

# Learning Affinities and Dependencies for Multi-Target Tracking using a CRF Model

Bo Yang, Chang Huang and Ram Nevatia

University of Southern California, Institute for Robotics and Intelligent Systems  
Los Angeles, CA 90089, USA

{yangbo|huangcha|nevatia}@usc.edu

## Abstract

We propose a learning-based Conditional Random Field (CRF) model for tracking multiple targets by progressively associating detection responses into long tracks. Tracking task is transformed into a data association problem, and most previous approaches developed heuristical parametric models or learning approaches for evaluating independent affinities between track fragments (tracklets). We argue that the independent assumption is not valid in many cases, and adopt a CRF model to consider both tracklet affinities and dependencies among them, which are represented by unary term costs and pairwise term costs respectively. Unlike previous methods, we learn the best global associations instead of the best local affinities between tracklets, and transform the task of finding the best association into an energy minimization problem. A RankBoost algorithm is proposed to select effective features for estimation of term costs in the CRF model, so that better associations have lower costs. Our approach is evaluated on challenging pedestrian data sets, and are compared with state-of-art methods. Experiments show effectiveness of our algorithm as well as improvement in tracking performance.

## 1. Introduction

Multi-target tracking is an important problem in computer vision. It aims at inferring trajectories for each target from a video sequence. This problem becomes very difficult in crowded scenes due to similar appearance, inter and intra occlusions, low resolution, etc. Figure 1 shows an example of our tracking results in a crowded scene.

With significant improvements in object detection, many association based tracking methods have been proposed to deal with crowded scenes by considering more frames and making global inference. Such approaches tend to associate detection responses or track fragments (*i.e.*, tracklets) gradually into long tracks [14][3][20][17][10]. Affinities

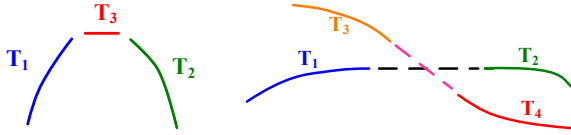


Figure 1. Sample tracking results in a crowded scene.

between tracklets, indicating linking probabilities, are often estimated by pre-defined parametric models [14][20] or an offline learning algorithm [13]; the best global association<sup>1</sup> is achieved by Hungarian algorithm [14] or cost flow method [20].

All these approaches share the assumption that the local associations are independent of each other, *i.e.*, the affinity for two tracklets does not change with associations of other tracklets; however, this assumption is not always valid in real cases. Motion smoothness is an important cue for affinity computation. In Figure 2(a), associating  $T_1$  and  $T_3$  is smooth, as only positions of a few detections in  $T_3$  need to be refined; similarly, associating  $T_2$  and  $T_3$  is also smooth. However, associating the three tracklets together would produce a sharp change in the motion direction. We call this *motion dependency*. Occlusions in the gap of two tracklets are often another important cue for tracklets affinities. In Figure 2(b), if  $T_3$  and  $T_4$  are associated, the gap between  $T_1$  and  $T_2$  is occluded; otherwise, it is not occluded. Affinities for two cases would be different; we call this *occlusion*

<sup>1</sup>We call an association between a tracklet pair *local association*, but associations for all tracklets *global association*.



(a) Motion dependency. (b) Occlusion dependency.  
Figure 2. Examples of dependencies between tracklet associations.

dependency.

We propose a framework for considering both the affinities between the tracklets and the dependencies among local associations by a Conditional Random Field (CRF) model. Each node in the CRF denotes a tracklet pair, and the label for the node indicates the association result of the pair. In a CRF model, affinities and dependencies are represented by unary and pairwise energy terms in the CRF respectively.

