Fundamenta Informaticae XXI (2001) 1–12
IOS Press

Particle swarm optimization based object tracking

Bogdan Kwolek
Rzeszów University of Technology, Rzeszów, Poland
bkwolek@prz.edu.pl

Abstract. This paper proposes a particle swarm optimization based algorithm for object tracking in image sequences. In each frame the particles are drawn from a Gaussian distribution in order to cover the promising object locations and afterwards the particle swarm optimization takes place in order to concentrate the particles near the true object state. The aim of the particle swarm optimization is to shift the particles toward more promising regions in the search area. A grayscale appearance model that is learned on-line is utilized in evaluation of the particles score. Experimental results that were obtained in a typical office environment show the feasibility of our approach, especially when the object undergoing tracking has a rapid motion or the appearance changes are considerable. The resulting algorithm runs in real-time on a standard computer.

Keywords: Swarm intelligence, particle swarm optimization, visual object tracking

1. Introduction

Swarm Intelligence (SI) is a technique comprising the study of collective behavior of relatively simple agents in decentralized systems. These systems are made up by a population of many agents that communicate with each other and with their environment. Although there is typically no centralized control dictating the behavior of the agents and the provision of a global model, each agent by cooperating with other agents is nevertheless able to solve complex tasks. The interactions between agents lead to the emergence of global behavior. Such self-organized systems include ant colonies, animal herding, bird flocking, fish schooling, honey bees, bacteria, and many more. Swarm-based algorithms, such as Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Stochastic Diffusion Search (SDS), have already been applied to solve real world applications such as, traffic routing, networking, crowd simulation, games, robotics.

Address for correspondence: Rzeszów University of Technology, W. Pola 2, 35-959 Rzeszów, Poland

In particle swarm optimization, simple software agents, called particles, move in the search space, where the position of a particle represents a candidate solution to the optimization problem. Each particle searches for better positions in the search space by changing its velocity according to rules originally inspired by behavioral models of bird flocking. Thus, potential solution flies via the search space by following the current optimum particles and by adjusting itself according to its former experience and its relationship to other agents in the population and the environment. The PSO is a remarkable algorithm in SI domain for several reasons: (i) it has a clear formulation that makes it very easy to implement and hybridize, (ii) there are few parameters to adjust, and (iii) it is an origin of complicated and emergent phenomena, which are at the essence of swarm intelligence. Much of the success of PSO algorithms comes from the fact that individual particles have a tendency to diverge from the best known position in any given iteration, enabling them to ignore local optima while the swarm as a whole gravitates towards the global extremum.

Particle swarm optimization was introduced by Kennedy and Eberhart [14]. Since it was originally introduced, the algorithm has been studied by a number of different authors [8]. There are also theoretical study for convergence performance analysis [3]. During the last decade, many modifications of the original particle swarm optimization algorithm have been proposed [18]. The algorithm has been demonstrated to perform well for many static optimization problems [19] as well dynamic optimization problems [2].

In computer vision, visual tracking is the act of consistently locating a desired object in each image of an input sequence. It usually comprises an optimization process for estimating the motion of an object from measured images in a video sequence. In the optimization process either deterministic [4] or stochastic [11] methods can be used. Deterministic methods are usually fast but they can easily become trapped in local extremas. Stochastic methods are more robust, but the processing time is larger, especially in high-dimensional state spaces. Particle filters [6] are currently the state-of-the-art tool for estimating the hidden state of a dynamical nonlinear systems on the basis of observations available on-line. In such algorithms the state is represented by a set of particles, which are propagated and then updated recursively in order to approximate the state probability density function (pdf) of the system. A particle filter can be perceived as an importance sampler for the posterior density, using the predictive density on the state as the proposal distribution. However, the particle filter can fail if the observation likelihood is largely peaked compared to the prior or if it lies in the tail of the prior. To cope with this problem several methods were proposed in the literature. Recent algorithms employ mean-shift based optimization methods to bias the prior towards regions in the state space with the high likelihood [20]. For this purpose an approach relying on Particle Swarm Optimization has already been proposed in [21].

Some PSO-based approaches for object tracking were proposed recently. An algorithm [1] utilizes both the PSO algorithm and Kalman Filter in a hybrid framework of region and object tracking. Each object consists of several non-overlapped blocks and each block is represented by a particle. Another hybrid approach has been proposed in [15], where two trackers with different failure models are employed. The first tracker is based on several grayscale non-overlapping image patches that vote for the possible positions of the object, whereas the second one employs adaptive appearance models, which are used to construct a probabilistic observation model of a PSO-based particle filter.

In this paper we present an object tracking algorithm that is based on particle swarm optimization. At the beginning of each frame the particles are drawn from a Gaussian distribution to cover the promising object locations and afterwards the particle swarm optimization takes place to concentrate the particles in the proximity of the center of the object template. Owing to such an approach the resampling step is not needed. This significantly simplifies the tracking in comparison to particle filter based tracking.

