Hindawi Publishing Corporation
Advances in Multimedia
Volume 2012, Article ID 343724, 14 pages
doi:10.1155/2012/343724

Research Article

Multitarget Tracking of Pedestrians in Video Sequences Based on Particle Filters

Hui Li, Shengwu Xiong, Pengfei Duan, and Xiangzhen Kong

School of Computer Science and Technology, Wuhan University of Technology, Wuhan 430070, China

Correspondence should be addressed to Hui Li, lipeilin1984xyz@163.com

Received 31 August 2012; Revised 4 November 2012; Accepted 9 November 2012

Academic Editor: Weidong Cai

Video target tracking is a critical problem in the field of computer vision. Particle filters have been proven to be very useful in target tracking for nonlinear and non-Gaussian estimation problems. Although most existing algorithms are able to track targets well in controlled environments, it is often difficult to achieve automated and robust tracking of pedestrians in video sequences if there are various changes in target appearance or surrounding illumination. To surmount these difficulties, this paper presents multitarget tracking of pedestrians in video sequences based on particle filters. In order to improve the efficiency and accuracy of the detection, the algorithm firstly obtains target regions in training frames by combining the methods of background subtraction and Histogram of Oriented Gradient (HOG) and then establishes discriminative appearance model by generating patches and constructing codebooks using superpixel and Local Binary Pattern (LBP) features in those target regions. During the process of tracking, the algorithm uses the similarity between candidates and codebooks as observation likelihood function and processes severe occlusion condition to prevent drift and loss phenomenon caused by target occlusion. Experimental results demonstrate that our algorithm improves the tracking performance in complicated real scenarios.

1. Introduction

Video target tracking is an important research field in computer vision for its wide range of application demands and prospects in many industries, such as military guidance, visual surveillance, visual navigation of robots, human-computer interaction and medical diagnosis [1–3], and so forth. The main task of target tracking is to track one or more mobile targets in video sequences so that the position, velocity, trajectory, and other parameters of the target can be obtained. Two main tasks needs to be completed by moving target tracking during the processing procedure: the first one is target detection and classification which detects the location of relevant targets in the image frames; the second one is the relevance of the target location of consecutive image frames, which identifies the target points in the image and determines their location coordinates, thus to determine the trajectory of the target as time changes. However, automated detection and tracking of pedestrians in video sequences is still a challenging task because of following reasons [4]. (1) Large intraclass variability which refers to various changes in appearance of pedestrians due to different poses, clothing, viewpoints, illumination, and articulation. (2) Interclass similarities which are the common likeness between pedestrians and other background objects in heavy cluttered environment. (3) Partial occlusions, which may change frequently in a dynamic scene, of pedestrians which are caused by other interclass or intraclass targets.

Considering the difficulties mentioned above in pedestrians detection and tracking tasks, pedestrians tracking has been studied intensively and a number of elegant algorithms have been established. One popular tracking method is mean shift procedure [5], which finds the local maximum of probability distribution in the direction of gradient. Comaniciu and Ramesh [6] gave a strict proof of the convergence of the algorithm and proposed a mean shift based on tracking method. As a deterministic method, mean shift keeps single hypothesis and is thus computationally efficient. But it may run into trouble when similar targets are presented in background or occlusion occurs. Another common approach is the use of the Kalman filter [7]. This approach is based on the assumption that the probability distribution of the target

state is Gaussian, and therefore the mean and covariance, computed recursively by the Kalman filter equations, can fully characterize the behavior of the tracked target. However, in video target tracking, tracking targets in real world rarely satisfy Gaussian assumptions required by the Kalman filter in that background clutter may resemble a part of foreground features. One promising category is sequential Monte Carlo approach, which is also known as particle filter [8], which recursively estimates target posterior with discrete sample-weight pairs in a dynamic Bayesian framework. Due to particle filters' non-Gaussian, non-linear assumption and multiple hypothesis property, they have been successfully applied to video target tracking [9].

2. Previous Work

Various researchers have attempted to extend particle filters to target tracking. Among others, one of the most successful features used in target tracking is color. Nummiaro et al. [10] proposed a tracking algorithm that considered color histograms, as a feature, that were tracked using the particle filter algorithm. Despite the algorithm being more robust to the partial blocked target and the target shape changes, the algorithm exhibits high sensitivity to illumination changes that may cause the tracker to fail. Vermaak et al. [11] introduced a mixture particle filter (MPF), where each component was modeled with an individual particle filter that formed part of the mixture. The filters in the mixture interacted only through the computation of the importance weights. By distributing the resampling step to individual filters, the MPF avoids the problem of sample depletion. Okuma et al. [12] extended the approach of Vermaak et al. and proposed a boosted particle filter. The algorithm combined the strengths of two successful algorithms: mixture particle filters and adaboost. It is a simple and automatic multiple target tracking system, but it is easy to fail in tracking when the background image is complex.

Therefore, a more effective method for target recognition is needed. Superpixel has been one of the most promising representations with demonstrated success in image segmentation and target recognition [13–15]. For this reason, Ren and Malik [16] proposed a tracking method based on superpixel, which regards tracking task as a figure/ground segmentation across frames. However, as it processes every entire frame individually with Delaunay triangularization and conditional random field (CRF) for region matching, the computational complexity is rather high. Further, it is not designed to handle complex scenes including heavy occlusion and cluttered background as well as large lighting change. Wang et al. [17] proposed a tracking method from the perspective of mid-level vision with structural information captured in superpixel. The method is able to handle heavy occlusion and recover from drifts. Thus in this paper, the observation model adopts superpixel which is combined with the LBP to extract the target feature.

In recent years, bag of features (BoF) representation has been successfully applied to object and natural scene classification owing to its simplicity, robustness, and good practical performance. Yang et al. [18] proposed a visual tracking approach based on BoF. The algorithm randomly samples image patches within the object region in training frames to construct two codebooks using RGB and LBP features instead of only one codebook in traditional BoF. It is more robust in handling occlusion, scaling and rotation, but it can only track one target. Based on the advantages of BoF in target tracking, the paper employs BoF to establish discriminative appearance model, which converts high-dimensional feature vector into low-dimensional histogram comparison, overcoming high computational complexity due to superpixel in the observation model.

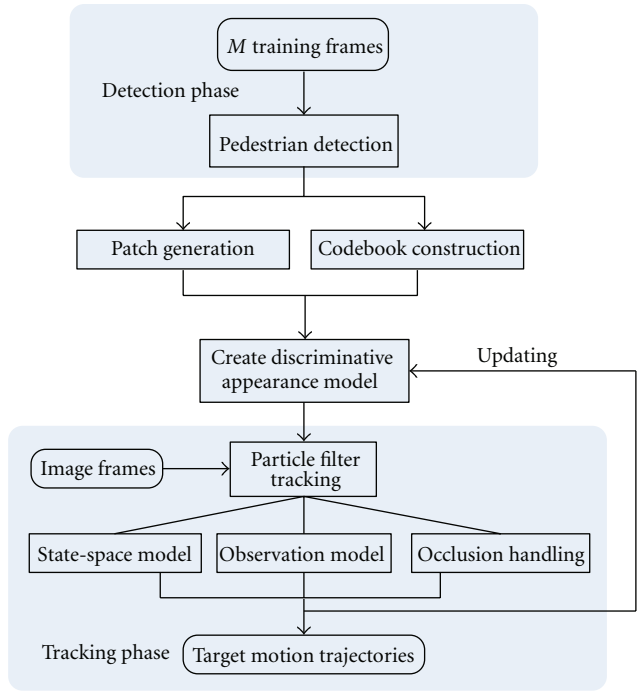


Figure 1: The flowchart of algorithm.

