

*Chapter 9*

# **OBJECT DETECTION AND TRACKING USING BAYES- CONSTRAINED PARTICLE SWARM OPTIMIZATION**

*Yuhua Zheng<sup>1</sup> and Yan Meng*

Department of Electrical and Computer Engineering  
Stevens Institute of Technology, Hoboken, NJ 07030, USA

## **ABSTRACT**

In this Chapter, we present a new face detection and tracking algorithm using Bayes-constrained particle swarm optimization (BC-PSO), which is a population based searching algorithm. A cascade of boosted classifiers based on Haar-like features is trained and employed for object detection. Then the PSO-based algorithm is applied for object tracking. Basically the searching can be divided into two steps in this method. First, the object model is projected into a high-dimensional feature space, and a PSO algorithm is applied to search over this high-dimensional space and converge to some global optima, which are well-matched candidates in terms of object features. Second, a Bayes-based filter is used to identify the one with the highest possibility among these candidates under the constraint of object motion estimation. The proposed algorithm considers not only the object features but also the object motion estimation to speed up the searching procedure. Experimental results demonstrate that the proposed method is efficient and robust under dynamic environment.

## **1. INTRODUCTION**

In computer vision, object detection and tracking is an active research area which has attracted extensive attentions from multi-disciplinary fields, and it has wide applications in many fields like service robots, surveillance systems, public security systems, and virtual reality interfaces. Detection and tracking of moving object like car and people are more

---

<sup>1</sup> [yzheng1@stevens.edu](mailto:yzheng1@stevens.edu).

concerned, especially flexible and robust tracking algorithms under dynamic environments, where lightening condition may change and occlusions may happen.

The general process of object detection consists of two steps. The first step is building models. According to the prior knowledge of the interested objects, the feature model is built up to describe the target object and separate it from other objects and backgrounds. And since most images are noisy, statistic information are usually adopted to quantify features. The second step is to find a particular region in the image, called area of interest (AOI), which either can best fit the object model or have the highest similarity with the model.

If the detection is executed successfully for each frame of a image sequence, it means that the object is tracked, where every frame is treated independently. Another category of tracking algorithms takes advantage of correlations between image frames to accelerate tracking process. Basically, the features of the object itself are local information, and the features of a image sequence belong to global information. For instance, movement correlations include predicted object heading and velocity.

Extensive works have been conducted for object detection and tracking. Most available algorithms focus on estimating movements of AOI using probabilistic theories. Some popular models and approaches, like HMM (Hidden Markov Model), Kalman filter [1], condensation [2], and particle filter [3] predict discrete probability distributions, while other algorithms, for example, mean shift methods [4] [5] study how to search the object model in a more robust manner. For Kalman filter based methods, some researchers proposed different control and noise models into the recursion function; however those assumptions are dependent on specific applications and need to be tuned carefully. Condensation and Particle filter methods mainly focus on how to sample probability and likelihood, so as to represent simultaneous alternative hypotheses of the object which could not be handled by Kalman filter. Generally speaking, the mean-shift method, as a well-known kernel-based local searching algorithm, is efficient for object tracking. However, the searching window may drift away from the object under dynamic conditions. For example, if the kernel is lost from the tracked target in one frame under some emergent situations, such as the changes in illumination condition, it would be difficult for the tracker to recover itself from this unpredicted event.

Some other methods have also been proposed. Wren et al. [6] proposed a Pfinder real-time system for people tracking, where a multi-class statistical model of color and shape was proposed to segment people from background scenes. Viola and Jones [7] proposed a boosted cascade algorithm, which is a learning method by combining a set of weak 0/1 classifiers to quickly detect objects. Olson and Brill [8] built a general-purpose system for moving object detection and event recognition, where objects were detected and tracked by both first-order prediction and nearest neighbor matching. The system proposed in [9] extracted moving targets from a real-time video stream, and classified them into pre-defined categories and tracked them. Tao et al. [10] tried to build configurations for each target to naturally handle appearance, disappearance and occlusion. Beleznaï et al. [11] adopted the fast mean shift to cluster and track people. In the Hydra system [12], a silhouette-based shape model and the correlation-based matching method were combined together to conduct classification.

In this Chapter, we will focus on using particle swarm optimization (PSO) based method for object tracking. Particle Swarm Optimization (PSO) was proposed by Kennedy and Eberhart in 1995 [13] [14] from the simulation of a simplified social model, which has its root in bird flocking, fish schooling and swarming theory particularly. This population-based technique involves simulating social behavior among individual particles “flying” through a

multidimensional search space, each particle representing a single intersection of all search dimensions. And with the movements of particles, a set of potential solutions evolve to approach an optimal solution for a problem. Being an optimization method, the aim is to find the global optimum of a real-valued fitness function defined in a given search space. Rather than just being a social simulation, PSO can be treated as a powerful new search algorithm, capable of optimizing a wide range of N-dimensional problems.