The rest of the paper is organized as follows. Section 2. is devoted to particle swarm optimization. In the first part of section 3. the particle filtering is overviewed, whereas in second one a particle swarm optimization based particle filter is discussed. In section 4. we present particle swarm optimization based object tracking. The use of adaptive appearance models in our tracker is explained in section 5. In section 6. we report and discuss experimental results. Finally, some conclusions follow in the last section.

2. Particle swarm optimization

Particle swarm optimization is a stochastic, population-based evolutionary algorithm for solving nonlinear, multimodal optimization problems [14]. PSO shares several similarities with evolutionary computation techniques. The swarm is initialized with a population of individuals representing random solutions and then it searches for optima by updating particle values. The particles iteratively evaluate the fitness of the candidate solutions and remember the locations $\hat{\mathbf{x}}_i$, where they achieved their best fit so far. Another important value for the swarm is the location $\hat{\mathbf{g}}$ corresponding to the best fitness, which was obtained so far by any particle. Therefore, while exploring the hyperspace spanned by the possible parameters the particles employ reasoning capabilities about their own best location and the knowledge of the global best one. Through changing the velocity of each particle toward best $\hat{\mathbf{x}}_i$ and $\hat{\mathbf{g}}$ locations, the particle swarm finds $\hat{\mathbf{x}}^* \subseteq \mathcal{X} \subseteq \mathrm{R}^m$ such that $\hat{\mathbf{x}}^* = \arg\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) = \{\mathbf{x}^* \in \mathcal{X} : f(\mathbf{x}^*) \leq f(\mathbf{x})\}\ \forall \mathbf{x} \in \mathcal{X}$, and $f : \mathrm{R}^m \rightarrow \mathrm{R}$

Let there be N particles, each with associated locations $\mathbf{x}_i \in \mathrm{R}^m$ and velocities $\mathbf{v}_i \in \mathrm{R}^m$. The optimization algorithm takes the following form:

```
• Initialize $\mathbf{x}_i$ and $\mathbf{v}_i$. Then $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$, $\hat{\mathbf{g}} = \arg\min_{\mathbf{x}_i} f(\mathbf{x}_i)$
• While stopping criterion is not satisfied
•   For each particle
•     Update the particle locations: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
•         $\mathbf{v}_i \leftarrow \omega\mathbf{v}_i + c_1\mathbf{r}_1 \circ (\hat{\mathbf{x}}_i - \mathbf{x}_i) + c_2\mathbf{r}_2 \circ (\hat{\mathbf{g}} - \mathbf{x}_i)$        (1)
•     Update the local best values, if $f(\mathbf{x}_i) < f(\hat{\mathbf{x}}_i)$, $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$
•     Update the global best, if $f(\mathbf{x}_i) < f(\hat{\mathbf{g}})$, $\hat{\mathbf{g}} \leftarrow \mathbf{x}_i$
```

The symbol ω represents an inertial constant, $\circ$ stands for array multiply, c_1 and c_2 are positive constants, which balance the influence of the particle's local best (cognition part) and the global best (social part), respectively, whereas $\mathbf{r}_1$ and $\mathbf{r}_2$ are vectors of uniform random numbers in the range $[0, 1]$, which are generated according to a uniform probability distribution.

In [3] a constriction factor K has been introduced in the PSO algorithm. The velocity updating given by (1) has been replaced by the following equation:

$$\mathbf{v}_i = K[\mathbf{v}_i + c_1\mathbf{r}_1 \circ (\hat{\mathbf{x}}_i - \mathbf{x}_i) + c_2\mathbf{r}_2 \circ (\hat{\mathbf{g}} - \mathbf{x}_i)] \tag{2}$$

where

$$K = \frac{2}{|2 - \phi\sqrt{\phi^2 - 4\phi}|} \tag{3}$$

is a function of c_1 and c_2, and $\phi = c_1 + c_2$. Typically, when such a velocity updating is employed, ϕ is set to 4.1 (assuming that $c_1 = c_2 = 2.05$) and the constriction factor assumes the value 0.729. Both terms $(\hat{\mathbf{x}}_i - \mathbf{x}_i)$ and $(\hat{\mathbf{g}} - \mathbf{x}_i)$ are multiplied by $0.729 * 2.05$ times a random number between 0 an 1, with expected value equal 0.5.

The use of particle swarm optimization methods presents various advantages over traditional optimization methods. In particular, a few simple rules result in complex action of the algorithm. It has been demonstrated that in many problems PSO achieves better results in a faster way in comparison with other methods. PSO has been successfully applied in wide range of applications. Another reason that PSO is attractive is that it needs a few parameters to adjust. The classical particle swarm optimization has its own disadvantages, such as low convergence speed and prematurity. All of these make solutions prone to convergence into local extremas. Once the algorithm gets into the local extremum it is very hard to jump out from such local optimizations.