Therefore, to achieve automated and robust tracking of pedestrians in complex scenarios, we present multi-target tracking of pedestrians in video sequences based on particle filters. The algorithm uses BoF algorithm to create discriminative appearance model which is then used to be combined with particle filter algorithm to achieve target tracking. In order to improve the efficiency and accuracy of the detection, firstly, background subtraction and the HOG detection methods are combined to get the target motion regions in the training frames. And then the discriminative appearance model established by the target regions is used to discriminate the candidate targets. During the process of tracking, severe occlusion condition is handled to prevent drift and loss phenomenon due to pedestrians' mutual occlusion. Figure 1 shows the entire algorithmic flowchart.

The paper is organized as follows: Section 3 introduces detection of pedestrians; Section 4 describes our particle filter algorithm; Section 5 presents the experimental results and the performance evaluation and conclusion work is given in Section 6.

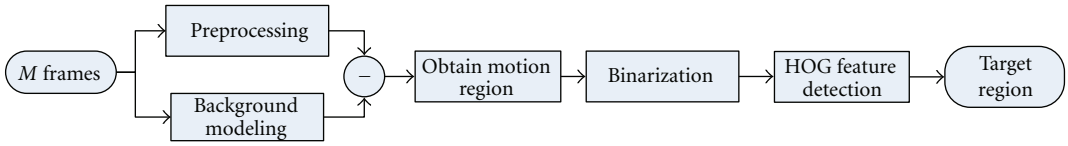


Figure 2: Flow diagram of target regions extraction of M frames.

3. Detection of Pedestrians

There are mainly two parts in this section, one is target regions extraction, and the other is the construction of the discriminative appearance model. The former aims to determine the target regions of video sequence in the first M frames, the latter aims to do sampling, feature extraction in the target region when these target regions are seen as a training set, and eventually establish the discriminative appearance model.

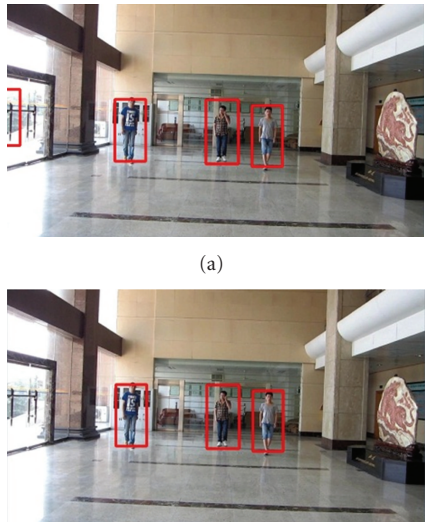
3.1. Target Regions Extraction. Before tracking, we need to detect the targets in the first M frames and get the target regions in each frame for later trainings. Figure 2 shows the whole flow diagram of target regions extraction of the first M frames.

We can see from Figure 2 that, first of all, in order to get motion region, a simple and fast approach is to perform background subtraction, which identifies motion targets from the portion of video sequences that differ significantly from a background model, as shown in Figure 3. Then we use the HOG descriptors [19] and Support Vector Machines (SVMs) to build a pedestrian detector. Since the method has been proved to be capable but time-consuming, we only detect motion regions which have been acquired by background subtraction and M frames. This not only reduces the HOG detection region, but also improves the efficiency and the accuracy of the detection. Figure 4 shows that adopting the HOG detection after background subtracting improves the accuracy of pedestrian detection, whereas using the HOG directly can lead to false detection.

3.2. Discriminative Appearance Model. During this stage, discriminative appearance model is created by target regions extraction of the first M frames to distinguish targets from cluttered backgrounds. The kth pedestrian in the tth frame is p_t^k $(t = 1, 2, \ldots, M; k = 1, 2, \ldots, F_s)$, where M is the number of training frames, F_s is the number of target pedestrians in the training frames. According to all M-frame regions in which pedestrian k appears, we draw the pedestrian's discriminative appearance model (We assume that the number of targets in the training frames is invariable.), and therefore we need get F_s discriminative appearance models.

3.2.1. Patch Generation. In the training stage, some patches with a constant scale are randomly sampled within the region of the pedestrian p_t^k. For pedestrian k, M image patches are collected and represented by superpixel descriptor and LBP descriptor, respectively, in each training frame. Superpixel descriptor and LBP descriptor extraction process in training frames is illustrated in Figure 5.

Figure 3: Background subtraction result.



(a)

(b)

Figure 4: The HOG detection result. (a) Detection results of using the HOG directly. (b) Detection results of adopting the HOG after background subtracting.

The superpixel segmentation method we adopt in this paper is SLIC [15] (Simple Linear Iterative Clustering) that clusters pixels in the combined five-dimensional color and image plane space to efficiently generate compact, nearly uniform superpixel. For superpixel descriptor, we segment target region in tth training frame into S_t superpixels, as shown in Figure 5. As the superpixel does not have a fixed shape, and its distribution is often irregular, it is unsuitable for extracting the local template information; in addition, due to the similarity of the superpixel's internal pixel texture as well as the similarity of color characteristics, more stable superpixel information can be obtained by extracting the color space histogram. However, RGB color space distribution does not accord with human's vision distribution, and it is not robust enough for illumination

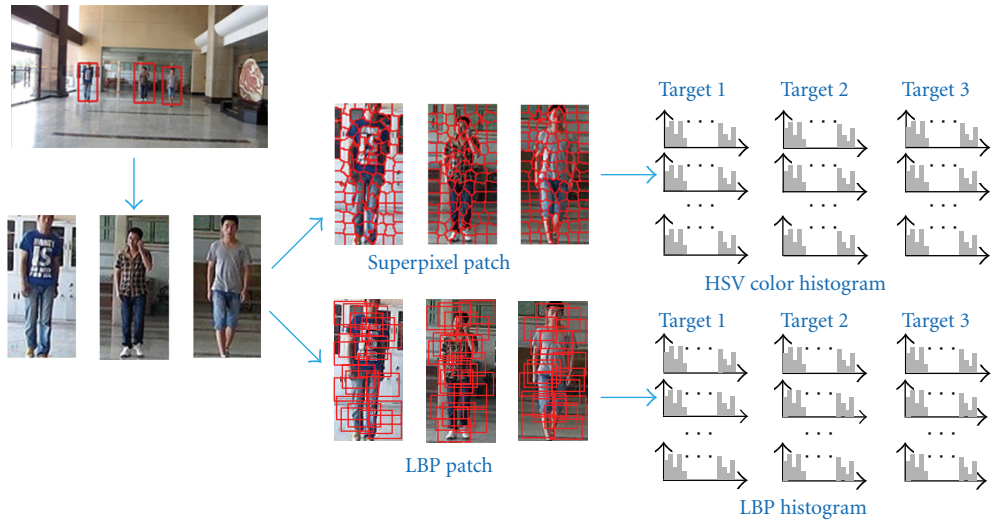


Figure 5: Extraction process of superpixel descriptor and LBP descriptor in training frames.

changes, therefore we only use the normalized histogram of HSV color space which is simple and accords with human's vision as a feature for all superpixels.