The social metaphor that leads to this algorithm can be summarized as follows: the individuals that are part of a society hold an opinion that is part of a "belief space" (the search space) shared by neighboring individuals. Individuals may modify this "opinion state" based on three factors: (1) the knowledge of the environment (inertia part); (2) the individual's previous history of states (individual part); and (3) the previous history of states of the individual's neighborhood (social part). An individual's neighborhood may be defined in several ways, configuring somehow the "social network" of the individuals. Following certain rules of interaction, the individuals in the population adapt their scheme of belief to the ones that are more successful among their social network. Over the time, a culture arises, in which the individuals hold opinions that are closely related.

In the PSO algorithm, each individual is called a "particle", and is subject to the movement in a multidimensional space that represents the belief space. Particles have memory, thus retaining part of their previous states. There is no restriction for particles to share the same point in belief space, but in any case their individuality is preserved. Each particle's movement is the composition of an initial random velocity and two randomly weighted influences: individuality, the tendency to return to the particle's best previous position, and sociality, the tendency to move towards the neighborhood's best previous position.

The velocity and position of the particle at any iteration is updated based on the following equations:

$$v_{id}^{t+1} = w \cdot v_{id}^t + c_1 \cdot \varphi_1 \cdot (p_{id}^t - x_{id}^t) + c_2 \cdot \varphi_2 \cdot (p_{gd}^t - x_{id}^t) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^t + v_{id}^{t+1} \quad (2)$$

where  $v_{id}^t$  is the component in dimension  $d$  of the  $i$ th particle velocity at iteration  $t$ ,  $x_{id}^t$  is the component in dimension  $d$  of the  $i$ th particle position at iteration  $t$ ,  $c_1, c_2$  are constant weight factors,  $p_{id}^t$  is the best position achieved by particle  $i$ ,  $p_{gd}^t$  is the best position found by the neighbors of particle  $i$ ,  $\varphi_1, \varphi_2$  are random factors in the  $(0,1)$  interval, and  $w$  is the inertia weight. The PSO requires tuning of some parameters: the individual and sociality weights  $c_1, c_2$ , and the inertia factor  $w$ . According to (1), each particle adjusts its velocity by combing three behaviors: keeping the velocity of last moment, moving to the best position from its own memory, and moving to the best position found by its neighbors. Different parameters in (1) provide various balances among those three factors. Then a particle moves in the search space according to the combined velocity calculated by (1) to achieve a new position, which presents a new solution.

During particles' movements, every visited point will be evaluated by a specific fitness function. And those points with highest fitness are assigned as the best positions. Then particles keep moving around until some stop conditions are met, such as reaching the maximum of iterations or a predefined threshold of the fitness value, or no more improvement can be made. In [15], a general idea of PSO has been explained, where, unlike hill-climbing, particles jump along the surface, which leads to a significant expedition in searching.

Since the first introduction of the basic PSO in 1995, many works have been conducted to improve the performance and to extend applications of PSO. According to the structure of the swarm, there are two categories called *gbest* and *lbest* PSO [16], related to global best and local best separately. For *gbest* PSO, all particles equally share the social information, which means that the neighborhood is the entire swarm. While in *lbest* PSO, the swarm consists of local neighborhoods and only particles of the same neighborhood can share information. Many topologies of neighborhoods can be used, like star, ring, wheel and so on. The *lbest* PSO reflects local information of the environment.

Initially, most proposed PSO methods deal with single solution. Some problems may have more than one global optimum or both global and local optima need to be located. Therefore, some variations have been developed to deal with particular problems with multiple solutions. Niching algorithms [17] have been proposed to locate multiple solutions. Multi-objective optimization with particles swarms, called MOPSO, was also developed to solve the problems that require the simultaneous optimization of a number of objectives [18]. Many PSO algorithms [19] have been developed to solve optimization problems under dynamic environments instead of static ones. Binary PSO and other discrete PSO try to handle optimization problems in discrete space, for example the traveling salesman problem [20].

PSO algorithms have been applied to many problems, including neural network training, design optimization, scheduling and image or data clustering. One of the first applications of PSO is the neural network training [21][22]. The general process is to develop a fitness function, while each particle presents a set of weights that need to be adjusted. With evolvments of particles, the optimized weights of the neural network are gained. Compared with other training methods like gradient-descent, the PSO-based methods are more robust and faster. Sousa *et al.* [23] used PSO for data mining, where each particle represents a single rule with a set of attributes. Rasmussen and Krink [24] used PSO to train Hidden Markov Model for multiple sequence alignment in biology. PSO was also applied to predict stock prices in financial companies by building the PSO-based social programming model [25]. Some clustering problems in the field of image processing are transformed into optimization problems and various PSO-based algorithms have been applied. In [26], PSO was used to tune the parameters embedded in the relevance evaluation for Content-Based Image Retrieval (CBIR) system. The ranking of the retrieved images are improved after using PSO. Omran *et al.* [27] used a PSO-based method for image classification, where each particle represented a clustering result and specific distance metric were used to measure the goodness of solutions.