3. Particle filtering using particle swarm optimization

3.1. Particle filter

The effectiveness of object tracking in image sequences has been greatly improved with the development of particle filtering (PF). The particle filter is an algorithm for estimating the posterior state of a dynamic system over time where the state cannot be measured directly, but may be estimated at the current time-step t [6]. Particle filters are attractive for nonlinear models, multi-modal, non-Gaussian or any combination of these models for several reasons. They utilize imperfect observation and motion models and incorporate noisy collection of observations through Bayes rule. The ability to represent multimodal posterior densities allows them to globally localize as well as relocalize the object of interest in case of failure during tracking. Particle filters are any-time because by supervising the number of samples on-line they can adapt to the available computational resources.

Two important components of each particle filter are motion model $p(\mathbf{z}_t \mid \mathbf{z}_{t-1})$ describing the state propagation and observation model $p(\mathbf{y}_t \mid \mathbf{z}_t)$ describing the likelihood that a state $\mathbf{z}_t$ causes the observation $\mathbf{y}_t$. Starting with a weighted particle set $S = \{(\mathbf{z}_{n,t-1}, \pi_{n,t-1}) \mid n = 1, ..., N\}$ approximately distributed according to $p(\mathbf{z}_{t-1} \mid \mathbf{y}_{1:t-1})$ the particle filter operates through predicting new particles from a proposal distribution. To give a new particle representation $S = \{(\mathbf{z}_{n,t}, \pi_{n,t}) \mid n = 1, ..., N\}$ of the posterior density $p(\mathbf{z}_t \mid \mathbf{y}_{1:t})$ the weights of the particles are set to $\pi_{n,t} \propto \pi_{n,t-1} p(\mathbf{y}_t | \mathbf{z}_{n,t})$

$p(\mathbf{z}_{n,t} \,|\, \mathbf{z}_{n,t-1}) / q(\mathbf{z}_{n,t} \,|\, \mathbf{z}_{n,t-1}, \mathbf{y}_t)$. When the proposal distribution from which particles are drawn is chosen as the distribution conditional on the particle state at the previous time step, the importance function reduces to $q(\mathbf{z}_{n,t} \,|\, \mathbf{z}_{n,t-1}, \mathbf{y}_t) = p(\mathbf{z}_{n,t} \,|\, \mathbf{z}_{n,t-1})$ and in consequence the weighting function takes the form $\pi_{n,t} \propto p(\mathbf{y}_t \mid \mathbf{z}_{n,t})$. This simplification leads to a variant of a particle filter, namely CONDENSATION [11]. From time to time the particles should be resampled according to their weights in order to avoid degeneracy [7].

The particle filter converges to the optimal filter in some sense as the number of particles grows to infinity. The most important property of the particle filter is its ability to handle complex, non-Gaussian and multimodal posterior distributions. However, the number of particles required to adequately approximate the conditional density grows exponentially with the dimensionality of the state space. This poses practical difficulties in applications such as articulated body tracking [20]. In such applications, weakness of the particle filter consists in that the particles can not cluster around the true state of the object and instead they can migrate towards local maximas in the posterior distribution. If particles are too diffused the track of the object can be lost. When the observation likelihood lies in the tail of prior distribution, most of the particles will become insignificant weights. If the system model is inaccurate, the prediction based on the system model may not be a good one. Predictions of poor quality can also be caused by the simulation in the particle filtering, when the system noise is large and the number of particles is not sufficient.

In order to cope with the mentioned effects several improvements to basic algorithm have been proposed, among others solutions combining extended Kalman filter/unscented Kalman filter with generic particle filter [16]. Such filters incorporate the current observation to generate the better importance density than the generic particle filter, which uses the prior as the importance density. In recent work [21][15] PSO algorithms have demonstrated their great ability to improve the particle filtering through shifting particles towards their promising locations.

