsymbol{d}$, in which the particle's position is $\boldsymbol{X}_i = (\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \cdots, \boldsymbol{x}_{id})$ and its speed is $\boldsymbol{V}_i = (\boldsymbol{v}_{i1}, \boldsymbol{v}_{i2}, \cdots, \boldsymbol{v}_{id})$. During each iteration, the particles can renew their own speed and position through partial extremum and global extremum so as to reach optimization. The update formula is:

$$\boldsymbol{V}_{id} = \omega * \boldsymbol{V}_{id} + \boldsymbol{c}_1 * \boldsymbol{Rand}() * (\boldsymbol{P}_{id} - \boldsymbol{X}_{id}) + \boldsymbol{c}_2 * \boldsymbol{Rand}() * (\boldsymbol{G} - \boldsymbol{X}_{id}) \tag{1}$$

$$\boldsymbol{X}_{id} = \boldsymbol{X}_{id} + \boldsymbol{V}_{id} \tag{2}$$

Where $\boldsymbol{Rand}$ is a random number within interval $(0,1)$, ω is the inertia coefficient, $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ are learning factors.

2.2 Principle of Standard PSO-PF Algorithm

In the re-sampling process of regular particle filtering, the particles with bigger weights will have more offspring, while the particles with smaller weights will have few or even on offspring. It inspires us to find more particles with small weights, which will make the proposal distribution closer to the posteriori distribution. And it is the aim of using particles swarm optimization in the particles filtering.

The most important issue of using particles swarm optimizer is the choice of fitness function. Giving the fitness function as follows:

$$z_k \sim \boldsymbol{fitness} = \exp[-\frac{1}{2\boldsymbol{R}_k}(z_k - \hat{z}^i_{k|k-1})^2] \tag{3}$$

Where z_k is the latest observed, $\hat{z}^i_{k|k-1}$ is the predictive observed value.

Take N particles $\{\boldsymbol{x}^i_{0:k}\}_{i=1}^N$ as samples from importance function at the initial time. Calculate the importance value: $\omega^i_k = \omega^i_{k-1} \exp[-\frac{1}{2\boldsymbol{R}_k}(z_k - \hat{z}^i_{k|k-1})^2]$

Then update the velocity and position of particle:

$$v_{k+1}^{i} = \omega * v_k^i + c_1 r_1 * (p_{pbest} - x_k^i) + c_2 r_2 * (p_{gbest} - x_k^i) \tag{4}$$

$$x_{k+1}^{i} = x_k^i + v_{k+1}^i \tag{5}$$

Where r_1 , r_2 is a random number within interval $(0,1)$, ω is the inertia coefficient, c_1 and c_2 are learning factors.

Calculate the importance weight of the particle after optimization and perform normalization: $\omega_k^i = \omega_k^i / \sum_{i=1}^{N} \omega_k^i$

Note that particle weights after re-sampling step are all equal to $1/N$, finally state output: $\hat{x}_k = \sum_{i=1}^{N} \omega_k^i x_k^i$

3 A New Adaptive Particle Swarm Optimization (NAPSO)

To overcome the problems of PSO which is easy to fall into local optimum in the whole iterative process and has a low convergence rate in the late iterative process, NAPSO is proposed. The adaptive adjustment process: inertia weight is a key parameter to balance the ability of local search and global search, a larger inertia weight value facilitates global exploration which enables the algorithm to search new areas, while a smaller one tends to facilitate local exploitation to increase the search accuracy [9]. In view this; the inertia weight will be adjusted adaptively by the fitness of particles in this paper. Adaptive weight is defined as follows:

$$\omega = \begin{cases} \omega_{\min} & \text{if } f_i \text{ is better than } f'_{avg} \\ \omega_{\min} + (\omega_{\max} - \omega_{\min}) \dfrac{|f_{best} - f_i|}{|f_{best} - f_{avg}|} & \text{if } f_i \text{ is better than } f''_{avg} \text{ but worse than } f'_{avg} \\ \omega_{\max} & \text{if } f_i \text{ is worse than } f''_{avg} \end{cases} \tag{6}$$

Where f_i is the fitness of particle, $f_{avg} = \frac{1}{N}\sum_{i=1}^{N} f_i$ is the average fitness of particles, f'_{avg} is the average fitness of the particles which are better than f_{avg} , f''_{avg} is the average fitness of the particles which are worse than f_{avg} , f_{best} is the fitness of the best particle, $\omega_{\max}$ is the maximum inertia weight and $\omega_{\min}$ is the minimum inertia weight. $0 < \frac{|f_{best} - f_i|}{|f_{best} - f_{avg}|} < 1$ Ensure that $\omega \in [\omega_{\min}, \omega_{\max}]$, and a better fitness of the particles tends to a smaller ω , which tends to facilitate local exploitation to increase the search accuracy.