## 2. RELATED WORK USING PSO-BASED METHODS

Some tracking methods adopt PSO into their own frameworks as supporting or accelerating tools, like tuning weights for neural networks. However, as a search algorithm, PSO itself can be applied directly for detection and tracking. In [28], PSO was combined with feature-based object classification. Every particle was treated as a self-contained classifier with different parameters to detect people in infrared images. Those classifiers swarmed in the solution space to converge to the optimal analysis window. Sequential niching technique was used to handle multiple objects. After searching, the swarm could find positions and sizes of objects and optimize the classifier parameters. Then in [29], this idea was extended and the PSO-based classifiers were embedded into a 3D framework to detect pedestrian from independently multiple views.

In [30], a PSO algorithm was proposed as a prey-predator scheme for real-time object tracking. Image pixels, as preys, are characterized by their scent intensity that reflects image features. Particles, as predators, fly over image pixels and hunt for preys that are interested to them. The rules of particle movements are specifically defined according to interactions between particles and their environment. During the tracking, particles try to keep themselves in a group as well as be adaptive with images changes.

Akbari *et al.* [31] employed both the PSO algorithm and Kalman filter in a hybrid framework of region and object tracking, where vehicles were tracked in a cluttered background. PSO was applied to conduct template matching and optimization recognition for photo time-stamp [32]. In [33], PSO was used to optimize elastic bunch graph matching technique to improve the performance of face recognition. A hybrid PSO algorithm was proposed in [34], which incorporated initial user guidance for single-slice 3-D-to-3-D biomedical image registration. By overcoming drawbacks of general local optimization methods, the proposed hybrid method produced more accurate registrations. Scheutz [35] introduced a hierarchical extension to the standard PSO to cope with dynamically changing fitness space, and applied it for dynamic face detection.

### **3. A BAYES-CONSTRAINED PSO (BC-PSO) ALGORITHM**

Usually object detection focuses on a single image to search for the object. It is difficult to build a simple model to describe the object, especially under dynamic environment. So offline-learning is widely used to improve the accuracy. In this section, we will use an offline-learning method, where a set of cascade classifiers are trained to learn features of the object and conduct detection.

For the tracking problem, given the AOI as well as the feature model, the goal is to search for the best fit region in the new image, which has the highest similarity with the object model. For the tracking, real-time performance and robustness are critical. To accelerate the searching procedure, both object features and the expectation of movements can come to play, which leads to two different ways for tracking. One is feature-based searching and the other is knowledge-based estimation. In this section, a combined framework of Bayes-based estimation and PSO-boosted searching is built up to achieve the system efficiency and robustness. The Bayes estimation constrains the searching space according to the motion vectors acquired from the previous frames. Then, the PSO-based searching method identifies the size and position of the AOI tracking window in a high-dimensional feature space.

### 3.1. Object Detection

A set of cascade classifiers with the ability of learning are trained and applied to detect objects. Firstly, a classifier with a set of parameters is built up based on the knowledge of the interest object. Then both positive and negative sample data are fed into the classifier to adjust those parameters. There is a mapping between the object and the classifier. For complex objects, multiple classifiers may have to be integrated, which is called cascade classifiers or boosted classifiers. The basic idea of these cascade classifiers is that several weak classifiers are used to cover different features of the object and combined to reach a better classification globally.

The widely used Haar-like features are a set of simple features including edge, line, and center-surround constructions [36]. Each feature is described by a template with size and scale factor. In real applications, hundreds of features and fast computation methods are used to calculate a decision value for each feature, where different weights are assigned to all of the decision values. And those decision values are fed into the complicated decision tree to make the final classification. OpenCV [37] employs this algorithm for the face detection. Since the features are independent of objects, they can also be applied for other objects as well, such as vehicles. Therefore, this detection method is adopted here for object detection.

### 3.2. Object Tracking

#### 3.2.1. PSO-Based Searching

When an object is detected, a tracking window is marked to highlight the AOI. The location and size of the window reflect information of the object. The window is also associated with the feature model of the object. Suppose an object is detected at frame  $t$ , its feature model can be expressed as  $H(t) = H(x, y, l, w, t)$ , where  $(x, y)$ ,  $l$ ,  $w$  represent the location, length, and width of the tracking window, respectively.  $H$  function contains feature information and is usually expressed by histograms. When a new frame at  $t+1$  comes, there are many possible regions or windows in the new image with different locations and sizes. First, we set a group of particles as followings, where each particle represents one possible window:

$$P = \{p_i \mid p_i(x_i, y_i, l_i, w_i), i = 1, 2, \dots, N\} \quad (3)$$

Where  $(x_i, y_i)$  represent the central point of the rectangle related to particle  $i$ ;  $l_i$  and  $w_i$  represents the length and width related to particle  $i$ , respectively; and  $N$  is the population of the swarm. Each individual particle is associated with a feature model:

$$H(t+1) = \{H(p_i, t+1) \mid i = 1, 2, \dots, N\} \quad (4)$$

So the objective of tracking is to find a region with  $\hat{H}(\hat{p}, t+1) \in H(t+1)$  so that  $\hat{H}(\hat{p}, t+1)$  has the highest similarity with the previous object model  $H(x, y, l, w, t)$ . The

measurement of similarities between object models can be defined as the fitness function of PSO, which will be discussed later.

