Multi-object tracking with inter-feedback between detection and tracking

Shu Tian, Fei Yuan, Gui-Song Xia

ABSTRACT

Multi-object tracking is an important but challenging task in computer vision. Tremendous investigations have been made on the topics, among which tracking-by-detection method first detects objects independently at each frame and then links the detected objects into trajectories. One shortcoming of this method, however, lies in the fact that it regards detecting and tracking as two separated processes and the tracking information are not used in detection, which often results in many false and missing detections and involves heavy computational complexity. In order to solve this problem, this paper proposes a multi-type multi-object tracking algorithm, by introducing on-line inter-feedback information between the detection and tracking processes into the tracking-by-detection method. Our tracking algorithm consists of two iterative components: detection by feedback from tracking and Tracking based on detection. In the detection step, objects are detected by the detectors adjusted by information from tracking. In the tracking step, we use group tracking strategy based on detection. Moreover, in order to handle tracking scenarios with different complexity, objects are classified into two categories, i.e. single object and multiple ones, and are dealt with different strategies. The proposed algorithm is evaluated on several real surveillance videos and achieve higher performance in contrast to the state-of-the-art methods. Besides the high precision, it also has demonstrated that the proposed algorithm needs less detector and searching scale and can run in real time for many tracking applications.

1. Introduction

Multi-object tracking is one of the most important tasks in computer vision. Its application is distributed widely, such as motion and scene analysis, video indexing, human-computer interaction, video surveillance and traffic monitoring, among which traffic video surveillance motivates most of the investigations on multi-object tracking. In contrast to object tracking in other scenarios, tracking objects in traffic videos has many difficulties: (1) the shapes of objects are complex and different; (2) too many objects are crowded with complex occlusion relationships; (3) the computational complexity is often very high.

One of the problems of object tracking is how to identity the objects. A recent popular method is to run an object detector trained off-line or online. Thanks to the significant progresses achieved on object detection in recent years, tracking-by-detection method becomes one of the most popular algorithm for the task of multi-object tracking, see [1–3]. In tracking-by-detection approach, objects are first detected, usually, by a detector trained off-line. Trajectories are then estimated by linking these detected objects within a temporal window through certain optimization algorithm. By this means, some temporal fractures of trajectories can be recovered and drift can be alleviated.

Though tracking-by-detection method can be utilized immediately for tracking objects in traffic videos, it takes detecting and tracking as two separated processes, which often leads to many unsatisfied tracking results. What is more, due to diversity and complex occlusions of objects in traffic videos, the detection of all objects often requires much more detectors. The increasing of detectors usually results in more false and missing detections and decreases the efficiency of the tracking algorithm.

To overcome these difficulties, we take the advantage of information obtained from tracking to feed back to the detection. Our tracking algorithm consists of two iterative components: detection by the feedback from tracking and tracking based on detection. In detection step, for a frame, we first use background subtraction and connected domain to get candidate regions. Then the predicted object position gotten from tracking is utilized to expand the scale of candidate regions. Finally, objects are detected...
by offline-trained detectors whose search region and scale are revised by feedback information from tracking. Observe that, once the type, size and position of objects are predicted, the search region and scale of detectors could be much smaller. Then the false detection rate and running-time of detectors can be reduced.

While, the tracking step is achieved by linking detected objects into trajectories. The key point is to classify objects into two categories: single object and multi-objects. Single object is dealt with simple strategy, while multi-objects are grouped into different clusters according to certain similarity measurements and each cluster is then treated separately. To eliminate effect of noises and collect more precise information fed back to the detection, a trajectory initialization phase is designed. Trajectory rematch is another important phase for recovering from temporal fracture of trajectories.

All the points mentioned above form a multi-type multi-object tracking algorithm, called MTDT for short, which can work in real time. The main contributions of our work are summarized as follows:

- We introduce a mechanism to integrating inter-feedback information between detection and tracking into multi-object tracking. Thanks to this feedback, detection is done with much more information such as object's type, size and predicted position.
- We use a group tracking strategy to speed up the tracking process. More precisely, we classify the objects into two groups: single object which is far away from other objects and multi-objects which are close to each other. According to their complexities, we then track them by different strategies, which leads to higher detection precision and reduces the running time of tracking.

The rest of this paper is organized as follows. Related work are reviewed in Section 2. Section 3 describes the proposed tracking method. Section 4 presents experiments on several challenging videos. Several final remarks are presented in Section 5.

2. Related work

In the past decade, a large amount of work has been devoted to multi-object tracking [4,5]. Not only the tracking algorithm is reviewed [6], but the appearance models in visual object tracking includes the recursive method, by which the current object's position is predicted by the information of previous frame. Kalman filter [8] is a recursive method, it and is optimal with the assumption of Gaussian probability distributions [9]. But real-world cases are usually non-Gaussian. So later, particle filter, also known as Sequential Monte Carlo method, was first introduced to single object tracking [10], and then applied to multi-object tracking [11–13]. Particle filter samples from proposal distribution to get a group of weighted particles for representing current status. These methods are fast but easy to be drift.

In recent years, impressive progress has been made on object detection [14–16,57] and it also has an enormous influence on multi-object tracking. Detection has been introduced into particle filters to reduce drift. For example, Okuma et al. [17] introduced the Haar feature-based cascade classifier to particle filtering framework. They used the classifier as a detector to discriminate the object and background. The particles are adjusted according to the detection results. Michael et al. [18] proposed a multi-object tracking algorithm based on particle filter and detection score, introducing detection score into the calculation formula of the matching score in data association.