The framework of our approach is shown in Figure 3. In the training part, we aim at finding proper energy functions for the CRF model. For any two global associations in the training set<sup>2</sup>, the preferred association should have lower energy than the other. This is achieved by a RankBoost algorithm [6], which continually selects weak rankers from a ranker pool to form a strong ranker. Each weak ranker is based on either a unary feature for tracklet affinities or a pairwise feature for dependencies. During the test part, a CRF graph is created from the input tracklets (or detection responses), and energies of all nodes and edges are computed from the learned strong ranker. By minimizing the energy for the CRF graph, the final tracking result is produced. The contributions of this paper are:

- A learning-based CRF framework that models both the affinities between the tracklets and the dependencies among local associations.
- Novel unary features for modeling affinities and pairwise features for modeling dependencies.
- Instead of learning the best local affinities like most previous work, we learn the best global association directly by a RankBoost algorithm, so that the learning target is consistent with the problem solution, whereas better local affinities do not necessarily assure better global associations.

The rest of this paper is organized as follows: related work is discussed Section 2; problem formulation is given in Section 3; Section 4 describes learning approaches for a CRF model; experimental results are shown in Section 5, followed by conclusion in Section 6.

## 2. Related Work

Tracking methods can be classified into two categories: using only information from previous frames, or using both past and future frames. Methods belonging to the former

<sup>2</sup>Creation of the set will be described in Section 4

often adopt a particle filtering framework and integrate multiple cues to estimate most probable state or lowest energy state [12][18][5][8][4]. Methods considering both past and future frames often adopt an association-based framework, which associates tracklets gradually into longer ones by Dynamic Programming [3], MCMC [16][19], or global optimization methods [14][11][20][17].

Association based methods use different approaches to better global associations, such as estimation of merge and split [14], occlusion reasoning [20][17], multi-cue offline learning [13]. Though great progress has been achieved by association-based approaches, dependencies between local associations are ignored. Great effort has also been put on producing better local affinity scores between tracklets, *i.e.*, producing high association confidence for tracklets belonging to the same target and low confidence for those belonging to different targets, for example, dynamic feature selection [4][2] and online learning [10][7]. However, better local affinities does not necessarily assure better global association results. To the best of our knowledge, there has been no explicit use of association dependencies or direct optimization for global tracking results.

Note that [20] and [16] both create graphs for tracking, but nodes in those graphs are either detection responses or tracklets, and relationships between nodes denote affinities between tracklets, not association dependencies. However, in our approach, each node in the graph denotes an association of two tracklets, and links between nodes indicate association dependencies.

## 3. CRF Formulation for Tracklet Association

Given an input video, we first detect pedestrians in each frame. Then we associate detection responses into confident low level tracklets  $\mathbf{S} = \{T_0, T_1, \dots, T_n\}$  as input; in this association process, we make conservative associations: two responses are only associated when they are in consecutive frames and are close enough in space and similar enough in appearance. A graph  $G = (N, E)$  is created for representing affinities between tracklets in  $\mathbf{S}$  and dependencies among tracklet association pairs, and are defined as  $N = \{n_0, n_1, \dots, n_k\}$ , and  $E = \{(n_i, n_j)\}$ , where  $n_i, n_j \in N$  and  $n_i$  and  $n_j$  has motion or occlusion dependency. Each node is defined as a tracklet pair as  $n_i = (T_{i_1}, T_{i_2})$ , where  $i_1 = i_2$  or  $T_{i_1}$  is linkable to  $T_{i_2}$  which is defined as

$$0 < T_{i_2}.Start - T_{i_1}.End < t_{max} \quad (1)$$

where  $T_{i_1}.End$  indicates the end time of  $T_{i_1}$ ,  $T_{i_2}.Start$  indicates the start time of  $T_{i_2}$ , and  $t_{max}$  is a maximum allowed frame gap between any linkable tracklet pairs. A tracking result becomes a label map on  $G$  as  $L = \{l_0, l_1, \dots, l_k\}$ , where  $l_i \in \{0, 1\}$  denotes the label for node



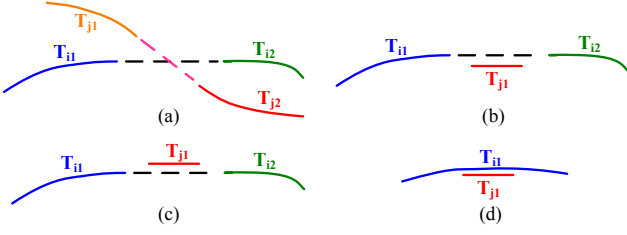


Figure 4. Four types of occlusion dependencies.

however, if  $T_{i_2}^l$  is long enough, there is no dependency, as the target has enough time to change the motion pattern.