3.2. Particle filter using particle swarm optimization

PSO algorithm and PF employ different schemes to concentrate particles near best locations. PSO shifts particles towards the optimum by updating position and velocity of each particle using its best previous value and global best value of all particles. Particle filter utilizes general prediction-update framework in which probabilistic motion and observation models are used alternately during approximating the conditional density. The algorithm for swarm-based particle filter that has been employed in work [15] is as follows:

```
1. Initialization. Sample $\mathbf{z}_{1,0}, ..., \mathbf{z}_{N,0}$ i.i.d from initial density $p_0$
2. Importance Sampling/Propagation. Sample $\mathbf{z}_{i,t}$ from $p(\mathbf{z}_t|\mathbf{z}_{i,t-1})$, $i = 1, ..., N$
3. Initialize PSO. Assign initial values $\mathbf{v}_i$, then $\mathbf{x}_i \leftarrow \mathbf{z}_{i,t}$, $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$, $\hat{\mathbf{g}} = \arg\min_{\mathbf{x}_i} f(\mathbf{x}_i)$
   3a. While not stop
          For each particle
              Update the particle locations: $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
              Update particle velocities: $\mathbf{v}_i \leftarrow \omega\mathbf{v}_i + c_1\mathbf{r}_1 \circ (\hat{\mathbf{x}}_i - \mathbf{x}_i) + c_2\mathbf{r}_2 \circ (\hat{\mathbf{g}} - \mathbf{x}_i)$
```

```
            Update the local best values, if $f(\mathbf{x}_i) < f(\hat{\mathbf{x}}_i)$, $\hat{\mathbf{x}}_i \leftarrow \mathbf{x}_i$
            Update the global best, if $f(\mathbf{x}_i) < f(\hat{\mathbf{g}})$, then $\hat{\mathbf{g}} \leftarrow \mathbf{x}_i$
    3b. $\mathbf{z}_{i,t} \leftarrow \hat{\mathbf{x}}_i$
4. Updating. Compute $\hat{p}(\mathbf{z}_t|\mathbf{y}_{1:t}) = \sum_{i=1}^{N} \pi_{i,t}\delta(\mathbf{z} - \mathbf{z}_{i,t})$ using normalized weighs:
      $\pi_{i,t} = p(\mathbf{y}_t|\mathbf{z}_{i,t})$
      $\pi_{i,t} = \pi_{i,t} / \sum_{j=1}^{N} \pi_{j,t}$, $i = 1, ..., N$
5. Resampling. Sample $\mathbf{z}_{1,t}, ..., \mathbf{z}_{N,t}$ i.i.d from $\hat{p}(\mathbf{z}_t|\mathbf{y}_{1:t})$
6. $t \leftarrow t + 1$, go to step 2.
```

Such a combined filter outperforms the generic particle filter both in speed of computations and accuracy [15].

4. PSO-based object tracking

The tracking algorithm presented in this section relies only on particle swarm optimization. After a convergence of the PSO algorithm we are given the estimate of the object state. This means that the particles typically form a compact cluster centered at the true state. When a new image is available, the particles are propagated over time according to Gaussian model of the locomotion. This step spreads the particles apart in order to start a new exploration of the search space, resulting in the updated estimate of the object state. Owing to such an approach the resampling process is not required and the estimates of the object state can be determined through the usage of particle swarm optimization. The PSO algorithm seeks for the values $\hat{\mathbf{x}}^*$, which minimize the fitness function $f(\mathbf{x})$. Given a new image and the estimate of the object state, a random propagation of the particles according to Gaussian model takes place, and afterwards a new optimization is done, which results in an updated estimate of the object state. The mentioned process is repeated if a new image is available. This modification greatly simplifies the tracking algorithm, which relies on particle swarm optimizations only, and owing to propagation of particles over time according to Gaussian model of the locomotion, it still has a probabilistic nature. In the course of tracking it repeatedly shifts the particles to the promising locations and then spreads they at the expected object locations in order to start a new exloration. Assuming the temporal continuity between frames the object can be tracked over time. As we mentioned above no resampling is needed. Moreover, owing to sharing of information between particles about promising locations, the search space can be explored more efficiently in comparison to particle filter based exploration. The flowchart of the algorithm can be depicted in the following manner:

```
1. Initialize $t = 0$, $\mathbf{z}_{i,t}$ and $\mathbf{v}_{i,t}$.
2. $\hat{\mathbf{z}}_{i,t} \leftarrow \mathbf{z}_{i,t}$, $\hat{\mathbf{g}}_t = \arg\min_{\mathbf{z}_{i,t}} f(\mathbf{z}_{i,t})$
3. While stopping criterion is not satisfied
4.    For each particle
5.       Update the particle locations: $\mathbf{z}_{i,t} \leftarrow \mathbf{z}_{i,t} + \mathbf{v}_{i,t}$
6.       Update: $\mathbf{v}_{i,t} \leftarrow \omega\mathbf{v}_{i,t} + c_1\mathbf{r}_1 \circ (\hat{\mathbf{z}}_{i,t} - \mathbf{z}_{i,t}) + c_2\mathbf{r}_2 \circ (\hat{\mathbf{g}} - \mathbf{z}_{i,t})$
7.       Update the local best values, if $f(\mathbf{z}_{i,t}) < f(\hat{\mathbf{z}}_{i,t})$, $\hat{\mathbf{z}}_{i,t} \leftarrow \mathbf{z}_{i,t}$
```

```
8.     Update the global best, if $f(\mathbf{z}_{i,t}) < f(\hat{\mathbf{g}}_t)$, $\hat{\mathbf{g}}_t \leftarrow \mathbf{z}_{i,t}$
9.  For each particle
10.   $\hat{\mathbf{z}}^*_{i,t} = \mathbf{g}_t$
11.   $\mathbf{z}_{i,t+1} \sim \mathcal{N}(\hat{\mathbf{z}}^*_{i,t}, \Sigma_t)$
12.  $t \leftarrow t+1$, go to 2.
```

The standard deviation Σ_t is proportional to the predicted motion of the object. In the convergence criterion the algorithm checks if $f(\mathbf{g}_t) > th$, where th is a predefined threshold, and if the predefined part of the particles converged into an assumed basin region, centered on $\mathbf{g}_t$.