Then, tracking-by-detection method, which can track objects after detecting, has become a popular multi-object tracking method [19,20]. The basic idea of tracking-by-detection approach is to detect objects frame-by-frame with an off-line trained detector, and then make a data association of the detection results within a certain time window. The recent method tried to model the multi-target, multi-frame data association problems into network flow [21], the nodes of graph are the candidate positions of objects. Edges connecting the nodes are all possible pairs of the candidate positions of objects, which directs from previous frame to the next frame. Costs of edge are functions only of the candidate pair of observations. Since this problem has been proved to be NP-hard [22], the edge of the graph is further simplified to all possible matches of object's candidate positions in successive frames. In this way, the result of data association can be calculated by min-cost network flow method. In order to speed up the algorithm, Pirsaviash et al. [23] put forward a more effective shortest path algorithm to solve the min-cost flow problem, while Berclaz et al. [24] proposed a linear programming based algorithm to get optimal solution with polynomial complexity. In order to add high order smooth constraints, Brendel et al. [25] formulated the data association as a maximum weight independent sets (MWIS) problem. They first got tracklets, then learned a distance to link all the tracklets into complete trajectories. Thus, MWIS method partly solves occlusion by hierarchically combining tracklets into trajectories, relying on the appearance and motion. Butt et al. [26] proposed a solution based on three-frame tracklets to leverage constant velocity motion constraints. However, tracking-by-detection method is difficult to achieve real-time performance, though it has the advantages of handling complex images, alleviating drift and processing temporary disappearance of objects.

Some methods try to track objects in the real time. Benfold et al. [27] combine asynchronous HOG detections with simultaneous KLT tracking and Markov-Chain Monte-Carlo Data Association (MCMCDA) to track objects. But they develop an asynchronous multi-threaded architecture to meet the demand of real-time. As detection stage is the bottleneck, using other method instead of detection is also a solution to reduce computational time. Posssegger et al. [28] generate an occupancy volume based on the local mass densities of the 3D visual hull reconstruction instead of detection for robust tracking using particle filtering in combination with Voronoi partitioning. Another way to reduce computational burden is to detect objects on Regions-Of-Interest (ROI) but not the whole image in each frame. Gavrila et al. [29] integrate four modules into a system: (sparse) stereo-based ROI generation, shape-based detection, texture-based classification and (dense) stereo-based verification. The ROIs in their system are generated in each frame using stereo depth information. Background subtraction is an efficient way to reduce the complexity of the tracking algorithm [30], too. Simple techniques include using a fixed image [31] or the average of some frames [32] as the background model. Background subtraction using Gaussian Mixture Model (GMM) is very popular [33,34] and has been improved by many researchers [35,36]. Compared to GMM, visual background extractor (ViBe) [37] is more simple, faster and easier to tune parameters.

Our method is an improvement of tracking-by-detection method focusing on increment of runtime performance. Object detection is still a difficult problem far from completely resolved due to many difficulties such as complex object occlusion, changes of object scales, variations of object pose, noises of image, and processing time limitation of the algorithm. In order to improve the detection performance in multi-object tracking and revise tracking-by-detection method to adapt to traffic video surveillance, combining ROIs extraction and background subtraction, we introduce inter-feedback between detection and tracking.
3. System review

We propose a multi-type multi-object tracking algorithm, called Multi-object Tracking with inter-feedback between Detection and Tracking (MTDT), which is on-line and can work in real time. The information of tracking is utilized for detection in our method. Thanks to this strategy, detection is more accurate and fast. We first formulate our method and then show its implementation.

3.1. Formulation

Denote \( D = \{ D_i \} \), \( i = 1, \ldots, n \) as all the \( n \) detections, and let \( T = \{ T_j \} \), \( j = 1, \ldots, m \) be all the \( m \) trajectories, each of which consists of detections \( T_j = \{ D_{j,t} \} \), \( t = t_{j,1}, \ldots, t_{j,e} \). \( D_{j,t} \) is the detection of the \( j \)th trajectory in the \( t \)th frame, and \( t_{j,1} \) and \( t_{j,e} \) are the starting and the ending frame of \( T_j \), respectively. The solution of tracking-by-detection method can be achieved by maximizing a posteriori

\[
T^* = \arg \max_T P(T \mid D).
\] (1)

In order to achieve on-line tracking, we need to make the algorithm real-time. So, in the \( t \)th frame, we only have the information before it. We define the trajectories in the \( t \)th frame in a simpler form for clarity:

\[
T_t = \{ T_{t,k} \}, \quad k = 1, \ldots, M_t,
\] (2)

where \( M_t \) is the number of trajectories in the \( t \)th frame. For each frame \( t \), our solution is

\[
T^*_t = \arg \max_{T_t} P(T_t \mid I_{t+1}).
\] (3)

In order to simplify the problem, Markov assumption is firstly made and a post-processing is added to use the lost information because of the Markov assumption to help tracking. Thanks to Markov assumption, the state in current frame \( t \) only depends on the state in previous one \( t-1 \). When we process frame \( t \), only the trajectories in frame \( t-1 \) and the image in the \( t \)th frame \( I_t \) are involved in the calculation. We introduce the detection stage

\[
P(T_t \mid I_t) = P(T_t \mid I_t, T_{t-1})
\]

\[
= \int P(T_t \mid O_t, I_t, T_{t-1}) P(O_t \mid I_t, T_{t-1}) \, dO_t
\] (4)

There are two important terms of probability. \( P(O_t \mid I_t, T_{t-1}) \) indicates how realistic the detection in frame \( t \) is, and the detection is gotten by utilizing the information from the tracking process. \( P(T_t \mid O_t, I_t, T_{t-1}) \) describes how well the detections in frame \( t \) matches trajectories in frame \( t-1 \).

After generating detection and calculating similarity between trajectory and detection, we maximize the similarities via Hungarian algorithm to match the detections and trajectories. Finally, the confirmation is made by pruning detection and trajectory pairs with the small similarities.

3.2. Outline of our approach

To realize our system, we segment our system into two phases: detection’s generation based on tracking and Tracking based on detection. The system overview is shown in Fig. 1. The components written in red are our main contributions.