LBP is vastly used for texture description which has good performance in texture classification, fabric defect detection and moving region detection. LBP is an illumination invariant descriptor which is not sensitive to the intensity change caused by the light changes. The LBP descriptor is stable as long as the differences among the image pixel values do not change a lot. In addition, there are certain complementary between LBP and color features, so we adopt LBP descriptor as a feature. The LBP descriptor is defined as follows:

$$\mathrm{LBP}(P_c) = \sum_{n=0}^{p-1} s(g_n - g_c)2^n, \qquad s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0, \end{cases} \tag{1}$$

where g_c is the intensity value of center pixel P_c and g_n is the intensity of neighboring pixels.

The image histogram obtained from the computation of LBP is defined as follows:

$$H_i = \sum_{x,y} I(f(x,y) = i), \quad i = 0, \ldots, n-1, \qquad I(f(x,y) = i) = \begin{cases} 1 & f(x,y) = i \\ 0 & f(x,y) \neq i, \end{cases} \tag{2}$$

where $n = 2^p$ represents the length of the encode bit generated by the LBP operator, p represents the number of pixels in the neighborhood, $f(x,y)$ is the LBP value at (x,y), in this way, H_i represents the number of pixels which have the LBP value of i, the histogram can reflect the distribution of the LBP values.

3.2.2. Codebook Construction. As frames slip, patches accumulate. For extracted collections of sample features $\mathrm{Sample}_k = \{S_n\}_{n=1}^{M}$, features are gathered into a number of clusters by performing mean shift clustering, and cluster centers $C = \{C_{\mathrm{cl}}\}_{\mathrm{cl}=1}^{\mathrm{cl\_num}}$ compose the codebook. Here cl_num is the number of cluster centers as well as the size of the codebook. Cluster centers which represent the most typical features are regarded as the keywords in the codebook and used to create bags. In this way, a large collection of sample characteristics is converted into a comparatively small codebook. Figure 6 shows the process of codebook construction.

After codebook construction, for each characteristic of a set of features Sample_k in each training sample image, find the codeword which has the nearest Euclidean distance from it, then count the appearance times of all features corresponding nearest codeword to acquire the final histogram. Repeat the above steps to M training sample images, a set of training images will be converted into a set of histograms called bags. A bag is equivalent to the occurrence frequency of codewords in an image and can be represented as a histogram. M training images are converted to a set of bags $\{B_m\}_{m=1}^{M}$ by raw counts.

Here the discriminative appearance model has been established for subsequent classification decisions.

3.2.3. Updating. Since appearance and pose changes of a target occur all the time, updating is necessary or even crucial. After M frames, a new collection of patches $\{P_i\}_{i=1}^{fp}$ is obtained. We then perform mean shift clustering again on $\{P_i\}_{i=1}^{fp}$ and the old codebook $\{C_{\mathrm{cl}}\}_{\mathrm{cl}=1}^{\mathrm{cl\_num}}$ using

$$C_{\mathrm{new}} = \{C_{\mathrm{cl}}\}_{\mathrm{cl}=1}^{\mathrm{cl\_num}} = \mathrm{meanshift}\left(\{P_i\}_{i=1}^{fp}, \mu\{C_{\mathrm{cl}}\}_{\mathrm{cl}=1}^{\mathrm{cl\_num}}\right). \tag{3}$$

Here, C_{new} denotes the new codebook. μ $(0 < \mu < 1)$ is a forget factor imposed on the old codebook to reduce its importance gradually so that the newly-constructed codebook pays more attention to the latest patches.

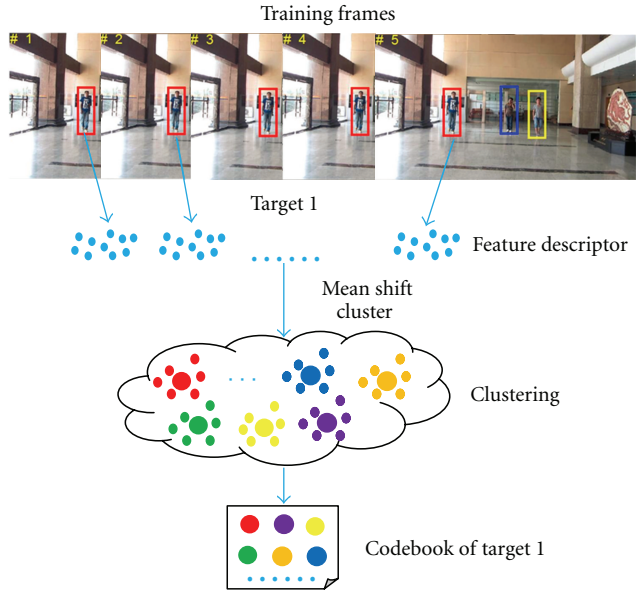


Figure 6: Codebook construction of target 1.

4. Particle Filter Tracking

The particle filter [8] is a Bayesian sequential importance sampling technique, which recursively approximates the posterior distribution using a finite set of weighted samples. It consists of two essential steps: prediction and update. We use $X_t = [x_{1,t}, \ldots, x_{F_t,t}]$ to express the set of states of the target system at moment t. In the set of states X_t, F_t stands for the target's states number at moment t; $x_{j,t}, j = 1, \ldots, F_t$ stands for the state of the jth target at moment t.

Given all available observations $Z_{1:t-1} = \{Z_1, \ldots, Z_{t-1}\}$, up to time $t-1$, the prediction stage uses the probabilistic system transition model $p(X_t \mid X_{t-1})$ to predict the posterior at time t as

$$p(X_t \mid Z_{1:t-1}) = \int p(X_t \mid X_{t-1}) p(X_{t-1} \mid Z_{1:t-1}) dX_{t-1}. \quad (4)$$

At time t, the observation Z_t is available, the state can be updated using Bayesian's rule:

$$p(X_t \mid Z_{1:t}) = \frac{p(Z_t \mid X_t) p(X_t \mid Z_{1:t-1})}{p(Z_t \mid Z_{1:t-1})}, \quad (5)$$

where $p(Z_t \mid X_t)$ is described by the observation equation.