The particles are divided into three group based on the fitness of the particles in order to adopt different inertia weight generating strategy. Combining with the degree of premature convergence and individual adaptive value adjust the inertia weight, this algorithm is effective in controlling the particle swarm diversity, and it also has good convergence rate and can avoid getting into the local optimum.

4 Video Target Tracking Based on NAPSOPF

4.1 The Dynamic Model

We represent the target regions by rectangles or ellipses, so that a sample is given as

$$S = [x, \hat{x}, y, \hat{y}, H_x, H_y]^T \tag{7}$$

Where (x, y) represent the location of the ellipse (or rectangle), $\hat{x}$, $\hat{y}$ are velocities in x and y directions, H_x , H_y are the length of the half axes.

The sample set is propagated through the application of a dynamic model [10]:

$$S_k = \begin{bmatrix} 1 & T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} S_{k-1} + \begin{bmatrix} \frac{T^2}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & T & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{T^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & T & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} W_k \tag{7}$$

Where T is the sampling period, and W_k is a random vector drawn from the noise distribution of the system.

4.2 Observation Model and the PSO Fitness Function

The observation model is used to measure the observation likelihood of the samples, and is important issue for target tracking.

In our tracking algorithm we need a similarity measure which is based on color distributions. A popular measure between two distributions $p(u)$ and $q(u)$ is the Bhattacharyya coefficient,

$$\rho[p, q] = \int \sqrt{p(u)q(u)}du \tag{8}$$

Considering discrete densities such as our color histograms $p = \{p^{(u)}\}_{u=1...m}$ and $q = \{q^{(u)}\}_{u=1...m}$ the coefficient is defined as

$$\rho[p, q] = \sum_{u=1}^{m} \sqrt{p_u q_u} \tag{9}$$

As distance between two distributions we define the measure, which is called the Bhattacharyya distance: $d = \sqrt{1 - \rho[p, q]}$

The smaller d is, the more similar the distributions are, so we choose the Bhattacharyya distance d as the fitness function of the NAPSO algorithm to Force the particles to be closer to the real state.

4.3 NAPSOPF in the Tracking Algorithm Process

As we know, the standard PSO-PF can improve the particle degradation of PF and is easier for actualization. Unfortunately it is a process of iterative optimization which will prolong the calculation time because of the high iterative frequency, and it may be easily trapped into local optimization, influencing the accuracy of video target tracking. Moreover, the fitness function (3) can't fit the tracking. To solve those problems, NAPSOPF tracking algorithm is proposed.

The tracking algorithm based on NAPSOPF:

Firstly, we manually select the target template in the first frame and calculate the color histogram of the target region, generate a particle set of N particles, then predict each particle using the dynamic model and calculate the color histograms of each particle, the distance between the target region color histogram and the color histograms of each particle is calculated using the Bhattacharyya similarity coefficient.

We use d as the fitness function of NAPSO , the algorithm can drive the particles into regions of the smaller fitness value, the smaller the fitness value of d is, the better match the particle is, accordingly forcing the particles to be closer to the real state:

Based on the Bhattacharyya distance, a color likelihood model is defined as

$$\boldsymbol{p}(z_k \mid \boldsymbol{S}_k^i) = \frac{1}{\sqrt{2\pi}\sigma}\left(-\frac{\boldsymbol{d}^2}{2\sigma^2}\right) \tag{10}$$

Finally state output: $\hat{\boldsymbol{S}}_k = \sum_{i=1}^{N} \omega_k^i \boldsymbol{S}_k^i = \sum_{i=1}^{N} \omega_{k-1}^i \boldsymbol{p}(z_k \mid \boldsymbol{S}_k^i)\boldsymbol{S}_k^i$

The best matched candidates for tracking can be obtained by comparing the Bhattacharyya distance between the target region and each particle. The estimated state is closer to the real state and the tracking accuracy is improved, even in the case of occlusions. Moreover, NAPSOPF solve the problem of low precision and complicated calculation of the standard PSO-PF, which can improve improves the real-time capability in the target tracking.

As we know, inertia weight is a key parameter to balance the ability of local search and global search, a larger inertia weight value facilitates global exploration which enables the algorithm to search new areas, while a smaller one tends to facilitate local exploitation to increase the search accuracy. The NAPSOPF algorithm adapts to different motion levels by setting the value of the maximum inertia weight in NAPSO, which can solve the low accuracy problem of fast moving target in video target tracking, because the displacement of the target between two consecutive frames is large.

Based on a large amount of experiments, $\omega_{\min}$ is set to 0.1. In case of fast motion object, $\omega_{\max}$ is set to 0.6~0.8; normal speed object, $\omega_{\max}$ is set to 0.4~0.6; slow motion object, $\omega_{\max}$ is set to 0.2~0.4.