In general, we first generate detections by feedback from tracking for each frame, then link these detections into trajectories. In the phase of detection, background subtraction and connected domain are used to speed up and reduce false detections. Then detectors are utilized to get final results based on objects’ type, size and probable location gotten from tracking phase. In the phase of tracking, we first classify objects into single-object and multi-object, then calculate the similarity between trajectory and detection. Then, Hungarian algorithm is used to link detections into trajectories. The detections and trajectories with no correspondence are handled separately.

Having introduced the basic method, we now turn to the occlusion processing which is the main reason why we classify objects. There are three types of occlusions: self-occlusion, inter-object occlusion, and occlusion by the background scene structure:

1. Self-occlusion often occurs when tracking articulated objects, where one part of object occludes the other parts. The occlusion of this type is assumed to be non-occurrence in our situation because all objects are assumed to be rigid of which car and motor are actually rigid and pedestrian can be seen as rigid due to its small changes between two successive frames.

2. Inter-object occlusion occurs when the targets occlude each other. It is most frequently occurred in our situation. We will discuss the processing mechanism latter.

3. Occlusion by the background scene structure occurs when objects are occluded by trees or other structures in the background. Our solution is to sentence the occluded objects to be disappeared and rematch them in next few frames.

We introduce trajectory rematch in Section 5.4.

Now, we discuss how to handle inter-object occlusion. First, we classify all objects into two categories: single object and object of multi-object. Single object means that there exits certain distance between the object and the others. Object of multi-object means that the object is one of some objects which are close to at least one of them. These two types of objects are handled separately.

We use \( T^*_t \) and \( T_t^m \) to represent single object and object of multi-object in frame \( t \), respectively. Eq. (4) is the base of processing for both \( T_t^1 \) and \( T_t^m \), but the matching processing is a little different.

We describe them in detail in the successive sections.

4. Detections by feedback from tracking

We first introduce detections by feedback from tracking which is the main contribution of our work. In other words, \( P(O_t \mid I_t, T_{t-1}) \) is described in detail in this section.
4.1. Formulation

We define

\[ P(C_i|l, t_{t-1}) = \begin{cases} P_{bg}P_{pr}P_{det} & \text{if the category of object is multi-objects} \\ P_{bg}P_{pr} & \text{or the trajectory is initializing} \\ 0 & \text{otherwise} \end{cases} \]

where

\[ P_{bg}(C_i|l, t_{t-1}) = \begin{cases} 1 & \text{if } C_i \text{ is in foreground regions by background subtraction} \\ 0 & \text{otherwise} \end{cases} \]

\[ P_{pr}(C_i|l, t_{t-1}) = \begin{cases} 1 & \text{if } C_i \text{ is in the predicted regions by trajectories's location} \\ 0 & \text{otherwise} \end{cases} \]

\[ P_{det}(C_i|l, t_{t-1}) = \begin{cases} 1 & \text{if } C_i \text{ is gotten by the detectors from tracking} \\ 0 & \text{otherwise} \end{cases} \]

It means that a region is considered as a detection result when it falls in predicted region and foreground region at the same time even if it is not classified into object by the detector whose type is from tracking when the object is a single object. Detector is needed when the object’s category is multi-objects or the trajectory is initializing. The reason is that multi-objects are complex situations and trajectory initialization is an important phase to collect object’s information. This process can improve the speed of the algorithm.

There are much useful information in \( T_{t-1} \) which is the trajectory set in frame \( t \). The most meaningful one is the objects’ positions in each frame. The objects’ positions in frame \( t \) are calculated according to them. We use Kalman filter to estimate an object’s position when its temporal length is short (less than 5 frames). When an object stays in video for a long time, we smooth its trajectory through algorithm reported in [38], calculate the average velocity in recent frames, and predict the object’s position in frame \( t \) by adding average velocity to object’s position in frame \( t-1 \). The size of object should be similar to the size of object in frame \( t-1 \).

4.2. Background subtraction

In this section, we describe how to calculate \( P_{bg} \). Background subtraction is the first phase of our system for reducing the complexity. In this work, ViBe [37], a recently proposed approach, is employed in our background subtraction process, and this approach has proven to be an effective and efficient background subtraction method by many researchers. ViBe is a pixel-based background subtraction approach which uses random policy. There are four components of ViBe described as follows:

**Background model definition:** Let \( v(x) \) be the pixel value of position \( x \). Let \( M_b(x) \), a collection of \( N_v \) background pixels sampled in position \( x \) or its neighborhood in previous frames, be the model of a background pixel, i.e.

\[ M_b(x) = \{v_1, v_2, \ldots, v_{N_v}\}. \]

**Pixel classification:** A pixel \( v(x) \) is considered as a background pixel when

\[ \#(S_b(v(x)) \cap M_b(x)) > \#_{\min} \]

where \( \#_{\min} \) is a given threshold, \( \#\{\cdot\} \) is the cardinality of set \( \cdot \) and

\[ S_b(v(x)) = \{v| \|v - v(x)\| < R\}. \]

The other pixels are considered as foreground ones.

**Background model initialization:** The model of background pixel is initialized from a single frame. Define \( N_s(x) \) as the spatial neighborhood of position \( x \), \( M^0_b(x) \) as the initial background model:

\[ M^0_b(x) = \{v(y_1), v(y_2), \ldots, v(y_{N_v})\}, \]

where \( y_i, i = 1, 2, \ldots, N_v \) are randomly chosen from \( N_s(x) \).

**Background model update:** For a background pixel \( v(x) \), we use a random pixel in \( N_s(x) \) to update a random element in the model at random frame. The time subsampling factor is \( \phi \) which means that the model updates with the probability of \( 1/\phi \) for a pixel at one frame.

4.3. Connected domain

After the background subtraction, we get the foreground mask which indicates whether each pixel belongs to foreground or not. From Fig. 2, we can see that the results of background subtraction are very noisy. In order to overcome this problem, we use morphological opening operation to suppress this type of noise. Then, the next step is to segment foreground pixels into regions naturally. Thus, not only the objects’ positions, but also their size and shape information can be utilized to detect and track.