4.1. State-Space Model. In the video scene, the movement of each target can be considered as an independent process, and therefore state-space model can be regarded as the joint product form of a single-target motion model:

$$p(X_t \mid X_{t-1}) = \prod_{i=1}^{F} p(x_{i,t} \mid x_{i,t-1}). \quad (6)$$

Suppose the target state number of both moment t and $t-1$ are $F(F \geq \max\{F_{t-1}, F_t\})$, $X_t = [x_{1,t}, x_{2,t}, \ldots, x_{F,t}]$, F_t is the state number in X_t, F_{t-1} is the state number in X_{t-1}, $x_{j,t}$ is the state of the jth video target at moment t, $x_{j,t} = [x_j, y_j, w_j, h_j]$, x_j and y_j are respectively the rectangle center's position in the direction of x and y in the image, w_j and h_j are the length and width of the rectangle.

To get the state transition density function of the jth target at moment t, random perturbation model is used to describe the state transition of the jth target from momet $t-1$ to moment t, that is,

$$p\left(x_{j,t} \mid x_{j,t-1}\right) = N\left(x_{j,t}; x_{j,t-1}, \sum\right), \quad (7)$$

where $N(x_{j,t}; x_{j,t-1}, \sum)$ is the normal density function whose covariance is $\sum$. $\sum$ is a diagonal matrix, and the variances of the four parameters in $\sum$'s diagonal elements corresponding state $x_{j,t}$ are $\sigma_x^2, \sigma_y^2, \sigma_w^2, \sigma_h^2$. Random perturbation model is used to describe the motion of each target mainly in the condition that the tracking targets of the video are pedestrians who have movement randomness, thus it is difficult to predict the state of motion for the next moment by using constant-velocity model or constant acceleration model.

4.2. Observation Model. When a new frame arrives, for target k, firstly, according to its location at the last frame, state-space model is used to randomly sample T candidate targets, as illustrated in Figure 7.

Secondly, each candidate target is handled as follows:

(1) extract N superpixel patches.

We adopt superpixel segmentation to each candidate target and obtain N superpixels. Then extract each superpixel's HSV color histogram and normalized them.

(2) extract N LBP patches.

Extract N patches from each candidate target, and then calculate each patch's LBP histogram and normalize them.

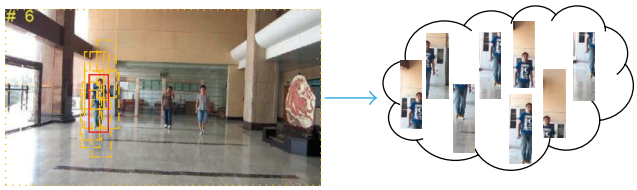


Figure 7: Collection of T candidate targets.

Then calculate the color histogram and the LBP histogram of N patches (each superpixel is also referred to as a patch) separately according to the following process:

We calculate the patches' similarities with codewords, so a similarity function is defined as follows:

$$\mathrm{sim}^i = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{d^2\left[S_i, C_j\right]}{2\sigma^2}\right), \quad (j = 1, \ldots, \mathrm{cl\_num}), \tag{8}$$

where sim^i denotes the similarity between patch i and each codeword j, $i = 1, \ldots, N$; S_i denotes the eigenvector of the test patch i, C_j denotes the eigenvector of the codeword j in the codebook, $d^2[S_i, C_j]$ denotes the histogram intersection distance between the two histogram images.

Thus, the patches in each candidate target all have their most similar codewords. Make a statistics of the occurring frequency of codewords in each candidate target T as a bag of features, B_T, which is illustrated in the following formula:

$$B_T = \sum_{i=1}^{N} I\left(\underset{j}{\mathrm{argmax}}\left(\mathrm{sim}^i\right) = j\right), \quad j = 1, \ldots, \mathrm{cl\_num},$$
$$I\left(\underset{j}{\mathrm{argmax}}\left(\mathrm{sim}^i\right) = j\right) = \begin{cases} 1 & \underset{j}{\mathrm{argmax}}\left(\mathrm{sim}^i\right) = j \\ 0 & \underset{j}{\mathrm{argmax}}\left(\mathrm{sim}^i\right) \neq j. \end{cases} \tag{9}$$

Then we compute the similarity of bags to get the weight of each candidate target:

$$w = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{d_t^2[B_T, B_m]}{2\sigma^2}\right), \tag{10}$$

where B_m denotes the eigenvector of the test sample, B_T denotes the eigenvector of the template, $d_t^2[B_T, B_m]$ denotes the bag of features intersection distance between the two patches.

The observation likelihood function is defined as follows:

$$p(Z_t \mid X_t) = \max\left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{d^2\left[S_i, C_j\right]}{2\sigma^2}\right)\right), \quad (i = 1, \ldots, N; j = 1, \ldots, \mathrm{cl\_num};). \tag{11}$$

In this way, we get $w_{\mathrm{superpixel}}$, $p_{\mathrm{superpixel}}(Z_t \mid X_t)$, w_{LBP}, and $p_{\mathrm{LBP}}(Z_t \mid X_t)$, respectively. In the condition that X_t is the given target state, the total observation likelihood function of the target is defined as follows:

$$p_{\mathrm{all}}(Z_t \mid X_t) = a \cdot p_{\mathrm{superpixel}}(Z_t \mid X_t) + b \cdot p_{\mathrm{LBP}}(Z_t \mid X_t)$$
$$a = \frac{w_{\mathrm{superpixel}}}{w_{\mathrm{superpixel}} + w_{\mathrm{LBP}}}, \quad b = \frac{w_{\mathrm{LBP}}}{w_{\mathrm{superpixel}} + w_{\mathrm{LBP}}}, \quad a + b = 1, \tag{12}$$

where $p_{\mathrm{superpixel}}(Z_t \mid X_t)$, $p_{\mathrm{LBP}}(Z_t \mid X_t)$ are the observation likelihood functions of superpixel and LBP features respectively, $0 \leq a, b \leq 1$ are the weights of the two characteristics information in the fusion. The feature weights can be dynamically calculated through the weight distribution of the particle sets.

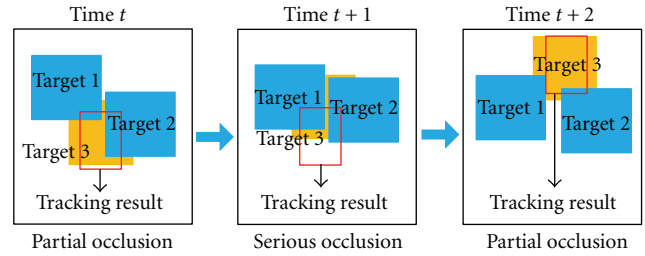


Figure 8: Tracking result in severe occlusion condition.

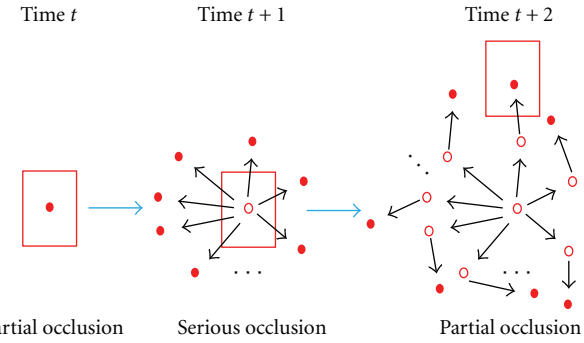


Figure 9: Particles' movements in severe occlusion condition.