Process of NAPSOPF tracker algorithm:

(1) Initialize particles $\left\{\boldsymbol{S}_{k-1}^i\right\}_{i=1}^N$ around the previous target and assign each particle an equal weight: $\omega_0^i = 1/\boldsymbol{N}$, calculate the color histogram of the target region $\boldsymbol{H}_0 = \left\{\boldsymbol{q}^{(u)}\right\}_{u=1}^{\tau}$.

(2) Propagate each sample set $\left\{\boldsymbol{S}_k^i\right\}_{i=1}^N$ from the set $\left\{\boldsymbol{S}_{k-1}^i\right\}_{i=1}^N$ by equation (7).

(3) Observe the color distributions: calculate the color distribution for each sample of the set S_k^i ; calculate the Bhattacharyya distance between the target template and each sample of the set S_k^i : $d = \sqrt{1-\rho[p,q]} = \sqrt{1-\sum_{u=1}^{\hat{r}}\sqrt{p_i^{(u)}q^{(u)}}}$

(4) The NAPSO algorithm is led in, and the particles are divided into three group based on the value of fitness function d in order to adopt different inertia weight generating strategy:

$$\omega = \begin{cases} \omega_{\min} & d_i < d'_{avg} \\ \omega_{\min} + (\omega_{\max} - \omega_{\min}) \dfrac{|d_{best} - d_i|}{|d_{best} - d_{avg}|} & d'_{avg} < d_i < d''_{avg} \\ \omega_{\max} & d_i > d''_{avg} \end{cases} \tag{11}$$

Where d_i is the fitness of particle, d_{avg} is the average fitness of particles, d'_{avg} is the average fitness of the particles which are smaller than d_{avg} , d''_{avg} is the average fitness of the particles which are bigger than d_{avg} , d_{best} is the fitness of the best particle, $\omega_{\max}$ is the maximum inertia weight and $\omega_{\min}$ is the minimum inertia weight. The velocity and position of the particles are updated according the following equations (4), (5):

(5) Calculate the observation likelihood: $p(z_k \mid S_k^i) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1-\rho[H_i, H_0]}{2\sigma^2}\right)$

(6) Calculate the importance weight of the particles after optimization and perform normalization: $\omega_k^i = \omega_{k-1}^i p(z_k \mid S_k^i)$, $\omega_k^i = \omega_k^i / \sum_{i=1}^{N} \omega_k^i$

(7) Estimate the mean state: $\hat{S}_k = \sum_{i=1}^{N} \omega_k^i S_k^i$.

(8) Re-sampling step then $k = k+1$ Go back to (2)

5 Simulation and Results Analysis

We run our algorithm on AMD Athlon (tm) II X2 B24 Processor 2.99GHz PC with 1.75GB memory. We implement all codes by MATLAB R2012a and visual studio.

5.1 Tracking Performance

Several real video sequences are used to compare the three implemented tracking algorithms. We illustrate our results with three test sequences A, B and C.

Sequence A is a football game video. As shown in Figure 1, the target is occluded by another athlete around frame 10; the figure shows that NAPSOPF algorithm is more accurate in case of occlusions than PF and PSOPF algorithm.

Sequence B is a racing video clip, the speed of the target is about 200km/h. To adjust the particle search range, we set the maximum inertia weight. Figure 2 shows the performance of NAPSOPF algorithm is more accurate than PF and PSOPF algorithm when tracking a very fast moving target.

Sequence C is a soccer match video clip, the speed of the ball is about 90-110km/h and the ball is occluded by the goal around frame 46, we set the maximum inertia weight $\omega_{\max} = 0.5$. As shown in Figure 3, the NAPSOPF algorithm is more accurate and robust in case of fast moving target which is occluded.