4.4. Detector

In this section, we describe how to calculate \( P_{det} \). In this work, we choose to utilize the classification based detection approach proposed by Dalal and Triggs [14], in which histograms of oriented gradients (HOG) feature are extracted to represent moving objects and support vector machine (SVM) is employed as the classifier. The effectiveness of this approach in detecting pedestrians and cars has been shown in applications. We considered three types of object: car, motor and pedestrian. So we train three detectors off-line. And each detector is used once to detect one region.

5. Tracking based on detection

After getting the detection result, we describe how to do tracking in detail. There are 4 subsections. The first is grouping, thus we can handle different situations separately. After grouping, we can calculate the similarity between trajectory and observation, then observations...
could be linked to a trajectory. The third is to initialize the trajectories. The last is to deal with trajectory disappearance and rematch.

5.1. Grouping

For every trajectory, we first compute its predicted positions. Then we apply a cluster algorithm described below to group them into different clusters. The clusters with more than one trajectories are the situations of multi-objects. The others are the situations of single object which is far from all the other objects.

The situations of single object and multi-objects are processed separately. The situations of multi-objects are clustered to many single object which is far from all the other objects. The clusters with more than one trajectories are the situations of multi-objects. The others are the situations of single object. For trajectory and a detection, the possibility based on the appearance model, Pl, is the possibility based on the type of object, and Ps is the possibility based on the appearance model.

5.2. The similarity between trajectory and detection

In this paper, we simplify P(T | C_{t-1}, l_{t-1}, T_{t-1}) as the product of some possibilities. For a trajectory and a detection, P(T | O_{t}, l_{t}, T_{t-1}) = P_{a} \cdot P_{l} \cdot P_{t} \cdot P_{s}.

5.2.1. Appearance model

We use an 8 \times 8 RGB color histogram as appearance model. For each detection D_{t}, the histogram is defined as

\[ H_{rgb}(D_{t}) = \{h_{j}^{N_{rgb}}\}_{j=0}^{N_{rgb}-1}, \]

where \( N_{rgb} \) is the total number of bins and \( h_{j} \) is the value of \( j \)th bin:

\[ h_{j} = \frac{1}{N_{rgb}} \sum_{k=0}^{N_{rgb}-1} k(x, c)r(x, j), \]

where \( C_{j} \) is a normalization parameter to guarantee all values of bins are summed to one, \( c \) is the center of interested region, \( k(x, c) \) is a weighting function which is set to 1 in our case and \( r(x, j) = 1 \) if the index of pixel's bin is \( j \), \( 0 \) otherwise.

The Hellinger distance, which is related to Bhattacharyya coefficients, is used to compare color models:

\[ d(H_{rgb,1}, H_{rgb,2}) = \sqrt{1 - \frac{1}{\sqrt{H_{rgb,1}H_{rgb,2}}} \sum_{q=0}^{N_{rgb}-1} \sqrt{h_{q,1}h_{q,2}}}, \]

where \( H_{rgb,1} \) and \( H_{rgb,2} \) are two histograms, and

\[ H_{rgb,k} = \frac{1}{N_{rgb}} \sum_{q=0}^{N_{rgb}-1} h_{q,k}, \quad k = 1, 2, \]

For each trajectory, the appearance model is updated frame by frame. The appearance in current frame is the most important one, so we update as follows:

\[ H_{rgb,k+1} = (1 - \alpha)H_{rgb,k} + \alpha H_{rgb,t}, \]

where \( H_{rgb,k} \) is the histogram after updating in kth frame, \( \alpha \) is a constant which is set to 0.8, and \( H_{rgb,t} \) is the histogram calculated in current frame.

The Hellinger distance is not used to evaluate similarity between trajectory and detection directly. The distance is assumed to fit a normal distribution with zero mean. So,

\[ p_{a} = \frac{1}{\sqrt{2\pi \sigma_a}} \exp \left( - \frac{d(H_{rgb}(D_{t}), H_{rgb,t})^2}{2\sigma_a^2} \right), \]

where \( H_{rgb}(D_{t}) \) is the histogram of detection in frame \( t \), \( H_{rgb,t} \) is the histogram of trajectory after updating in frame \( t \), \( d(\cdot) \) is the function to calculate Hellinger distance, and \( \sigma_a \) is a given threshold.

5.2.2. Dynamic model

In the section of detections by feedback from tracking, we have shown how to calculate the predicted position of trajectories. Then, object's position is assumed to be changed only due to Gaussian noise:

\[ x_{t} \sim N(x_t^0, \Sigma_t), \]

where \( x_{t} \) is the object's position in frame \( t \), \( x_t^0 \) is the object's predicted position in frame \( t \), and \( \Sigma_t = \text{diag} (\sigma_x^2, \sigma_y^2) \) is the covariance matrix for position. Then,

\[ P_t = P_s(x_t), \]

where \( x_{t} \) is the position of detection in frame \( t \), and \( P_s \) is the Gaussian distribution described above.

5.2.3. Object's size model

There are some features to represent the size of object: width, height, area, and aspect ratio. We let \( w_{t}, h_{t}, a_{t}, \) and \( d_{t} \) be the width, height, area, and aspect ratio of region of an object in frame
5.3. Multi-objects handling

Assignments between the trajectory and detection occurs in several frames. These frames are key frames in which we collect the trajectory's information to help tracking in successive frames. The most important information is the type of object. We train classifiers off-line, then use the classifier to classify the foreground of the object for several frames. Accumulating the scores of classifier, we choose the type with most scores as the type of object.

5.4. Trajectory initialization

There are many noises among the detections. A trajectory is considered to be meaningful when it can be constantly tracked for several frames. These frames are key frames in which we collect the trajectory's information to help tracking in successive frames. The most important information is the type of object. We train classifiers off-line, then use the classifier to classify the foreground of the object for several frames. Accumulating the scores of classifier, we choose the type with most scores as the type of object.