5. Visual appearance modeling using adaptive models

Let $I_{x,t}$ denote the brightness value at the location (x_1, x_2) in an image $\mathcal{I}$ that was acquired in time t. Let $\mathcal{R}$ be a set of J locations $\{x^{(j)} \,|\, j = 1, 2, ..., J\}$ defining a template. $Y_t(\mathcal{R}) = \{I^{(j)}_{x,t} \,|\, j = 1, 2, ..., J\}$ is a vector of the brightness values at locations $x^{(j)}$ in the template. Our intensity-based appearance model consists of three components, namely, the W-component expressing the two-frame variations, the S-component characterizing the stable structure within all previous observations and F component representing a fixed initial template. The model $A_t = \{W_t, S_t, F_t\}$ represents the appearances existing in all observations up to time $t-1$. It is a mixture of Gaussians [12] with centers $\{\mu_{i,t} \,|\, i = w, s, f\}$, their corresponding variances $\{\sigma^2_{i,t} \,|\, i = w, s, f\}$ and mixing probabilities $\{m_{i,t} \,|\, i = w, s, f\}$.

The update of the current appearance model A_t to A_{t+1} is done using the Expectation Maximization (EM) algorithm. For a template $\hat{Y}(\mathcal{R}, t)$ corresponding to the estimated state we evaluate the posterior contribution probabilities as follows:

$$o^{(j)}_{i,t} = \frac{m^{(j)}_{i,t}}{\sqrt{2\pi\sigma^2_{i,t}}} \exp\left[-\frac{\hat{I}^{(j)}_{x,t} - \mu^{(j)}_{i,t}}{2\sigma^2_{i,t}}\right] \tag{4}$$

where $i = w, s, f$ and $j = 1, 2, ..., J$. This prevents the slowly varying component from updating by background pixels. The posterior contribution probabilities (with $\sum_i o^{(j)}_{i,t} = 1$) are utilized in updating the mixing probabilities in the following manner:

$$m^{(j)}_{i,t+1} = \gamma o^{(j)}_{i,t} + (1-\gamma) m^{(j)}_{i,t} \quad | \, i = w, s, f \tag{5}$$

where γ is accommodation factor. Then, the first and the second-moment images are determined as follows:

$$M^{(j)}_{1,t+1} = (1-\gamma) M^{(j)}_{1,t} + \gamma o^{(j)}_{s,t} \hat{I}^{(j)}_{x,t} \tag{6a}$$

$$M^{(j)}_{2,t+1} = (1-\gamma) M^{(j)}_{2,t} + \gamma o^{(j)}_{s,t} (\hat{I}^{(j)}_{x,t})^2 \tag{6b}$$

In the last step the mixture centers and the variances are calculated as follows:

$$\mu^{(j)}_{s,t+1} = \frac{M^{(j)}_{1,t+1}}{m^{(j)}_{s,t+1}}, \quad \sigma^{(j)}_{s,t+1} = \sqrt{\frac{M^{(j)}_{2,t+1}}{m^{(j)}_{s,t+1}} - (\mu^{(j)}_{s,t+1})^2} \tag{7}$$

$$\mu_{w,t+1}^{(j)} = \hat{I}_{x,t}^{(j)}, \quad \sigma_{w,t+1}^{(j)} = \sigma_{w,1}^{(j)} \tag{8}$$

$$\mu_{f,t+1}^{(j)} = \mu_{t,1}^{(j)}, \quad \sigma_{f,t+1}^{(j)} = \sigma_{f,1}^{(j)} \tag{9}$$

In order to initialize the model A_1 the initial moment images are set using the following formulas: $M_{1,1} = m_{s,1} I(\mathcal{R}, t_0)$ and $M_{2,1} = m_{s,1}(\sigma_{s,1}^2 + I(\mathcal{R}, t_0)^2)$.

The fitness score has been evaluated according to the following equation:

$$f(\mathbf{x}_t) = \prod_{j=1}^{J} \sum_{i=w,s,f} \frac{m_{i,t}^{(j)}}{\sqrt{2\pi\sigma_{i,t}^2}} \exp\left[-\frac{I_{x,t}^{(j)} - \mu_{i,t}^{(j)}}{2\sigma_{i,t}^2}\right] \tag{10}$$

In the fitness function we utilize a recursively update appearance model, which depicts stable structures seen so far, initial object appearance as well as two-frame variations.