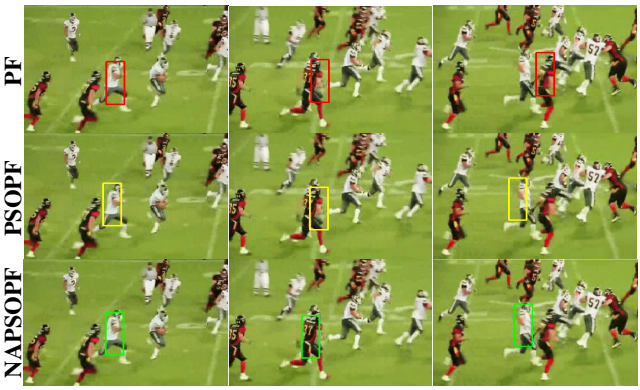

Fig. 1. Trackers performance with Sequence A frame 5, 10, 15

Fig. 2. Trackers performance with Sequence B frame 55, 65, 70, 75

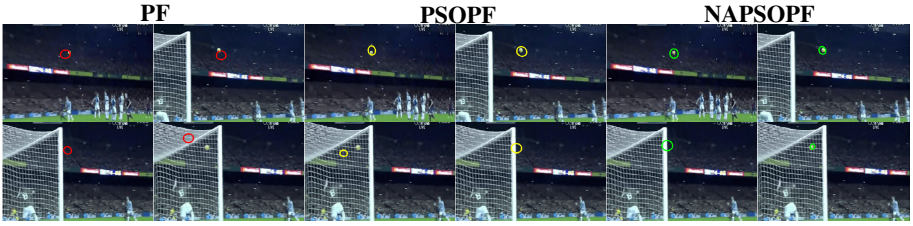

Fig. 3. Trackers performance with Sequence C frame 31,41,46,48

5.2 Results Analysis

Experimental results of Sequence A, B and C are shown in Table 1~Table 2.

Based on the results provided in these two tables, it is clear that, PF has the worst results, but has the best running time. While the NAPSOPF algorithm has the best

results, it reduce the re-sampling frequency of particles filter, so that the computational cost of particle filter is effectively reduced, the algorithm can meet the requirement of real-time tracking by handling about 10 frames per second.

Table 1. comparison between PF, PSOPF and NAPSOPF simulation parameters

Algorithm	Sequence A		Sequence B		Sequence C	
N=300	Effective Number	Re-sampling	Effective Number	Re-sampling	Effective Number	Re-sampling
PF	155.969	13.3	167.087	14.4	141.391	15.9
PSOPF	173.124	10.7	178.369	12.6	147.827	14.3
NAPSOPF	189.317	7.7	187.570	9.9	150.945	13.0

Table 2. comparison of running time between PF, PSOPF and NAPSOPF

Particles Number	Algorithm	Sequence A (20 frames)	Sequence B (25 frames)	Sequence C (22 frames)	Average frames per second
N=300	PF	1.954	2.249	2.076	10.649
	PSOPF	2.174	2.632	2.372	9.324
	NAPSOPF	2.044	2.351	2.177	**10.175**
N=100	PF	1.580	1.835	1.631	13.257
	PSOPF	1.776	2.189	1.895	11.431
	NAPSOPF	1.689	1.941	1.747	**12.438**

6 Conclusions

The NAPSOPF tracking algorithm can enable the particles to fit the environment better and then estimate a more realistic state, thereby it improves the accuracy and robustness of occlusions and fast moving target in video target tracking. The experiment results indicate that the algorithm has a good tracking accuracy and real-time in case of occlusions and fast moving target in video target tracking.

Acknowledgements. This research was supported by the National Science Foundation Council of Guangxi (2012GXNSFAA053227).

References

1. Collns, R., Lipton, A., Kanadeandt.: A System for Video Surveillance and Monitoring VSAM Final report. Carnegie Mellon University (2000)
2. Belagiannis, V., Schubert, F., Navab, N., Ilic, S.: Segmentation based particle filtering for real-time 2D object tracking. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 842–855. Springer, Heidelberg (2012)

3. Loris, B., Marco, C., Vittorio, M.: Decentralized Particle Filters for Joint Individual-Group Tracking. In: CVPR (2012)
4. Chong, Y., Chen, R., Li, Q., Zheng, C.-H.: Particle filter based on multiple cues fusion for pedestrian tracking. In: Huang, D.-S., Gupta, P., Zhang, X., Premaratne, P. (eds.) ICIC 2012. CCIS, vol. 304, pp. 321–327. Springer, Heidelberg (2012)
5. Wang, A.X., Li, J.J.: Target Tracking Based on Multi-core Particle Filtering. Computer Science 39(8), 296–299 (2012)
6. Doucet, A., Godsill, S.: On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. Statistics and Computing 10(1), 197–208 (2000)
7. Katja, N., Esther, K., Luc, V.G.: Object Tracking with and Adaptive Color-based Particle filter. In: Proceedings of the 24th DAGM Symposium on Pattern Recognition, Zurich, Switzerland, September 16-18, pp. 353–360 (2002)
8. Kennedy, J., Eberhart, R.C.: Particle Swarm Optimization. In: Proc of the IEEE Intl. Conf. on Neural Networks, Perth, Australia, pp. 1942–1948. IEEE Service Center, Piscataway (1995)
9. Shi, Y.H., Eberhart, R.C.: A Modified Particle Swarm Optimizer. In: Proceedings of The IEEE Congress on Evolutionary Computation, pp. 69–73. IEEE Service Center, Piscataway (1998)
10. Li, A.P.: Research on Tracking Algorithm for Visual Target under Complex Environments, pp. 20–31. Shanghai Jiao Tong University, Shanghai (2006)