As as shown in Figure 4, occlusion dependency has more types defined as:

- Dependency between  $n_i = (T_{i_1}, T_{i_2})$  and  $n_j = (T_{j_1}, T_{j_2})$  as shown in Figure 4(a), when the gap between  $T_{i_1}$  and  $T_{i_2}$  is occluded by the gap between  $T_{j_1}$  and  $T_{j_2}$ , and vice versa.
- Dependency between  $n_i = (T_{i_1}, T_{i_2})$  and  $n_j = (T_{j_1}, T_{j_1})$  as shown in Figure 4(b)(c), when the gap between  $T_{i_1}$  and  $T_{i_2}$  is occluded by  $T_{j_1}$ , and vice versa.
- Dependency between  $n_i = (T_{i_1}, T_{i_1})$  and  $n_j = (T_{j_1}, T_{j_1})$  as shown in Figure 4(d), when  $T_{i_1}$  is occluded by  $T_{j_1}$ , and vice versa.

The definition is based on the idea that whenever the occluded information of one node is influenced by association of another tracklet pair or judging another tracklet as a false alarm, they have an occlusion dependency.

By checking motion and occlusion dependencies between any node pair, the edge set of the graph is defined as  $E = \{(n_i, n_j)\}$ , where  $n_i, n_j \in N$  and there is a dependency between them.

## 4.2. Learning Energy Terms

In order to find the best energy function in Equ. 5, we continually add non-parametric weak energy functions to form the final function by a RankBoost algorithm. Each weak energy function is either a unary function  $u_t$  defined on the label of a node or a pairwise function  $b_t$  defined on the labels of a node pair. Therefore,  $U(l_i|n_i) = \sum_t u_t(l_i|n_i)$  and  $B(l_i, l_j|(n_i, n_j)) = \sum_t b_t(l_i, l_j|(n_i, n_j))$ .

The training set is composed of a set of ranking association pairs  $R = \{(L_{p_1}, L_{p_2})\}$ , where  $\forall p$   $F(A_{p_1}) < F(A_{p_2})$  and each  $L_{p_1}$  or  $L_{p_2}$  is a global association result, not local labels for some nodes. In order to produce an initial training set, we first generate a global association set  $T = \{L_1, L_2, \dots, L_m\}$  by randomly performing the following operations: **1)** breaking an existing link; **2)** connecting two linkable tracklets; **3)** changing status of a tracklet, *i.e.*, false alarm to true tracklet and vice versa. Then we are able to compare the correctness of any two associations in

**Input:** global association set  $T = \{L_1, L_2, \dots, L_m\}$ ,

ranking sample pair set  $R = \{(L_{p_1}, L_{p_2})\}$ .

Initialize the weight for each sample  $(L_{p_1}, L_{p_2})$ ,  $\omega_i = \frac{1}{|R|}$ .

Initialize unary and pairwise energy functions:  $U(\cdot) = B(\cdot) = 0$

**For**  $t = 1, \dots, n$  **do**:

- Find best weak energy function  $h(L)$  and its weight  $\alpha$ .
- Update sample weight as  $\omega_i = \omega_i * \exp(\alpha(h(L_{p_1}) - h(L_{p_2})))$ .
- Normalize weight so that  $\sum_i \omega_i = 1$ .
- If  $h$  is a unary function,  $U = U + \alpha h$ ; otherwise,  $B = B + \alpha h$ .
- Find best association result  $L^*$  under current energy function  $E$ .  $\forall L_p \in T$ ,  $R = R \cup (L^*, L_p)$  if  $F(L^*) < F(L_p)$ ; otherwise,  $R = R \cup (L_p, L^*)$ .

**Output:** CRF energy function  $E(L) = U(L) + B(L)$ .

Table 1. Learning algorithm for CRF energy function.