6. Experiments

In order to demonstrate the usefulness of the PSO algorithm in tracking we performed a simple experiment consisting in seeking for the extremum in a probability image, see Fig. 1c. The probability image expresses the similarity between the rectangular object template, see Fig. 1a and the image depicted at Fig. 1b. The object template has been moved over mentioned above image and the corresponding similarities have been calculated. The object window has been represented via the covariance matrix [22] of feature points. As features we employed pixel locations (x, y), RGB color values, and the norm of the first order derivatives of the intensities with respect to x and y. Each pixel has been converted to a seven-dimensional feature vector $\mathbf{h} = [x, y, R(x,y), G(x,y), B(x,y), |dI(x,y)/dx|, |dI(x,y)/dy|]^T$, where R, G, B are the RGB color values, and I is the intensity. The covariance of a region is a 7×7 matrix $\mathbf{c} = 1/(n-1)\sum_{k=1}^{L}(\mathbf{h}_k - \mu)(\mathbf{h}_k - \mu)^T$, where μ is the mean value and L denotes the number of pixels in the template. To measure the dissimilarity between the covariance matrixes $\mathbf{c}_1$ and $\mathbf{c}_2$ we employed the following distance $\rho(\mathbf{c}_1, \mathbf{c}_2) = \sqrt{\sum_{i=1}^{L} \ln^2 \lambda_i(\mathbf{c}_1, \mathbf{c}_2)}$, where $\{\lambda_i(\mathbf{c}_1, \mathbf{c}_2)\}_{i=1,\ldots,L}$ are the generalized eigenvalues of $\mathbf{c}_1$ and $\mathbf{c}_2$, which are calculated on the basis $\lambda_i \mathbf{c}_1 \mathbf{x}_i - \mathbf{c}_2 \mathbf{x}_i$, where $\mathbf{x}_i \neq 0$, $i = 1, ..., L$, are the generalized eigenvectors.

The image Fig.1d depicts the initial location of the particles, whereas the images Fig.1e-i depict the location of the particles in the course of the optimization. Fig.1i depicts the location of the most similar region to the object template. The algorithm converged at the proper location despite relatively small size of the object template, i.e. 11×13, peaked probability function, relatively large distance between starting location of the particles and the extremum. Similar behavior of the algorithm has been observed in many runs of the algorithm using this image as well as another probability images. In almost all runs the algorithm converged at proper location. In typical tracking experiments the motion of the object is smaller than in the considered experiment and the particles are typically located in proximity of the true location.

In order to investigate the usefulness of the adaptive appearance models for the construction of the fitness function of the PSO we conducted some experiments, see Fig. 2. The particle swarm optimization

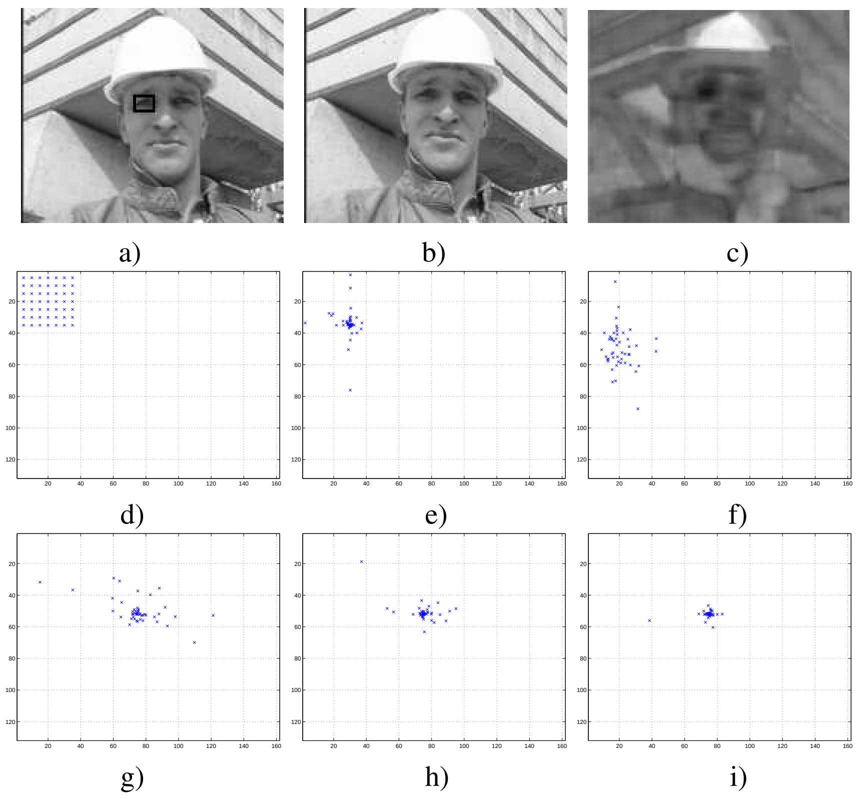

Figure 1. Template tracking using particle swarm optimization, a) *Foreman* test sequence, frame #1, b) frame #50, c) probability image, the optimization in d) 1-st it., e) 10-th it, f) 20-th it., g) 30-th it., h) 35-th it., i) 40-th iteration, respectively

based algorithm has been employed and the tracking has been done on gray images, which were acquired from a camera mounted at the top of the computer screen. The number of the particles has been set to 30. The experiments demonstrated that the algorithm can cope with temporal occlusions. In the discussed experiment the location of the tracking window has not been shifted considerably and thus only the part of the template pixels learned the occluding object. Similar effect has been observed in other experiments devoted to testing the robustness of the algorithm during occlusions. In all experiments the template was parameterized by the vector $[x, y, \alpha, \beta, \theta, s]^T$ in which the pair x, y stands for the template translation and the remaining parameters characterize template deformations.