5.5. Trajectory disappearing and rematch

Our goal is to get good performance on real challenging traffic surveillance videos. So we cannot assume that objects must disappear at some place. The only assumption is that objects are considered disappeared when they satisfy the following:

- They are near the edge of the image.
- They move toward the edge of the image.
- They are not matched with any detections.

In order to improve the robustness of our tracking system, especially to deal with lost information due to Markov assumption, when an active trajectory is not matched with any detections but it does not satisfy the other conditions, we push this active trajectory to the history trajectory list instead of considering the trajectory disappeared. Only if a history trajectory is not matched to any detections for several frames, it is considered to be disappeared.

6. Experiments

6.1. Data set and evaluation metrics

For now, there is no generally accepted benchmark available for multi-type multi-object tracking in traffic video. We need to deal with real situation that cars, motors and pedestrians appeared at the same time in the scene. We carry out experiments on real challenging traffic surveillance videos. All videos are annotated via ViPER (Video Performance Evaluation Resource) tools [39]. The size of image is $704 \times 576$ pixels.

The sequences we used are all taken from the static cameras. We assume that no scene knowledge is known such as the scale of objects, the location of the road, the entry/exit zones of objects, and the ground plane calibration. The first sequence is called crossroad. The second sequence is called straight-road. We segment straight-road into 13 clips which are focused on occlusion. We just care about the performance for occlusion on these 13 clips because the ghosts of other objects are not suppressed due to the short length of clips.

For comparing with more state-of-the-art methods, we evaluate our method on several challenging, publicly available video sequences. Some videos belong to PETS 2009/2010 benchmark [40,41]. Among these videos, S2L1 is one of the most widely used sequences in the previous literature. We show the results on this video separately. The TUD-Stadtmitte sequence [42] is another widely used sequence which is filmed in a busy pedestrian street. Because all the objects of the sequences are pedestrians, we do not consider about multi-type issue in this experiment.

Though [43] shows that evaluating multi-target tracking is surprisingly challenging for various reasons, we still need to use some existing metrics to evaluate our algorithm. We follow the evaluation metrics in [44,45] because a large amount of metrics are needed. All the metrics are listed in Table 1. The issue should be noted is that how to judge whether a region of ground truth and a region of tracking result matched or not. We define...
The threshold of detector is set a little lower in order to improve the recall of the detection. For the experiments on real surveillance videos, detector’s result is generated by three different ways. The first is to use ground truth as the detector’s result denoted as GT. It is not in accord with the real situation, but it is easy to see the possibly best performance of the tracking algorithm. The second is to detect on the foreground denoted as DF. Since our algorithm is based on the background model, the compared tracking algorithm should be adapted to foreground for fairness. The third is to detect on the whole image denoted as DI. This is the default way required by tracking algorithms to be compared with. Our key innovation is to take advantage of tracking information to detect more quickly and precisely, so detection and tracking cannot be separated in our system. The only way to generate detector’s result for our system is to detect with the help of tracking information.

All the parameters of tracking are set experimentally and not changed in the most experiments. But for the experiments on real surveillance videos, some videos are very hard due to the high density of objects. We set different parameters on these videos. For example, for an algorithm outputs an empty result, both MOTA and IDs would be zero. If we only observe IDs, we will get a wrong conclusion that this algorithm is the best. We suggest using both MOTA and IDs.

### Table 2

<table>
<thead>
<tr>
<th>Method</th>
<th>MOTA</th>
<th>MOTP</th>
<th>Prcn</th>
<th>RcII</th>
<th>FAR</th>
<th>GT</th>
<th>MT</th>
<th>PT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDs</th>
<th>FM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ours</td>
<td>43.5</td>
<td>60.9</td>
<td>71.3</td>
<td>74.1</td>
<td>2.31</td>
<td>23</td>
<td>12</td>
<td>11</td>
<td>0</td>
<td>1676</td>
<td>1469</td>
<td>60</td>
<td>148</td>
</tr>
<tr>
<td>BPF(GT)</td>
<td>−72.5</td>
<td>26.4</td>
<td>4.6</td>
<td>3.7</td>
<td>5.94</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>4310</td>
<td>5466</td>
<td>11</td>
<td>168</td>
</tr>
<tr>
<td>BPF(DF)</td>
<td>89.1</td>
<td>32.3</td>
<td>4.9</td>
<td>4.9</td>
<td>7.34</td>
<td>23</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>5322</td>
<td>5399</td>
<td>13</td>
<td>72</td>
</tr>
<tr>
<td>BPF(DI)</td>
<td>−223.9</td>
<td>32.1</td>
<td>1.6</td>
<td>3.8</td>
<td>17.81</td>
<td>23</td>
<td>0</td>
<td>1</td>
<td>22</td>
<td>12,911</td>
<td>5459</td>
<td>12</td>
<td>66</td>
</tr>
<tr>
<td>DC(GT)</td>
<td>59.1</td>
<td>41.5</td>
<td>86.6</td>
<td>70.2</td>
<td>0.85</td>
<td>23</td>
<td>11</td>
<td>8</td>
<td>4</td>
<td>618</td>
<td>1690</td>
<td>12</td>
<td>35</td>
</tr>
<tr>
<td>DC(DF)</td>
<td>16.2</td>
<td>43</td>
<td>83.4</td>
<td>20.6</td>
<td>0.32</td>
<td>23</td>
<td>1</td>
<td>6</td>
<td>16</td>
<td>233</td>
<td>4506</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>DC(DI)</td>
<td>26.7</td>
<td>38.8</td>
<td>31.2</td>
<td>21.9</td>
<td>3.77</td>
<td>23</td>
<td>1</td>
<td>6</td>
<td>16</td>
<td>2731</td>
<td>4434</td>
<td>28</td>
<td>32</td>
</tr>
</tbody>
</table>

### 6.3. Experimental result on real surveillance videos

We compare our method with two state-of-the-art algorithms: Boosted Particle Filter (BPF) [17] and the method proposed by Milan et al. [46] which is called DC for simplicity in this paper. There are two experiments. The first experiment is carried on a long time sequence for evaluating the performance of tracking algorithms quantitatively by some metrics. The second experiment is carried on 12 clips which are focused on occlusion.