$T$  by the evaluation function  $F$ , and form the initial training set. For any global association result  $L$ , a weak function  $u_t$  or  $b_t$  would output the cost for all labels in  $L$  as

$$u_t(L) = \sum_{n_i \in N} u_t(l_i|n_i)$$

$$b_t(L) = \sum_{(n_i, n_j) \in E} b_t(l_i, l_j|(n_i, n_j)) \quad (8)$$

The best energy function is learned by a RankBoost algorithm. We aim at finding a function to satisfy as many ranking pairs as possible. Therefore, the loss function for boosting is defined as

$$Z = \sum_i \omega_i^0 \exp(E(L_{i_1}) - E(L_{i_2})) \quad (9)$$

where  $\omega_i^0$  is the initial weight for each training sample, and the weight is updated during boosting process. In the  $t$ -th round, we aim at finding a best weak energy function  $h_t(A)$  and a weight factor  $\alpha$  which minimizes

$$Z_t = \sum_i \omega_i^t \exp(\alpha(h_t(L_{i_1}) - h_t(L_{i_2}))) \quad (10)$$

where  $h_t$  is either a unary or a pairwise energy function defined in Equ. 8. In the training process, we update the training association set  $T$  by continuously adding new samples, which is the association result generated by minimizing current energy function. Then we could have more training pairs, and these pairs are probably more difficult samples, just like samples from a bootstrap process. The learning algorithm is shown in Table 1.

## 4.3. Learning the Weak Energy Function

A weak energy function  $h$  is either a unary function or a pairwise function, which takes a label or a label pair as

**Input:** ranking sample pair set  $R = \{(L_{p_1}, L_{p_2})\}$  and current weight  $\omega$  for each sample.

Initialize training loss  $Z^* = +\infty$

**For each feature  $f$  do:**

**For each threshold  $\eta$  do:**

- $h(x) = \begin{cases} 1 & \text{if } f(x) > \eta \\ -1 & \text{otherwise} \end{cases}$ .
- Compute loss function  $Z$  on current sample weight distribution as  $Z(\alpha) = \sum_i \omega_i^t \exp(\alpha(h(L_{i_1}) - h(L_{i_2})))$
- Find  $\tilde{\alpha} = \operatorname{argmin}_{\alpha} Z(\alpha)$
- If  $Z(\tilde{\alpha}) < Z^*$ , let  $Z^* = Z(\tilde{\alpha})$ ,  $h^* = h$ ,  $\alpha^* = \tilde{\alpha}$

**Output:** weak energy function  $h^*(x)$  and  $\alpha^*$ .

Table 2. Learning algorithm for a weak energy function.

input and outputs a real value for energy cost. By treating different labels as different functions, *e.g.*,  $u_t$  is divided into  $u_t^0$  and  $u_t^1$  which estimate energy costs for label 0 and 1 respectively, we take a node or a node pair as the input instead of the labels. In the following, we use  $h$  and  $x$  to represent a weak energy function and its input respectively;  $x$  is a node indicating a tracklet pair, when  $h$  is a unary function;  $x$  is a node pair, when  $h$  is a pairwise function. The input  $x$  is represented by a series of features which include as most cues as possible to model tracklet affinities and association dependencies. Each feature is a function that maps a node or a node pair into a real value. A weak energy function  $h$  is defined based on one single feature  $f$  as

$$h(x) = \begin{cases} 1 & \text{if } f(x) > \eta \\ -1 & \text{otherwise} \end{cases} \quad (11)$$

where  $\eta$  is a threshold. The output of  $h$  on an association  $L$  is defined as  $h(L) = \sum_{x \in L} h(x)$ . The optimum  $\alpha$  for  $h$  could be found by Newton's method. The learning algorithm of the weak energy function is given in Table 2.

The feature pool used in our learning process is defined in Table 3 and Table 4. More features can be added into the feature pool if necessary; the purpose is to include any possible cues for modeling affinities and dependencies. In [13], a feature pool for learning local associations is proposed; that pool can be viewed as a subset of the unary feature pool, as it not only models affinities between tracklet pairs but also models costs of one tracklet being false alarm or true tracklet. In addition, pairwise feature pool does not exist in [13] owing to their independence assumption on local associations.