Obviously, brute search is a straightforward method. However, it is very time consuming. Some methods take advantage of the gradient of feature planes to find the best fit. However, due to the variance of images, the gradient-descent methods are not always working well and may fail at local maxima. On the other hand, PSO-based searching algorithms can balance exploration and exploitation through the weight parameters of cognitive and social components. Each individual particle represents a different tracking window, which has different values of parameters for window size and location. These parameters build up a four-dimensional searching space, as shown in Figure 1, where every point in this space has an object model  $H$  which represents a potential solution for tracking. Therefore, instead of searching for object features in the image itself, the PSO-based method searches in this four-dimensional searching space.

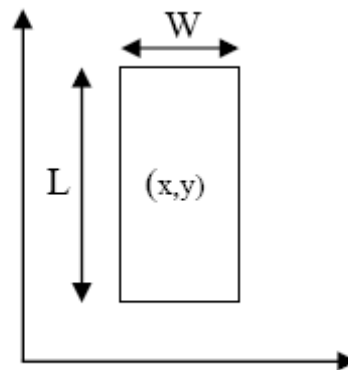


Figure1. The Four-Dimensional Space

Considering the correlations between image frames, object positions of consecutive frames should be close to each other. So the PSO search can be executed in a constrained small region instead of the whole space of an image frame, which can expedite the tracking and secure the efficiency. Based on previous information  $H(x, y, l, w, t)$ , the initial search region can be constrained as  $\{x \pm \Delta x, y \pm \Delta y, l \pm \Delta l, w \pm \Delta w\}$ . The shifting volumes depend on specific problems, for instance, the size of the object, or the movement velocity.

Following the rules of (1) and (2), particles start to move in the solution space. At each iteration, each particle calculates the fitness value according to its current position, updates its velocity, and move to next position. A total of  $N$  updates are performed per iteration, where  $N$  is the population of the swarm. This process is automatic and independent on knowledge of image contents. To accelerate convergence, the weight of inertia component  $w$  can be reduced step by step. Since  $w$  presents the ability of exploration of particles. When  $w$  goes down, particles are more attracted by founded solutions and are reluctant in exploring new area. Similarly, the weight of cognitive component  $c_1$  can also be reduced if the swarm focuses more on the global best. More detailed discussion of parameter tuning can refer to [4].

Particles keep moving around until some stop conditions are met. The stop conditions of iterations can be different, such as reaching the maximum number of iterations, reaching a predefined goodness of the fitness value, or no more improvement can be observed. In practice, those conditions can be used together or independently. In most experiments, a good result can be approached after 6 or 7 iterations. When iterations are over, particles cluster around one or several optimal points in the space, which present several regions in the image with varied locations and sizes. These regions are good candidates for the Bayes-based estimation, which will be discussed in the later section.

### 3.2.2. Fitness Function

As mentioned above, a feature model is constructed as the fitness function to drive the particles in PSO-based method. Most commonly-used features, such as color, shape, texture, and structure, can be employed independently or combined together. Among features, color might be the most popular one and it is independent with objects, view point and scale. The major drawback of color is the affection of lightening condition and physical characters of cameras. We will use histogram to measure these features.

First, images are transformed from RGB format into HSV format, which is more natural for people's eyes. Then, the values of hue are abstracted to build the histogram. Since hue specifies the dominant wavelength of the color, such histogram refers to the gradation of color within the visible spectrum. When a PSO-based searching algorithm is applied, each particle at every moment is associated with such a histogram expressed in (3) and (4). The best matched ones can be obtained by comparing these histograms with the target histogram. Therefore, a special criterion is required to measure the similarity between the searched window and the target window.

In statistics [38], the Bhattacharyya Coefficient is used to measure the similarity of two discrete probability distributions. It is a divergence-type measure that can be seen as the scalar product of the two vectors having components as the square root of the probability of the points  $x$ . It thereby lends itself to a geometric interpretation: the Bhattacharyya Coefficient is the cosine of the angle enclosed between these two vectors. Therefore, the Bhattacharyya Coefficient can be used to measure the similarity between these two histograms, which is defined as:

$$BC(H(t), H(p_i, t+1)) = \sum_{x \in X} \sqrt{H_x(t) H_x(p_i, t+1)} \quad (5)$$

where  $H(p_i, t+1)$  represents the histogram of particle  $i$ ,  $H(t)$  represents the histogram of the object, and  $X$  denotes the distribution domain, which is the range of hue values from 0 to 255.  $H_x(p_i, t+1)$  and  $H_x(t)$  are pixel numbers with a specific hue value  $x$  for the particle and target, respectively. By using (5), the distance between two histograms can be defined as [39]:

$$D(H(t), H(p_i, t+1)) = \sqrt{1 - BC(H(t), H(p_i, t+1))} \quad (6)$$

This distance is invariant to the scale of the target, while the popularly-used histogram intersection is scale variant. The smaller this distance is, the better the particle matches with

the target object. Thus, given the target histogram, the fitness function for particle  $i$  should be inversely proportional to the distance between  $H_x(p_i, t+1)$  and  $H_x(t)$  :

$$F(p_i, t+1) = 1/D(H(t), H(p_i, t+1)) \quad (7)$$

The higher the fitness value, the more similar the corresponding area is with the object. If the tracked object is occluded partially by other subjects, the proposed PSO-based tracking window would eventually converge to the non-occluded part of the object since the histogram value is independent of the window size. For the same reason, when the occlusion is released, the object can be recovered to its original window size.

### 3.2.3. Bayes-Based Estimation

After the PSO searching, more than one optimum may be found. Then Bayes-base estimator can be used as a filter to identify the best-fit result. During the tracking, the object has a motion trajectory that consists of a series of motion vectors from one frame to another as  $V(t)$ .  $V(t)$  can be calculated according to locations of consecutive tracking windows. The motion vector should be set up as zero in the first frame. For other frames, it can be represented by the shift from the previous position to the current one. Respect to previous object model  $H(t)$ , particles have a set of motion vectors as  $\{V(p_i, t+1), i=1,2,\dots,N\}$ , where  $V(p_i, t+1)$  represents motion vectors of particle  $i$ , and  $N$  represents the total number of particles.

Given the previous tracking window associated with the target histogram and the motion vector  $\{H(t), V(t)\}$ , the PSO-based searching algorithm can produce a set of candidate windows, which can be represented by  $\{H(p_i, t+1), V(p_i, t+1) | i=1,2,\dots,m\}$ , where  $H(p_i, t+1)$  represents histograms of particle  $i$ ,  $V(p_i, t+1)$  represents motion vectors of particle  $i$ , and  $m$  is the number of the selected candidates that is usually less than  $N$ . All of these candidate windows are good enough in terms of appearance features since their fitness values are higher than a preset threshold. According to the Bayes law, the estimation of the best fit can be described as:

$$p(H(p_i, t+1), V(p_i, t+1) | H(t), V(t)) = \frac{p(H(t), V(t) | H(p_i, t+1), V(p_i, t+1)) p(H(p_i, t+1), V(p_i, t+1))}{p(H(t), V(t))} \quad (8)$$

where  $p(H(p_i, t+1), V(p_i, t+1) | H(t), V(t))$  represents the condition probability of particle  $i$ .  $p(H(t), V(t))$  represents the probability of the target window, which is the same for all particles.  $p(H(t), V(t) | H(p_i, t+1), V(p_i, t+1))$  represents the back projection from candidates to the previous tracking window. Since all of particles can point back to the target window in different ways, it is hard to tell which particle is the most possible one without any predefined knowledge of the image environment. It is simply assumed that all  $p(H(t), V(t) | H(p_i, t+1), V(p_i, t+1))$ , for  $i=1,2,\dots,m$ , are equal. However this assumption may not hold in some practical applications, for instance, a mobile vision system,

where the previous motion trajectory of the mobile platform would provide more information for the back projection.

Considering that the PSO-based searching algorithm returns candidates which are good enough in appearance histogram, it is reasonable to ignore the histogram here and simplify (8) as:

$$p(V(p_i, t+1) | V(t)) = cp(V(p_i, t+1)) \quad (9)$$

where  $c$  is a positive constant factor, and  $p(V(p_i, t+1))$  represents the probability of a particle on the motion trajectory. According to the inertia of motion,  $p(V(p_i, t+1))$  depends on the distance between  $V(p_i, t+1)$  and  $V(t)$ . The closer two vectors are, the higher the possibility of the corresponding particle has. Then formula (9) turns into:

$$p(V(p_i, t+1) | V(t)) = cp(V(p_i, t+1)) = k / D(V(p_i, t+1), V(t)) \quad (10)$$

where  $k$  is a positive factor. If two vectors are shifted to the same original point, the distance between two vectors turns into the distance between two points, where Euclidean distance can be calculated.