As we already mentioned, the results depicted at Fig. 1 demonstrated a great ability of the particle swarm optimization algorithm in exploring the search space. In order to verify this capability in practice we performed several experiments consisting in tracking an object with rapid motion. Some experimental results are depicted at Fig. 3, where a considerable change of object appearance can be observed too. Despite such untoward conditions the algorithm was able to track the object with arbitrary motion. The particle filter built on particle swarm achieves similar tracking performance, but due to resampling process the computational load is far larger.

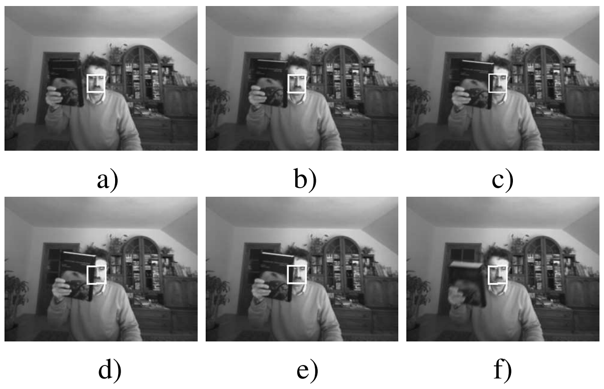
Figure 2. Particle swarm optimization based tracking of face undergoing temporal occlusions, frames #10 a), #15 b), #20 c), #25 d), #27 e), #28 f)

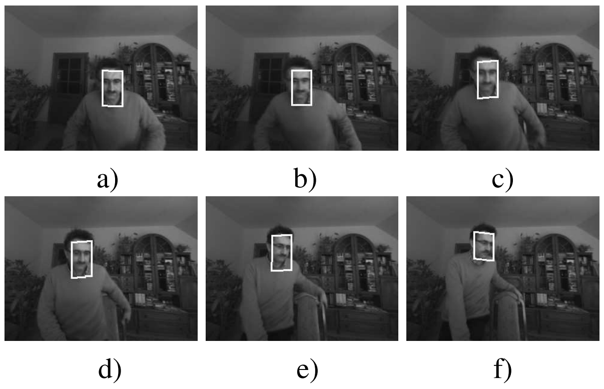
Figure 3. Particle swarm optimization based tracking of face with arbitrary motion, frames #17 a), #18 b), #19 c), #20 d), #21 e), #22 f)

The maximal number of iterations in the PSO algorithm has been set to 5. In order to overcome the swarm explosion we heuristically maintained the velocity in the particle swarm optimizations. In particular, the maximal value of constriction factor K in (2) has been accommodated according to predictions of the object motion. If the current particle velocity exceeds the predicted particle velocity, the value of K is proportional to ratio between predicted velocity and the current velocity.

In general, deterministic methods require substantial overlap between object windows in the consecutive frames. On the other hand, the particle filter typically requires a significant number of particles to cover the regions of the high likelihood, especially in high-dimensional state-spaces. In Fig. 4 we can observe an undesired behavior of the particle filter based tracker. The tracker failed because of the particle impoverishment, i.e. loss of the diversity of the particles. Larger number of particles should be sampled to deal with this undesirable effect. However, this would cause additional computational overheads. In the discussed experiment the number of particles has been set to 600.

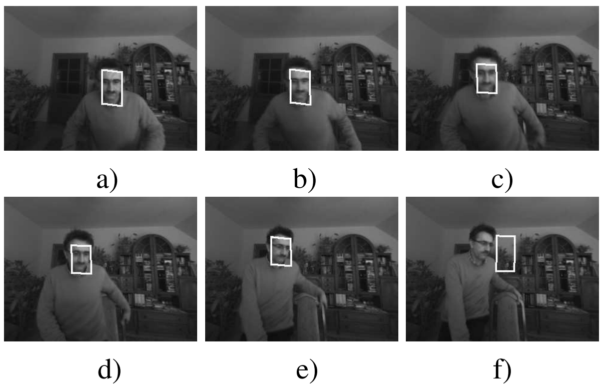

Figure 4. Particle filter based tracking of face with arbitrary motion, frames #17 a), #18 b), #19 c), #20 d), #21 e), #22 f)

We tested the particle swarm optimization tracker on various image sequences, among others Caviar[1], *Foreman*, *Carphone*. The experimental results presented above are representative, because of rapid motion and considerable appearance changes of the object undergoing tracking. In such sequences the particles employ the power of reasoning about their own best location and the knowledge of the global best one. In image sequences, in which the objects have low motion, e.g. Caviar, the PSO-based tracker and particle filter built on PSO demonstrated similar behavior. The MSE of position of the particle filter is larger in comparison to the mentioned above algorithms. Some results comparing the ordinary particle filter with particle filter based on PSO can be found in [15].