4.3. Occlusion Handling. The above procedure can be used to handle partial occlusion of the target. However, when there is severe or complete occlusion, the total observation likelihood value of the target becomes extremely small. As to that situation, when the total observation likelihood value is smaller than certain threshold, we keep the target's last tracking state unchanged and the particles continue state transition. Tracking result and particles' movements in severe occlusion condition are illustrated in Figures 8 and 9, respectively.

4.4. The Algorithmic Process. The entire algorithmic process can be summarized as in Algorithm 1.

Table 1: Parameters of our algorithm.

Parameters	Value
Number of training frames	5
Size of codebook	20
Number of LBP patches	50
Number of superpixel patches	200
Forget factor	0.9
Number of particles	300

5. Experimental Verification and Analysis

To verify performance of our algorithm, we evaluate our algorithm on some video sequences. These sequences are acquired from our own dataset, PETS 2012 Benchmark data and CAVIAR database where the target pedestrians move in different conditions which include complex background, severe occlusion, illumination and changes of walking speed, and so forth.

In our algorithm, parameter settings are shown in Table 1. These parameters are fixed for all video sequences.

5.1. Comparison with Other Trackers. For comparison purposes, these sequences are utilized to evaluate the performance of superpixel tracking, boost particle filter (BPF) and our algorithm under the situation of occlusion.

The video parameters in the evaluation are shown in Table 2.

First of all, sequence "three pedestrians in the hall" is tested, in which three pedestrians are walking in the hall from our own dataset. In Figure 10, the first row and the second row represents the outcomes of the algorithm which are contrasted with those of superpixel tracking and BPF respectively. We can see from these frames that BPF tracker leads to drifts under the situation of the pedestrian's occlusion and the pedestrian's distraction in that BPF tracker constructs proposal distribution using a mixture model that incorporates information from the dynamic models of each pedestrian and the detection hypotheses generated by Adaboost. However, when partial occlusion occurs, BPF tracker cannot get enough pedestrian feature descriptions, which leads to the failure. By contrast, both superpixel tracking and our algorithm track the targets because they require only part of the feature to track targets, and they are able to handle severe occlusion and recover from drifts. Therefore, both superpixel tracking and our algorithm can track the targets accurately, but the latter has better tracking accuracy and robustness than the former.

The pedestrians' weight variation curves of superpixel weight and LBP weight in the process of tracking are illustrated in Figure 11. Because occlusion does not occur in the tracking process to pedestrian 3, there is no obvious fluctuation of superpixel weight and LBP weight. Superpixel weight begins to decline after the 107th frame in which the occlusions between pedestrians emerge and LBP weight begins to increase. As the targets move, the interferences of the occlusions between pedestrians move away after the 123th frame, therefore superpixel weight regains the state of being higher than LBP weight.

Figure 12 shows three pedestrians' position error respectively in the process of target tracking. For each pedestrian, the position error is defined as follows:

$$\text{PositionError}_t = \sqrt{(x'_t - x_t)^2 + (y'_t - y_t)^2}, \quad (13)$$

where (x'_t, y'_t) denotes the estimation value of target position at moment t, (x_t, y_t) denotes the real position at moment t, PositionError_t denotes the mean-square-root error at moment t.

We can see that the our algorithm has better accuracy than any of the other two in that using the superpixel tracking and the BPF tracking. It can be seen that the robustness of tracking is improved by using our algorithm.

Figure 13 shows target motion trajectories from the first frame to the last by using our algorithm. The different colors represent different pedestrian trajectories. The points in the graph constitute target motion trajectory, and each point represents the target location of each frame.

Secondly, sequence "five pedestrians in the corridor" is tested from CAVIAR database, in which there are twice severe occlusions. Figure 14 shows that our algorithm has better tracking accuracy and robustness, although the pedestrians' severe mutual occlusion occurs. Figure 15 shows target motion trajectories from the first frame to the last by using our algorithm.

Thirdly, sequence "sparse crowd" is tested from PETS 2012 Benchmark data. It can be seen from Figure 16 that there are failures in tracking when either the superpixel tracking or the BPF tracking is used. However, our algorithm can track all the targets in the condition of severe occlusion, pose variation, or changes of walking speed. Figure 17 shows target motion trajectories from the first frame to the last by using our algorithm.

Finally, sequence "two pedestrians in the square" is tested, in which one pedestrian was severely obscured by another pedestrian at a time. It differs from the first group of videos in that certain changes happen to pedestrians' walking environment illumination, that is, from the strong illumination into the weak illumination environment. Figure 18 shows our algorithm has better tracking accuracy and robustness. Although the pedestrians' walking illumination changes and severe mutual occlusion occurs, they are tracked out with accurate location. Figure 19 shows target motion trajectories from the first frame to the last by using our algorithm.

The quantitative evaluations of the superpixel tracking, BPF, and our algorithm are presented in Table 3. It can be seen from the table that our algorithm has smaller average errors of center location in pixels than the other two algorithms, thus it has better tracking accuracy. For each pedestrian, the average position error is defined as follows:

$$\overline{\text{PositionError}} = \frac{1}{\text{Frames}} \sum_{i=1}^{\text{Frames}} \text{PositionError}_i, \quad (14)$$

where Frames denotes the total frame numbers of the tracked video sequence, $\overline{\text{PositionError}}$ denotes the average mean-square-root error which measures the experiment results

(1) Extract target regions in the first M frames
  (1.1) Perform background subtraction and get motion region.
  (1.2) Building a pedestrian detector using HOG descriptors and SVM.
  (1.3) Detect F pedestrians in each frame.
(2) Build up the discriminative appearance model
  For $t = 1, 2, \ldots, M$
    (2.1) Randomly sample patches
      Generate superpixel patches and LBP patches for each target.
      Extract superpixel descriptor and LBP descriptor for all patches.
  End For
  For target $= 1, 2, \ldots, F$
    (2.2) Construct Codebook
      perform meanshift clustering for all superpixel descriptor, cluster centers $C_{\text{superpixel}} = \{C_{\text{cl}}\}_{\text{cl}=1}^{\text{cl\_num}}$ compose the superpixel codebook.
      perform meanshift clustering for all LBP descriptor, cluster centers $C_{\text{LBP}} = \{C_{\text{cl}}\}_{\text{cl}=1}^{\text{cl\_num}}$ compose the LBP codebook.
      For $t = 1, 2, \ldots, M$
        Find its nearest keyword and make statistics of the appearance times of the keyword.
      End For
      Compose trained bags $\{B_m\}_{m=1}^{M}$.
  End For
(3) Tracking
  For target $= 1, 2, \ldots, F$
    (3.1) Initialize particle state distribution $\{X_M^{(i)}\}_{i=1}^{T}$ using the center of specifying region.
    (3.2) Set initial weight value of feature information $a = b = 0$.
  End For
  For $t = M+1, M+2, \ldots$
    For target $= 1, 2, \ldots, F$
      (3.3) Important sampling step
        Propagate $\{X_M^{(i)}\}_{i=1}^{T}$ and get new particles $\{X_{M+1}^{(i)}\}_{i=1}^{N}$ using (7).
      (3.4) Update the weights
        Compute the observation likelihood function $p_{\text{superpixel}}(Z_t \mid X_t)$ and $p_{\text{LBP}}(Z_t \mid X_t)$ for each particle using (11).
        Update weights value of features information using (12).
        If $p_{\text{all}}(Z_t \mid X_t) < TH$
          $X_t = X_{t-1}$
        End if
    End For
  End For
  (3.5) Update codebook
    For each M frames
      Perform mean shift clustering again on $\{P_i\}_{i=1}^{fp}$ and the old codebook $\{C_{\text{cl}}\}_{\text{cl}=1}^{\text{cl\_num}}$ using (3).
    End For
  (3.6) State estimation
    Estimate the state $X_t = E(X_t \mid Z_{1:t}) \approx \sum_{i=1}^{N} \widetilde{w_t^{(i)}} \widetilde{X_t^{(i)}}$