#### 3.2.4. BC-PSO Algorithm Summary

The Bayes-Constrained PSO (BC-PSO) algorithm for object tracking can be summarized by the following pseudo codes.

```

while (a new image arrives)
  {if (no object detected)
    {run the cascade classifiers to detect objects;
     if (new object detected)
       {build up object model  $H(t) = H(x, y, l, w, t)$ ;
        build up object motion vector  $V(0) = 0$ ;
        change system state to tracking;
       }
     else go back to while for next image
    }
  else
    {initialize particles  $P = \{p_i | p_i(x_i, y_i, l_i, w_i), i = 1, 2, \dots, N\}$ ;
    while (stop condition no met)
      {for (each particle in the swarm)
        {calculate fitness  $F(p_i, t+1) = 1 / D(H(t), H(p_i, t+1))$ ;
         update particles states by (1) and (2);
         adjust weights  $w, c_1, c_2$ ;
        }
      }
    get good candidates  $\{H(p_i, t+1), V(p_i, t+1) | i = 1, 2, \dots, m\}$ ;
  }

```

```
Bayes filter executes  $p(V(p_i, t + 1) | V(t)) = k / D(V(p_i, t + 1), V(t))$ ;  
Object is tracked; update  $H(t)$  and  $V(t)$ ;  
go back to while for next image  
}  
} end
```

## 4. EXPERIMENTAL RESULTS

To verify the robustness and efficiency of the proposed algorithm, it has been applied for several video clips under dynamic environment, which includes various situations like partially or completely occlusion, disappearance, reappearance. Some experimental results are conducted, as shown in Figure 2, Figure 3, and Figure 4. The algorithm is written in C++ using OPENCV library. Figure 2 shows the procedure of the proposed PSO searching for candidate windows. In the left picture, a number of particles are distributed, then move around and eventually converge. The new tracker is showed in the right picture.

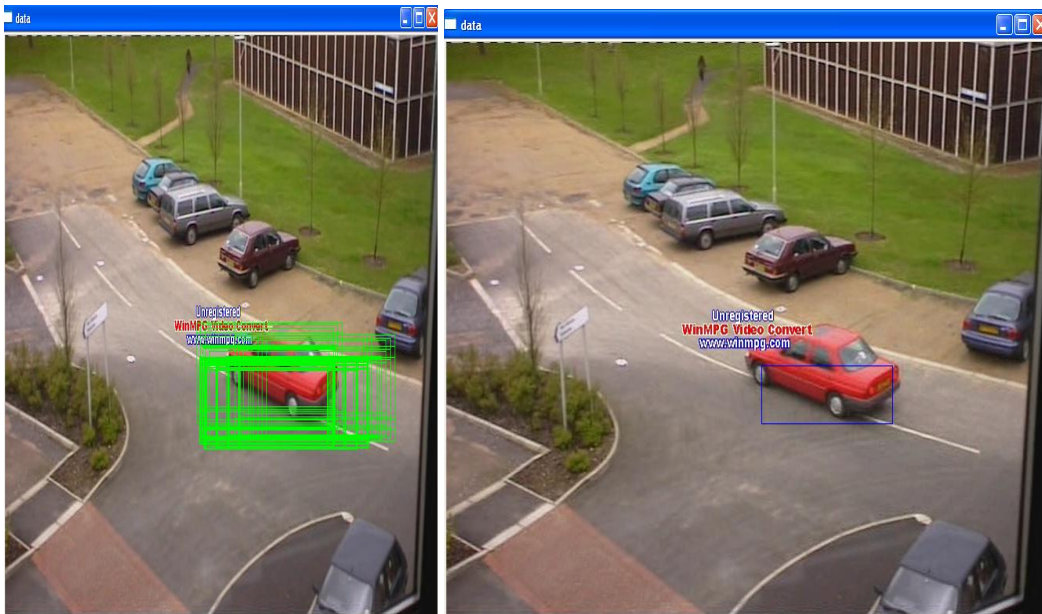


Figure 2. PSO searching process.

Figure 3 shows a car tracking process when occlusions happened. First, a white car drives in and is detected as the target. Then, it passes the scene and is occluded by the texts in the middle. During the occlusion, the tracking window changes, but still tracks the car. When the car moves away, the window becomes smaller until disappeared.



Figure 3. A tracking process when an occlusion happened.

In Figure 4, the student's face is initially identified by a red window. Then the system tracks the student using a green window while he is walking around in the office. When he enters a closet, his face is partially occluded. However, the system can still track the parts that are not occluded. When his face is completely occluded by a closet door, the tracking window loses its object. When the student comes out of the door again, the detection/tracking system recovers the tracking window immediately. During this procedure, the mobile platform of the camera keeps adjusting the camera's position and its own position to following the movement of the object.

The experimental results demonstrate the proposed algorithm is efficient and robust for object detection and tracking. The presented algorithm transfers the tracking problem into a searching problem by projecting potential solutions into a particle swarm searching. Each particle is a high dimensional vector that presents a possible tracking window instead of only a single pixel. Compared with pixel-classification methods [28], the presented algorithm focuses more on pitches, which makes it more robust and gives it more natural and continuous results. For tracking problem, due to the unpredictable character of images, it is difficult to build up adaptive tracking windows both in locations and sizes. The proposed algorithm integrates all parameters together into the frame of particles, which brings more flexibility and reliability. With movement of particles, all parameters are adjusted naturally without specific addressing, which makes the proposed algorithm easy to implement.