The parameters of ViBe are tuned well by the author. The default parameters are suitable for most situations. But we hope the objects not to merge too fast, so we set \( N_v = 20, R = 12, h_{\text{min}} = 2 \) and \( \phi = 16 \). The fact that ViBe needs some frames to initialize means that there is also an initialization process in our method. It is unfair for our method as the lengths of all the sequences are short. So we generate a background image by considering the ground truth to initialize the ViBe model. Then the objects are tracked from the first frame.
6.3.1. Quantitative evaluation on real surveillance videos

Table 2 shows the quantitative evaluation results. Fig. 3 shows some screenshots of tracking result. Because of “drift”, the performance of BPF is very poor no matter what would be the detector’s result. DC is a recent state-of-the-art method. It achieves the best performance at most metrics when ground truth is used as detector’s result. But actually in video surveillance, we cannot get ground truth before tracking. DC (DF) and DC(DI) are the actually used methods. The performance of DC(GT) is the upper bound of DC. The performance of DC(DF) and DC(DI) are much appropriate to compare with our method since the ground truth is not used in our method. We can see that our method achieves best score at most metrics. FP of DC is low, and FN of DC is high. This is the feature of DC. But there is a fine balance between FP and FN in our method. The score of Rcll and Prcn is similar.

One point to explain is the negative number of MOTA. The definition of MOTA can be found in Table 1. If the total sum of the number of misses, the number of false positives and the number of mismatch errors for all frames are bigger than the total number of objects for all frames, then MOTA would be a negative number. The main reason of negative number’s emerging in our experiments is that the number of false positive is too many.

The number of IDs and FM of our method is a little big. This is because that our method covers more trajectories which can be seen from the higher Rcll of our method. So IDs and FM of our method are easily more than other algorithms. DC(GT) is special. The Rcll of CD(GT) is high too, but IDs and FM are low. The reason is that the performance of DC is improved much due to the ground truth being used as the detector’s results.

6.3.2. Qualitative evaluation on real surveillance videos

This experiment is carried on 12 clips which are focused on occlusion. We just annotate the inter-occlusion objects at a few frames before and after the occlusion for evaluating algorithms’ performance on occlusion. From Fig. 4, we can see that the objects could be rematched after the occlusion. The performance of BPF and DC is lower than our method.

6.4. Experimental result on public available videos

We report the results of some state-of-the-art methods: DC [46], KSP [24], Milan’s method [47], GMCP [48]. In Table 3, “–”
means no data reported in the previous literatures. From Table 3, we can see that our method performs good but not the best. DC achieves best on almost all the videos. It is reasonable that DC cost much more time than our method to detect and track. Compared with KSP, our method performs better on PETS-S2L1, TUD-Stadtmitte, PETS-S3-MF1, PETS-S2L2, PETS-S2L3, but worse on PETS-S1L1-2, PETS-S1L2-1. We can conclude that our method is a little better than KSP. It means that our method approaches a comparable performance, at the same time, our method is, on average, real-time.

A significant phenomenon is that the MOTA of our method is more closer to the MOTA of DC on PETS-S2L1, PETS-S3-MF1 and PETS-S2L3, but much worse on the other videos. The first reason is that the objects of these videos are relatively low dense (the density of objects of PETS-S2L3 is high, but some persons are alone). But our detector performs not very good when the objects are dense.

The performance of the detection on the whole image is shown in Table 4. The definitions of metrics Prcn and Rcll are listed in Table 1. Compared with the detection, our methods get a better performance than the detection on the whole image. The fact demonstrates the detection benefit to the feedback from tracking in our method. Fig. 5 shows visual results of our approach.

6.5. Runtime performance

The essential demand of our system is that the whole algorithm consisting of detection and tracking should run in real-time. Even if the algorithm gets a high performance on detection and tracking, it is still considered failed if it runs slowly. So the runtime performance is crucial for our method.

There are two experiments. The first one is evaluation methods on the videos we collect. All the related runtime performances are shown. The second is evaluation methods on the benchmark. As we report all the performance reported in other literature but not gotten by running the source code, we cannot report the runtime performances. Thus, we just analyze our own method.

6.5.1. Evaluation on our videos

Our system has been implemented in VC++ using the OpenCV library on a 3.4 GHz Quad-Core Intel Core i7 CPU with 8 GB RAM. The tracking times of algorithms are different depending on the number of detections of a sequence which are listed in Table 6. The detection time is another related metrics which is listed in Table 5. BPF and DC both are detection-based tracking algorithm. The detection of all images of the sequence is done before the
beginning of the tracking algorithm. The total processing time of BPF and DC is the sum of detection time and tracking time. The total processing time of our algorithm has nothing to do with the detection time listed in Table 5, because the detection and tracking are fused to the unified framework. Table 7 shows the total processing times including detection and tracking.

From Table 7, we can see that our algorithm runs, on the average, in real-time, but BPF and DC are much slower. From Tables 5 and 6, we can see more information. The bottleneck is the detection stage as our results shown. Our tracker runs fast because we take full advantage of the information provided by tracker such as object type, object scale and object prediction location to detect smaller region. But BPF and DC need to do a much larger scale detection.

From Table 6, we can see that for BPF and DC, runtime of tracking is approximated to our system's runtime when we use GT as the detector’s result. When we detect on the foreground, BPF and DC will be slower. BPF and DC will be slowest when we detect on the whole image. The reason for this phenomenon is that false detection increase. False detection’s increasement lead to object number's increasement, then more process is need. False detection is not a very big problem when the number of detector is one. But three detectors' false detection accumulation makes it seriously. Our method improves the runtime performance through the decrease of false detection.