Algorithm 1: Our algorithm.

Table 2: Video parameters in simulation.

Sequences	Frame size	Total frames	fps (frames per second)
(1) Three pedestrians in the hall	800×450	131	30
(2) Five pedestrians in the corridor	384×288	238	25
(3) Sparse crowd	768×576	74	30
(4) Two pedestrians in the square	640×360	151	30

Figure 10: Sequence 1: tracking results. The results by our algorithm, superpixel tracking, and BPF methods are represented by solid line, dashed line, and dotted line rectangles. Rectangles in different colors denote the tracking results of different pedestrians.

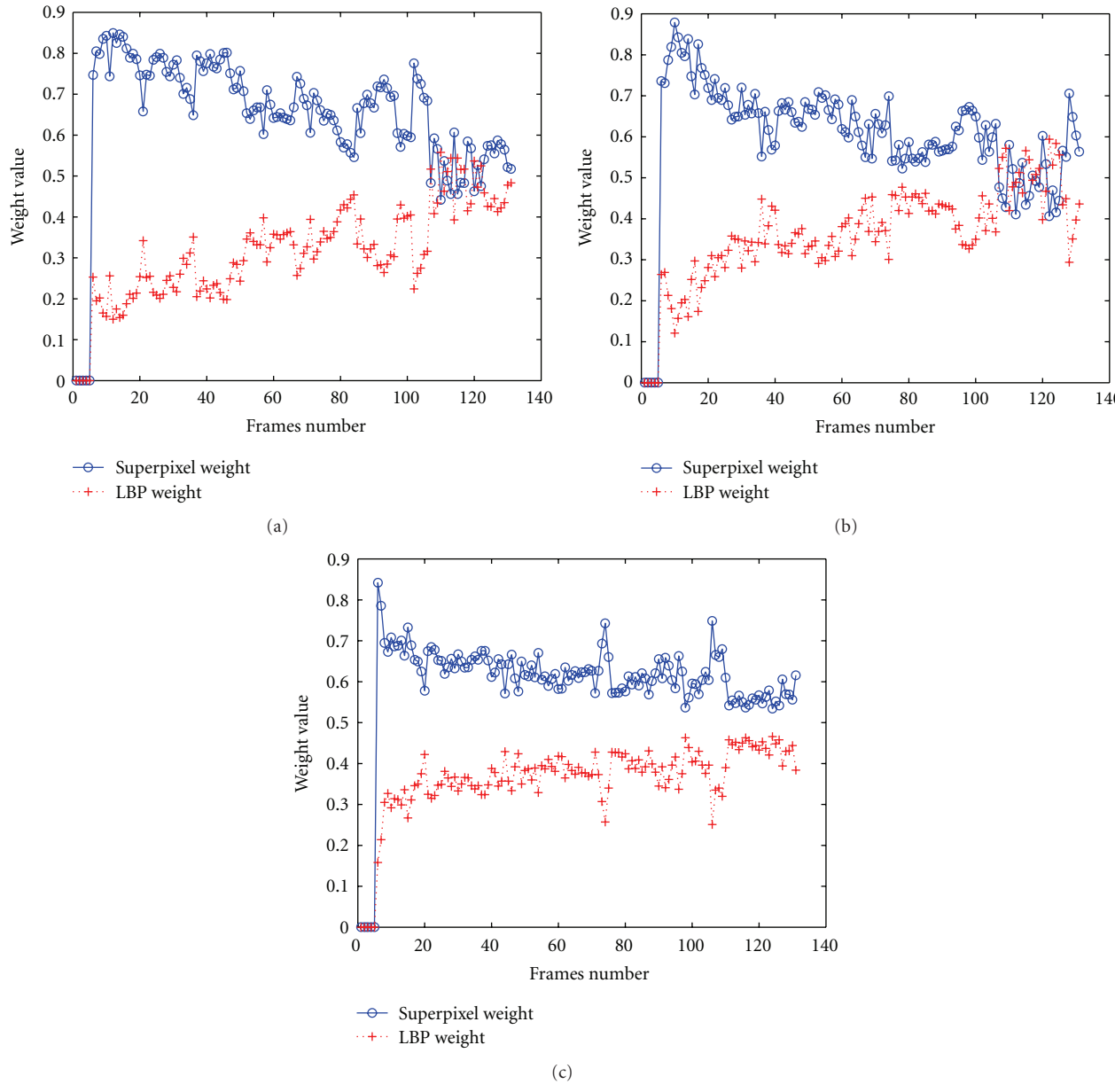


Figure 11: Sequence 1: pedestrians' weight variation curves. (a) pedestrian 1, (b) pedestrian 2, and (c) pedestrian 3.