Figure 4. Face detection and tracking with disappear and recapture.

The Bayes-based filter takes advantage of correlations between image frames to guide and constrain PSO searching. In this way, the setting of PSO can be simplified and the tracking procedure can be accelerated. Actually the introduction of the Bayes filter opens a way to integrate PSO with other global algorithms. This frame can supply a good balance between exploration and exploitation, which is more powerful than traditional searching methods.

## 5. CONCLUSION

In this Chapter, an automatic object detection and tracking algorithm is proposed. First, a cascade of classifiers based on Haar-like features is applied to detect people face. Then a PSO-based searching algorithm searches for good candidates of adaptive tracking windows with parameters on the current frame. Finally, Bayes-based filter is employed to identify the best-matched tracking window under the motion constraints of the tracked object. The proposed algorithm was applied for several video clips under dynamic environment in an indoor office environment, which includes various situations like partially or completely occlusion, disappearance, reappearance. Experimental results showed that our proposed method is robust enough to adapt to the dynamic environment.

There are still several issues remained and need to be improved and extended in the future work. As we mentioned above, more object features need to be embedded to train the object model under different environment and light conditions. Furthermore, the employed classifier method can be extended to multiple classifiers so that the structure of weak

classifiers can be more adaptive to achieve higher robustness performance. The Bayes filter can also be concreted by using some predefined knowledge of the tracked targets.

## REFERENCE

- [1] Kuranov, R. Lienhart, and V. Pisarevsky, "An Empirical Analysis of Boosting Algorithms for Rapid Objects With an Extended Set of Haar-like Features," *Intel Technical Report MRL-TR-July02-01*, 2002.
- [2] Lipton, H. Fujiyoshi, and R. Patil, "Moving Target Detection and Classification from Real-Time Video", In *Proceedings of IEEE Workshop on Application of Computer Vision*, 1998,
- [3] A.P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*, John Wiley and Sons, Ltd. 2005.
- [4] AR.C. Eberhart and J. Kennedy. "A New Optimizer using Particle Swarm Theory", in *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pp. 39-43, 1995.
- [5] Bao Fumin, Li Aiguo, and Qin Zheng, "Photo Time-Stamp Recognition Based on Particle Swarm Optimization", *Proceedings of Web Intelligence*, 2004, pp. 529-532.
- [6] Beleznai, B. Fruhstuck, and H. Bischof, "Human Tracking By Mode Seeking," in *Proceedings of the 4th International Symposium on Image and Signal Processing and Analysis*, pp.1-6, Sept. 2005.
- [7] Wren, A. Azarbayejani, T. Darrell, and A. Pentland. "Pfinder: Real-Time Tracking of the Human Body", *Technical Report 353*, MIT Media Lab Perceptual Computer Section, 1995.
- [8] Comaniciu, and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Trans. Pattern Analysis Machine Intel.*, Vol. 24, No. 5, pp. 603-619, 2002.
- [9] Comaniciu, V. Ramesh, and P. Meer, "Real-Time Tracking of Non-Rigid Objects using Mean Shift", in *Proceedings of Computer Vision and Pattern Recognition*, Vol. 2, pp. 142-149, 2000.
- [10] Bradski, A. Kaehler, and V. Pisarevsky, "Learning-Based Computer Vision with Intel's Open Source Computer Vision Library," *Intel Technical Report*, Volume 09, Issue 02, 2005.
- [11] Welch and G. Bishop, "An Introduction to Kalman Filter", *SIGGRAPH 2001*, Course 8, 2001.
- [12] Tao, H. S. Sawhney and R. Kumar, "A sampling algorithm for tracking multiple objects," in *Proc. Workshop of Vision Algorithms with Int. Conf. on Computer Vision*, pp. 53-68, 1999.
- [13] Haritaoglu, D. Harwood, and L. S. Davis, "Hydra: Multiple People Detection and Tracking Using Silhouettes," in *10th International Conference on Image Analysis and Processing (ICIAP'99)*, pp. 280-285, 1999.
- [14] G. Allen, Y. D. Xu, and J. S. Jin, "Object tracking using Camshift algorithm and multiple quantized feature spaces," in *Pan-Sydney Area Workshop on Visual*