6.5.2. Evaluation on benchmarks

It is shown in Table 8 how many milliseconds each component of our method needs on the public videos. The time consumed by ViBe and tracking is stable relatively. The time consumed by detection is related to the number of objects appearing in the same frame and the size of the object. We can see that in the most widely used sequences PETS-S2L1 PETS-S3-MF1, our method is very close to real time. The objects in sequence TUD-Stadtmitte are a little bigger than the size of PETS, so the runtime performance of TUD-Stadtmitte is less than that of two previous sequences. The objects of the remain sequences are too many (20–40 objects per frame) to get the good runtime performance. As we show the results of the state-of-the-art methods reported in their own literatures, we do not have their runtime performance. We can just report the runtime performance of detection in Table 9. DC, Milan's method and GMCP are tracking-by-detection method of which the first step is to detect objects on the whole image. The

### Table 3
Quantitative results on the benchmark.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Method</th>
<th>MOTA</th>
<th>MOTP</th>
<th>Prcn</th>
<th>Rcll</th>
<th>FAR</th>
<th>GT</th>
<th>MT</th>
<th>PT</th>
<th>ML</th>
<th>FP</th>
<th>FN</th>
<th>IDs</th>
<th>FM</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETS-S2L1</td>
<td>DC</td>
<td>90.6</td>
<td>80.2</td>
<td>98.4</td>
<td>92.4</td>
<td>0.07</td>
<td>23</td>
<td>21</td>
<td>1</td>
<td>1</td>
<td>59</td>
<td>302</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>KSP</td>
<td>80.3</td>
<td>72.0</td>
<td>96.3</td>
<td>83.8</td>
<td>0.16</td>
<td>23</td>
<td>17</td>
<td>4</td>
<td>2</td>
<td>126</td>
<td>641</td>
<td>13</td>
<td>22</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>90.3</td>
<td>74.3</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>18</td>
<td>–</td>
<td>0</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>GMCP</td>
<td>90.3</td>
<td>69.02</td>
<td>93.64</td>
<td>96.45</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>8</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>86.8</td>
<td>67.4</td>
<td>95.3</td>
<td>92.4</td>
<td>0.23</td>
<td>23</td>
<td>22</td>
<td>22</td>
<td>1</td>
<td>0</td>
<td>181</td>
<td>301</td>
<td>39</td>
</tr>
<tr>
<td>TUD-Stadtmitte</td>
<td>DC</td>
<td>71.1</td>
<td>65.5</td>
<td>86.7</td>
<td>84.7</td>
<td>0.51</td>
<td>9</td>
<td>7</td>
<td>2</td>
<td>0</td>
<td>92</td>
<td>108</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>KSP</td>
<td>45.8</td>
<td>56.7</td>
<td>79.2</td>
<td>63.1</td>
<td>0.65</td>
<td>9</td>
<td>7</td>
<td>1</td>
<td>7</td>
<td>117</td>
<td>261</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>56.2</td>
<td>61.6</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>4</td>
<td>–</td>
<td>–</td>
<td>0</td>
<td>–</td>
<td>–</td>
<td>15</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>GMCP</td>
<td>77.7</td>
<td>63.4</td>
<td>95.6</td>
<td>81.4</td>
<td>0.61</td>
<td>10</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>100</td>
<td>354</td>
<td>19</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>58.2</td>
<td>53.5</td>
<td>87.9</td>
<td>69.4</td>
<td>0.61</td>
<td>10</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>100</td>
<td>354</td>
<td>19</td>
<td>24</td>
</tr>
<tr>
<td>PETS-S3-MF1</td>
<td>DC</td>
<td>92.7</td>
<td>82.7</td>
<td>99.0</td>
<td>97.7</td>
<td>0.05</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>12</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>KSP</td>
<td>83.7</td>
<td>77.8</td>
<td>95.4</td>
<td>87.9</td>
<td>0.21</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>1</td>
<td>22</td>
<td>62</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>89.2</td>
<td>71.3</td>
<td>98.3</td>
<td>91.5</td>
<td>0.09</td>
<td>7</td>
<td>5</td>
<td>2</td>
<td>0</td>
<td>11</td>
<td>53</td>
<td>4</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>GMCP</td>
<td>56.9</td>
<td>59.4</td>
<td>89.8</td>
<td>65.5</td>
<td>1.43</td>
<td>74</td>
<td>7</td>
<td>28</td>
<td>34</td>
<td>12</td>
<td>622</td>
<td>2881</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>24.2</td>
<td>60.9</td>
<td>92.1</td>
<td>26.8</td>
<td>0.44</td>
<td>74</td>
<td>7</td>
<td>27</td>
<td>40</td>
<td>193</td>
<td>6117</td>
<td>22</td>
<td>38</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>46.0</td>
<td>59.8</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>25</td>
<td>–</td>
<td>8</td>
<td>–</td>
<td>–</td>
<td>126</td>
<td>105</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>28.3</td>
<td>64.8</td>
<td>79.1</td>
<td>44.3</td>
<td>2.24</td>
<td>74</td>
<td>6</td>
<td>50</td>
<td>18</td>
<td>977</td>
<td>4649</td>
<td>363</td>
<td>443</td>
</tr>
<tr>
<td>PETS-S2L2</td>
<td>DC</td>
<td>45.4</td>
<td>64.6</td>
<td>90.9</td>
<td>57.0</td>
<td>0.70</td>
<td>44</td>
<td>9</td>
<td>17</td>
<td>18</td>
<td>169</td>
<td>1572</td>
<td>38</td>
<td>27</td>
</tr>
<tr>
<td></td>
<td>KSP</td>
<td>28.8</td>
<td>61.8</td>
<td>95.7</td>
<td>30.4</td>
<td>0.19</td>
<td>44</td>
<td>5</td>
<td>8</td>
<td>31</td>
<td>45</td>
<td>2269</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>39.8</td>
<td>65.0</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>8</td>
<td>–</td>
<td>19</td>
<td>–</td>
<td>–</td>
<td>27</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>42.5</td>
<td>65.2</td>
<td>95.2</td>
<td>48.8</td>
<td>0.45</td>
<td>44</td>
<td>7</td>
<td>25</td>
<td>12</td>
<td>108</td>
<td>2242</td>
<td>106</td>
<td>127</td>
</tr>
<tr>
<td>PETS-S1L1-2</td>
<td>DC</td>
<td>57.9</td>
<td>59.7</td>
<td>91.8</td>
<td>64.5</td>
<td>0.61</td>
<td>36</td>
<td>19</td>
<td>6</td>
<td>11</td>
<td>148</td>
<td>918</td>
<td>21</td>
<td>13</td>
</tr>
<tr>
<td></td>
<td>KSP</td>
<td>51.5</td>
<td>64.8</td>
<td>93.6</td>
<td>55.5</td>
<td>0.41</td>
<td>36</td>
<td>16</td>
<td>6</td>
<td>14</td>
<td>98</td>
<td>1151</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>60.0</td>
<td>61.9</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>21</td>
<td>–</td>
<td>11</td>
<td>–</td>
<td>–</td>
<td>22</td>
<td>19</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>37.8</td>
<td>65.1</td>
<td>87.6</td>
<td>47.4</td>
<td>0.72</td>
<td>36</td>
<td>7</td>
<td>20</td>
<td>9</td>
<td>174</td>
<td>1360</td>
<td>74</td>
<td>90</td>
</tr>
<tr>
<td>PETS-S1L2-1</td>
<td>DC</td>
<td>30.8</td>
<td>49.0</td>
<td>86.4</td>
<td>38.5</td>
<td>1.13</td>
<td>43</td>
<td>7</td>
<td>16</td>
<td>20</td>
<td>227</td>
<td>2308</td>
<td>61</td>
<td>35</td>
</tr>
<tr>
<td></td>
<td>KSP</td>
<td>19.5</td>
<td>60.6</td>
<td>92.6</td>
<td>21.4</td>
<td>0.32</td>
<td>43</td>
<td>4</td>
<td>10</td>
<td>29</td>
<td>64</td>
<td>2950</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>Milan’s</td>
<td>26.5</td>
<td>60.2</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>6</td>
<td>–</td>
<td>23</td>
<td>–</td>
<td>–</td>
<td>15</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ours</td>
<td>11.1</td>
<td>49.8</td>
<td>78.8</td>
<td>20.4</td>
<td>1.42</td>
<td>43</td>
<td>3</td>
<td>11</td>
<td>29</td>
<td>206</td>
<td>2984</td>
<td>145</td>
<td>163</td>
</tr>
</tbody>
</table>