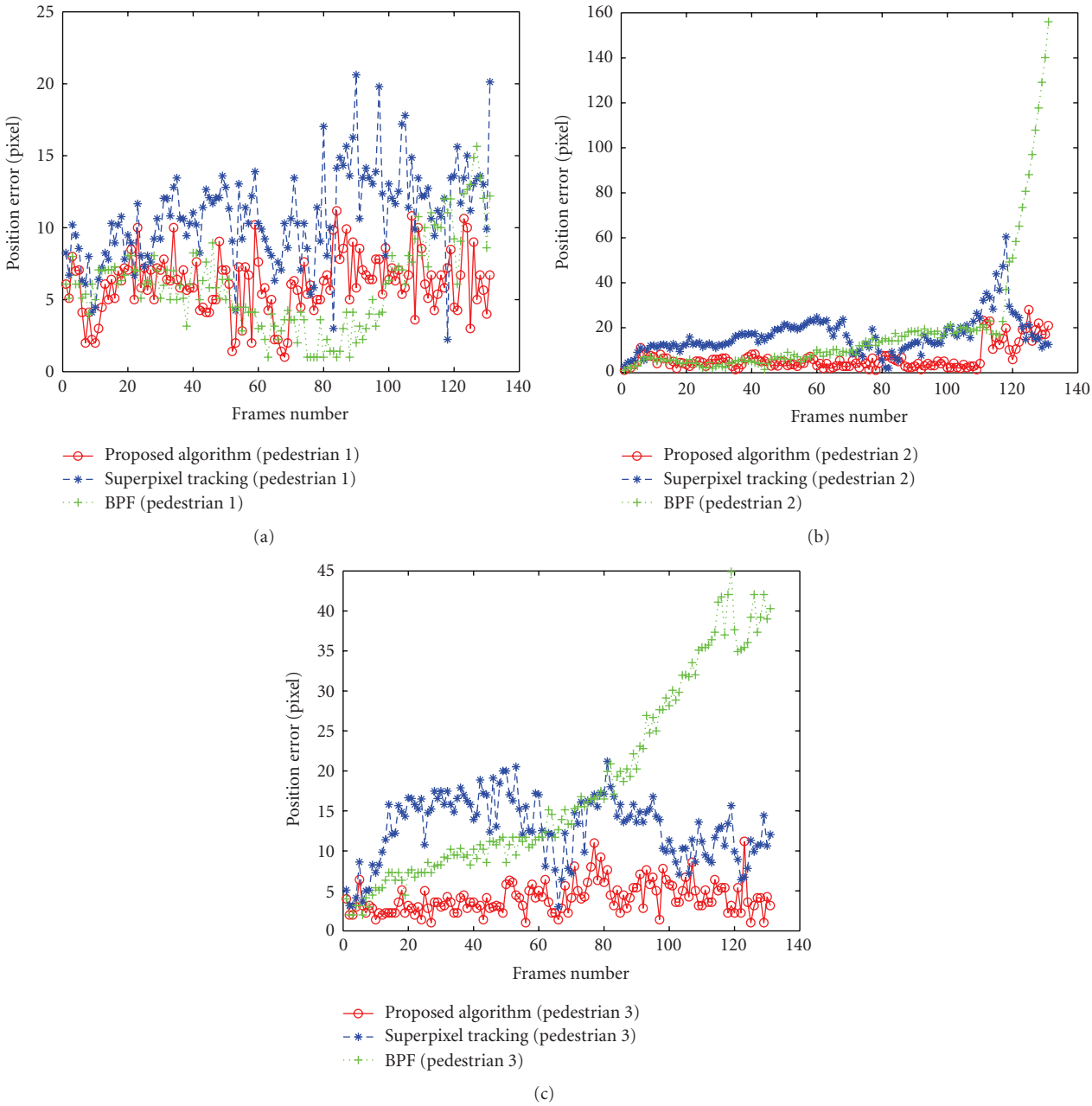


Figure 12: Sequence 1: pedestrians' error curves. (a) pedestrian 1, (b) pedestrian 2, and (c) pedestrian 3.

Figure 13: Sequence 1: target motion trajectories.

Figure 14: Sequence 2: tracking results. The results by our algorithm, superpixel tracking, and BPF methods are represented by solid line, dashed line, and dotted line rectangles. Rectangles in different colors denote the tracking results of different pedestrians.

Figure 15: Sequence 2: target motion trajectories.

Figure 16: Sequence 3: tracking results. The results by our algorithm, superpixel tracking, and BPF methods are represented by solid line, dashed line, and dotted line rectangles. Rectangles in different colors denote the tracking results of different pedestrians.

error; the smaller the $\overline{\text{PositionError}}$, the better the tracking effect.

5.2. More Tracking Results. Our algorithm is tested in more sequences which are acquired from our own dataset, PETS 2012 Benchmark data and CAVIAR database. Tracking results are showed in Figure 20.

It can be seen from the test results of the above three groups of video sequences, the our algorithm has better tracking performances in dealing with complex situations

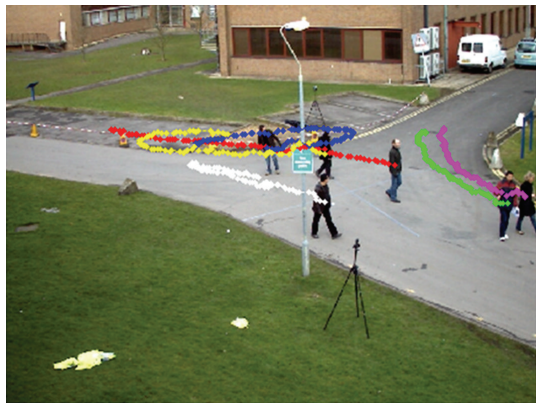


Figure 17: Sequence 3: target motion trajectories.

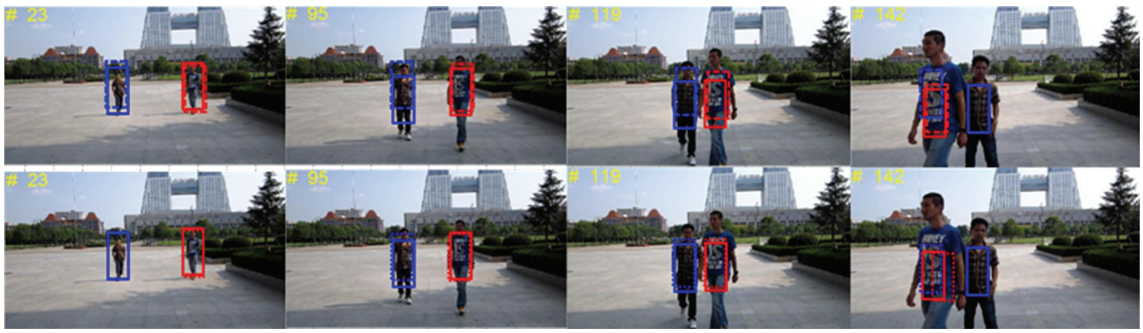


Figure 18: Sequence 4: tracking results. The results of our algorithm, superpixel tracking, and BPF methods are represented by solid line, dashed line, and dotted line rectangles. Rectangles in different colors denote the tracking results of different pedestrians.

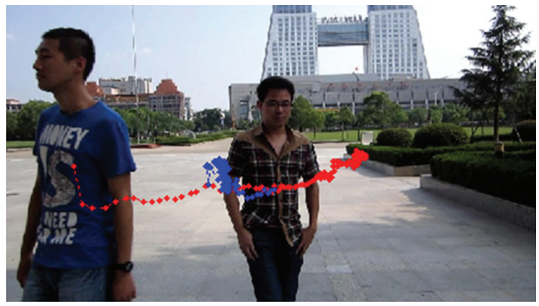


Figure 19: Sequence 4: target motion trajectories.

such as the target's translation, severe occlusion, illumination, and changes of walking speed, as well as analogue interference, and so forth.

6. Conclusions

In this paper, we propose multi-target tracking of pedestrians in video sequences based on particle filters. The contribution of our work can be listed as the following: (1) we apply background subtraction and HOG to getting target regions in training frames rapidly and accurately. (2) Our algorithm builds discriminative appearance model to collect training samples and construct two codebooks using superpixel and LBP features. (3) We integrate BoF into particle filter to get better observation results, and then automatically adjust the weight value of each feature according to the current tracking environment. Our algorithm was tested on a pedestrian tracking application in campus environment. In that case the algorithm can reliably track multiple targets and targets' motion trajectories in difficult sequences with dramatic illumination changes, partial or severe occlusions, and background clutter edges. Experimental results demonstrate the effectiveness and robustness of our algorithm.