- Information Processing (VIP2003)*, Sydney. Conferences in Research and Practice in Information Technology, Vol. 36, pp. 3-7, 2003.
- [15] Kennedy and R.C. Eberhart, "Particle Swarm Optimization", in *Proceedings of IEEE International Conference on Neural Networks*, 1942-1948, 1995.
- [16] <http://www.projectcomputing.com/resources/psovis/index.html>
- [17] J. Kennedy, R.C. Eberhart, and Y. Shi, *Swarm Intelligence*, Morgan Kaufmann, 2001
- [18] K.E. Parsopoulos, V.P. Plagianakos, G.D. Magoulas, and M.N. Vrahatis, "Stretching Technique for Obtaining Global Minimizers through Particle Swarm Optimization", in *Proceedings of the IEEE Workshop on Particle Swarm Optimization*, pp. 22-29 2001
- [19] Keisuke Kameyama, Nozomi Oka, and Kazuo Toraichi, "Optimal Parameter Selection in Image Similarity Evaluation Algorithms Using Particle Swarm Optimization", *IEEE Congress on Evolutionary Computation*, pp. 1079-1086, 2006.
- [20] Anton-Canalis, M. Hernandez-Tejera, and E. Sanchez-Nielsen etc, "Particle Swarms as Video Sequence Inhabitants For Object Tracking in Computer Vision", *Sixth International Conference on Intelligent System Design and Applications*, Oct. 16-18, 2006.
- [21] Isard, "Visual Motion Analysis by Probabilistic Propagation of Conditional Density", *D.Phil. Thesis*, Oxford University, 1998.
- [22] M.P. Wachowiak et al. "An Approach to Multimodal Biomedical Image Registration Utilizing Particle Swarm Optimization", *IEEE Transaction on Evolutionary Computation*, Vol. 8, pp. 289-301, 2004.
- [23] M.S. Voss and J.C. Howland III, "Financial Modeling using Social Programming", in *International Conference on Financial Engineering and Applications*, pp. 1-10, 2003
- [24] Mahamed G.H. Omran, "Particle Swarm Optimization Methods for Pattern Recognition and Image Processing", *Ph.D. Dissertation*, University of Pretoria etd, South Africa, 2005.
- [25] Mattias, Scheutz, "Real-Time Hierarchical Swarm for Rapid Adaptive Multi-level Pattern Detection and Tracking", *Swarm Intelligence Symposium*, 2007, pp. 234-241.
- [26] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *IEEE Conf. on Computer Vision and Pattern Recognition*, Vol. 1, pp. 511-518, 2001.
- [27] Payam Saisan, Swarup Medasani, and Yuri Owechko, "Multi-View Classifier Swarms for Pedestrian Detection and Tracking", in *Proceedings of Computer Vision and Pattern Recognition*, Vol. 03, pp 18, 2005.
- [28] R. Akbari, M. D. Jazi, and M. Palhang, "A Hybrid Method for Robust Multiple Objects Tracking in Cluttered Background", Vol. 1, 24-28, *IEEE International Conference on Information and Communication Technologies: from Theory to Applications (ICTTA06)*, 2006.
- [29] R. C. Eberhart and J. Kennedy. "A New Optimizer Using Particles Swarm Theory", in *Proc. Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan., IEEE Service Center, Piscataway, NJ, 39-43, 1995.
- [30] R. Senaratne, S. Halgarnuge, "Using Particle Swarm Optimization for Elastic Bunch Graph Matching to Recognize Faces", *Tencon 2005*, pp. 1-6.
- [31] R.C. Eberhart and Y. Shi, "Tracking and Optimizing Dynamic Systems with Particle Swarms", in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 1, pp. 94-100 2001

- [32] S. Maskell and N. Gordon, "A Tutorial on Particle Filters for On-line Nonlinear/Non-Gaussian Bayes' Law Tracking", *IEEE Transactions on Signal Processing*, 50, 2 (Feb. 2002), pp.174—188, 2002.
- [33] T. Kailath, "The Divergence and Bhattacharyya Distance Measures in Signal Selection," *IEEE Trans. Commun. Techn.*, COM-15:52-60, 1967.
- [34] T. Olson and F. Brill. "Moving Object Detection and Event Recognition Algorithms for Smart Cameras", in *Proc. DARPA Image Understanding Workshop*, pp. 159-175, 1997.
- [35] T. Sousa, A. Silva and A. Neves, "Particle Swarm Based Data Mining Algorithms for Classification Tasks", *Parallel Computing*, 30(5-6): 767-783, 2004
- [36] T.K. Rasmussen and T.Krink, "Improved Hidden Markov Model Training for Multiple Sequence Alignment by a Particle Swarm Optimization-Evolutionary Algorithm Hybrid", *Biosystems*, pp. 5-17 2003
- [37] X. Hu and R.C. Eberhart, "Multiobjective Optimization using Dynamic Neighborhood Particle Swarm Optimization", in *Proceedings of the IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1677-1681 2002
- [38] X. Xiao, E.R. Dow, R.C. Eberhart, Z. Ben miled, and R.J. Oppelt, "Gene Clustering using Self-Organizing Maps and Particle Swarm Optimization", in *Proceedings of the Second IEEE International Workshop on High Performance Computational Biology*, pp. 10 2003.
- [39] Y. Owechko, S. Medasani, and N. Srinivasa, "Classifier Swarms for Human Detection in Infrared Imagery," in *2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04)*, Vol. 8, pp. 121, 2004.