### Table 4
Detection's quantitative evaluation.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Prcn</th>
<th>Rcll</th>
</tr>
</thead>
<tbody>
<tr>
<td>PETS-S2L1</td>
<td>74.3</td>
<td>86.9</td>
</tr>
<tr>
<td>TUD-Stadtmitte</td>
<td>45.5</td>
<td>54.6</td>
</tr>
<tr>
<td>PETS-S3-MF1</td>
<td>77.0</td>
<td>85.5</td>
</tr>
<tr>
<td>PETS-S2L2</td>
<td>72.7</td>
<td>57.3</td>
</tr>
<tr>
<td>PETS-S2L3</td>
<td>61.3</td>
<td>36.3</td>
</tr>
<tr>
<td>PETS-S1L1-2</td>
<td>74.6</td>
<td>58.6</td>
</tr>
<tr>
<td>PETS-S1L2-1</td>
<td>41.4</td>
<td>19.4</td>
</tr>
</tbody>
</table>
milliseconds they cost is more than the detection. Our method is much faster than the detection and certainly, much faster than tracking-by-detection method.

The bottleneck of improving runtime performance is the detection stage. As HOG detector has real-time GPU implementations [49], our system can be further to speed up. The dream that every frame is handled in real-time will not be too far away.

7. Conclusion

Tracking and detection are separated in tracking-by-detection method. For the sake of traffic video surveillance, we combine detection and tracking, and propose a multi-object tracking called Multi-object Tracking with inter-feedback between Detection and Tracking (MTDT). We show that objects’ type, size and probable location gotten from tracking phase which is ignored in state-of-

Table 5
Run time of detection (ms per frame).

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Detection on the foreground</th>
<th>Detection on the image</th>
</tr>
</thead>
<tbody>
<tr>
<td>crossroad</td>
<td>134</td>
<td>546</td>
</tr>
<tr>
<td>straight-road1</td>
<td>64</td>
<td>529</td>
</tr>
<tr>
<td>straight-road2</td>
<td>63</td>
<td>539</td>
</tr>
<tr>
<td>straight-road3</td>
<td>106</td>
<td>559</td>
</tr>
<tr>
<td>straight-road4</td>
<td>60</td>
<td>559</td>
</tr>
<tr>
<td>straight-road5</td>
<td>90</td>
<td>554</td>
</tr>
<tr>
<td>straight-road6</td>
<td>115</td>
<td>571</td>
</tr>
<tr>
<td>straight-road7</td>
<td>106</td>
<td>531</td>
</tr>
<tr>
<td>straight-road8</td>
<td>77</td>
<td>532</td>
</tr>
<tr>
<td>straight-road9</td>
<td>95</td>
<td>565</td>
</tr>
<tr>
<td>straight-road10</td>
<td>132</td>
<td>602</td>
</tr>
<tr>
<td>straight-road11</td>
<td>65</td>
<td>539</td>
</tr>
<tr>
<td>straight-road12</td>
<td>90</td>
<td>534</td>
</tr>
</tbody>
</table>
the-art algorithm are useful in detection. Experimental results demonstrate that our method is effective and efficient.

Future work will focus on how to calculate similarity especially in multi-object situation, and flexible detector which could be modified quickly according to information feedback from tracking.

References