Table 3: Tracking average error. The numbers denote average errors of center location in pixels.

Sequences	Superpixel tracking	BPF	Our algorithm
(1) Three pedestrians in the hall	11, 17, 13	6, 19, 18	6, 6, 4
(2) Five pedestrians in the corridor	51, 15, 23, 5, 14	48, 7, 6, 3, 8	8, 6, 10, 4, 3
(3) Sparse crowd	9, 34, 67, 11, 4, 6	35, 31, 102, 3, 4, 3	4, 3, 5, 3, 6, 6
(4) Two pedestrians in the square	17, 38	7, 23	6, 9

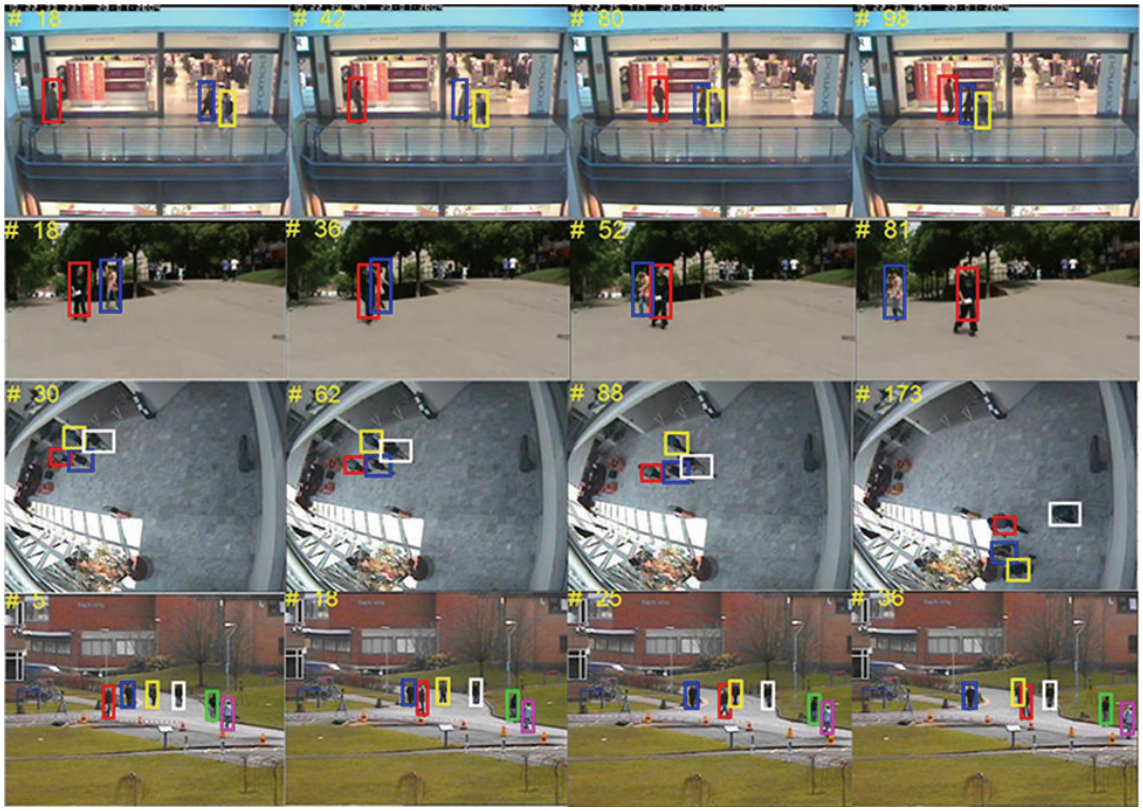


Figure 20: More tracking results of our algorithm.

Acknowledgments

This work was supported in part by the National Science Foundation of China under Grant no. 61170202 and Wuhan Municipality Programs for Science and Technology Development under Grant no. 201210121029.

References

[1] B. Babenko, S. Belongie, and M. H. Yang, "Visual tracking with online multiple instance learning," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2009*, pp. 983–990, June 2009.

[2] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '10)*, pp. 1269–1276, June 2010.

[3] A. Makris, D. Kosmopoulos, S. Perantonis, and S. Theodoridis, "A hierarchical feature fusion framework for adaptive visual tracking," *Image and Vision Computing*, vol. 29, no. 9, pp. 594–606, 2011.

[4] B. Leibe, E. Seemann, and B. Schiele, "Pedestrian detection in crowded scenes," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 878–885, June 2005.

[5] H. Zhou, Y. Yuan, and C. Shi, "Object tracking using SIFT features and mean shift," *Computer Vision and Image Understanding*, vol. 113, no. 3, pp. 345–352, 2009.

[6] D. Comaniciu and V. Ramesh, "Mean shift and optimal prediction for efficient object tracking," in *Proceedings of the International Conference on Image Processing (ICIP '00)*, pp. 70–73, Vancouver, Canada, September 2000.

[7] B. O. S. Teixeira, M. A. Santillo, R. S. Erwin, and D. S. Bernstein, "Spacecraft tracking using sampled-data Kalman filters," *IEEE Control Systems Magazine*, vol. 28, no. 4, pp. 78–94, 2008.

[8] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[9] S. L. Tang, Z. Kadim, K. M. Liang, and M. K. Lim, "Hybrid blob and particle filter tracking approach for robust object tracking," in *Proceedings of the 10th International Conference on Computational Science (ICCS '10)*, pp. 2559–2567, June 2010.
[10] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "An adaptive color-based particle filter," *Image and Vision Computing*, vol. 21, no. 1, pp. 99–110, 2003.
[11] J. Vermaak, A. Doucet, and P. Pérez, "Maintaining multimodality through mixture tracking," in *Proceedings of the 9th IEEE International Conference on Computer Vision*, pp. 1110–1116, October 2003.
[12] K. Okuma, A. Taleghani, N. De Freitas et al., "A boosted particle filter: Multitarget detection and tracking," in *Proceedings of the European Conference on Computer Vision*, pp. 28–39, 2004.
[13] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Proceedings of the European Conference on Computer Vision*, pp. 705–718, 2008.
[14] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "TurboPixels: fast superpixels using geometric flows," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2290–2297, 2009.
[15] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "Slic superpixels," EPFL Technical Report 149300, 2010.
[16] X. Ren and J. Malik, "Tracking as repeated figure/ground segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2007.
[17] S. Wang, H. C. Lu, F. Yang, and M. H. Yang, "Superpixel tracking," in *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1323–1330, 2011.
[18] F. Yang, H. Lu, and Y. W. Chen, "Bag of features tracking," in *Proceedings of the International Conference on Pattern Recognition (ICPR '10)*, pp. 153–156, August 2010.
[19] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, pp. 886–893, June 2005.