7. Conclusions

We have presented a particle swarm optimization based algorithm for object tracking. In each frame the particles are drawn from a Gaussian distribution in order to cover the promising object locations and afterwards the particle swarm optimization takes place in order to concentrate the particles near the true object state. Owing to propagation of the particles according to Gaussian model of the locomotion the resampling of particles is not needed. The tracker built on particle swarm optimization demonstrated its robustness in experiments consisting in tracking object with arbitrary motion as well as under considerable appearance changes.

References

[1] Akbari, R., Jazi, M. D., Palhang, M.: A Hybrid Method for Robust Multiple Objects Tracking in Cluttered Background, *Proc. of 2nd Int Conf. on Information and Communication Technologies*, IEEE, Piscataway, NJ, 2006.

[2] Blackwell, T. M., Branke, J.: Multiswarms, explosion, and anti-convergence in dynamic environments, *IEEE Trans. on Evolutionary Computation*, **10**(4), 2006, 459–472.

[1]Downloaded from site at: http://groups.inf.ed.ac.uk/vision/CAVIAR/

[3] Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space, *IEEE Trans. on Evolutionary Computation*, **6**(1), 2002, 58–73.

[4] Comaniciu, D., Ramesh, V., Meer, P.: Real-time tracking of non-rigid objects using mean shift, *Proc. Int. Conf. Comp. Vision Pattern Recognition*, 2000.

[5] Dempster, A., Laird, N., Rubin, D.: Maximum Likelihood from Incomplete Data via the EM Algorithm, *Statistical Society, Series B*, **39**(1), 1977, 1–38.

[6] Doucet, A., Freitas, N., Gordon, N.: *Sequential Monte Carlo Methods in Practice*, Springer-Verlag, New-York, 2001.

[7] Doucet, A., Godsill, S., Andrieu, C.: On sequential Monte Carlo sampling methods for bayesian filtering, *Statistics and Computing*, **10**(1), 2000, 197–208.

[8] Engelbrecht, A. P.: *Fundamentals of Computational Swarm Intelligence*, Wiley, 2005.

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

[10] Horn, B. K. P.: *Robot Vision*, The MIT Press, 1986.

[11] Isard, M., Blake, A.: Contour Tracking by Stochastic Propagation of Conditional Density, *European Conf. on Computer Vision*, Cambridge, 1996.

[12] Jepson, A. D., Fleet, D. J., El-Maraghi, T.: Robust on-line appearance models for visual tracking, *Int. Conf. on Comp. Vision and Pattern Rec.*, 2001.

[13] Jepson, A. D., Fleet, D. J., El-Maraghi, T.: Robust on-line appearance models for visual tracking, *PAMI*, **25**(10), 2003, 1296–1311.

[14] Kennedy, J., Eberhart, R.: Particle swarm optimization, *Proc. of IEEE Int. Conf. on Neural Networks*, IEEE Press, Piscataway, NJ, 1995.

[15] Kwolek, B.: Object Tracking Using Grayscale Appearance Models and Swarm Based Particle Filter, *Int. Conf on Hybrid Artificial Intelligence Systems*, Lecture Notes in Artificial Intelligence, Springer-Verlag Berlin Heidelberg, Burgos, Spain, 2008.

[16] van der Merve, R., de Freitas, N., Doucet, A., Wan, E.: The Unscented Particle Filter, *Advances in Neural Information Processing Systems*, **13**, 2001, 584–590.

[17] Nabney, I.: *Netlab. Algorithms for Pattern Recognition*, Springer-Verlag, London, Berlin, Heidelberg, 2002.

[18] Pant, M., Thangaraj, R., Abraham, A.: New Particle Swarm Optimization Algorithm Incorporating Reproduction Operator for Solving Global Optimization Problems, *Proc. of the 7th Int. Conf. on Hybrid Intelligent Systems*, IEEE, Piscataway, NJ, 2007.

[19] Parsopoulos, K. E., Vrahatis, M. N.: Recent approaches to global optimization problems through particle swarm optimization, *Neural Computing*, **q**(2–3), 2002, 235–306.

[20] Schmidt, J., Fritsch, J., Kwolek, B.: Kernel particle filter for real-time 3D body tracking in monocular color images, *IEEE Int. Conf. on Face and Gesture Rec.*, *IEEE Computer Society Press*, Southampton, UK, 2006.

[21] Tong, G., Fang, Z., Xu, X.: Particle Swarm Optimized Particle Filter for Nonlinear System State Estimation, *IEEE Congress on Evolutionary Computation*, IEEE, Piscataway, NJ, Vancouver, BC, 2006.

[22] Tuzel, O., Porikli, F., Meer, P.: Region Covariance: A Fast Descriptor for Detection and Classification, *European Conference on Computer Vision*, Lecture Notes in Artificial Intelligence, vol. 3952, Springer-Verlag Berlin Heidelberg, Graz, Austria, 2006.

[23] Zhou, S. K., Chellappa, R., Moghaddam, B.: Appearance tracking using adaptive models in a particle filter, *Proc. Asian Conf. on Comp. Vision*, 2004.