#### 4.4. Implementation Details

In this section, we will introduce the definition of evaluation function  $F$ , energy minimization method, and some

<sup>3</sup>an occlusion is assumed when two detection responses have an overlap ratio larger than 0.5.

	<b>Id</b>	<b>Feature description</b>
Length	1	Length of $T_i$ or $T_j$
	2	Number of detection responses in $T_i$ or $T_j$
	3	Number of detection responses in $T_i$ or $T_j$ divided by length of $T_i$ or $T_j$
Appearance	4	Similarity of color histogram between $T_i$ 's tail part and $T_j$ 's head part
	5	Self appearance smoothness of $T_i$
	6	Appearance smoothness in the gap of $T_i$ and $T_j$
Gap	7	Frame gap between $T_i$ 's tail and $T_j$ 's head
	8	Number of miss detected frames in $T_i$ or in the gap between $T_i$ and $T_j$
	9	Number of frames occluded <sup>3</sup> by other tracklets in $T_i$ or in the gap between $T_i$ and $T_j$
	10	Number of miss detected frames divided by the tracklet length or the gap length
	11	Number of frames occluded divided by the tracklet length or the gap length
End	12	Estimated time from $T_i$ 's head to the nearest entry point
	13	Estimated time from $T_j$ 's tail to the nearest exit point
Motion	14	Motion smoothness in image plane or ground plane if connecting $T_i$ and $T_j$
	15	Self motion smoothness of $T_i$ in image plane or ground plane

Table 3. Feature pool for learning unary terms.

	<b>Id</b>	<b>Feature description</b>
Motion	1	Motion smoothness in image plane or ground plane if connecting $T_i$ , $T_j$ , and $T_k$ together sequentially.
	2	Motion smoothness divided by the sum of detection responses in $T_i$ , $T_j$ , and $T_k$ .
Occlusion	3	Number of frames in the gap of $T_{i_1}$ and $T_{i_2}$ occluded by the gap of $T_{j_1}$ and $T_{j_2}$ .
	4	Number of frames in the gap of $T_{i_1}$ and $T_{i_2}$ occluded by $T_j$ .
	5	Number of frames in $T_i$ occluded by the gap of $T_{j_1}$ and $T_{j_2}$ .
	6	Number of frames in $T_i$ occluded by $T_j$ .

Table 4. Feature pool for learning pairwise terms.

other details.

**Evaluation Function.** As there is no commonly used single score for evaluating tracking results, we adopt multiple scores together, including: recall rate and precision, number of id switches, number of fragments. An association  $L_p$  is better than  $L_q$ , when the former has better scores in all four evaluations than the latter.

**Energy Minimization.** With an energy function and a CRF association graph, we need to find a label map with minimum energy as the final association result. However the energy function is not sub-modular. For exam-

ple, the constraint in Equ. 2 is achieved by defining an infinite cost for assigning two nodes in  $\chi_{i_1}^{head}$  or  $\chi_{i_2}^{tail}$ , i.e.,  $B(l_i = 1, l_j = 1) = +\infty$  if  $n_i, n_j \in \chi_{i_1}^{head}$  or  $n_i, n_j \in \chi_{i_2}^{tail}$ ; yet a sub-modular energy function should satisfy  $B(0, 0) + B(1, 1) < B(0, 1) + B(1, 0)$ . Therefore, we cannot find a globally optimal solution easily. We first use Hungarian algorithm [14] to find an initial optimum solution using only the unary energy costs. Then a simulated annealing algorithm is used to iteratively find a solution with lower energies; in early iterations, the global association result can change easily to avoid sticking to a local minimum, and it gradually goes to a stable state with a low energy cost.

**Speed Optimization.** As each node in a CRF graph denotes two linkable tracklets, whereas edges are defined between potential nodes, the numbers of nodes and edges could be  $O(n^2)$  and  $O(n^4)$  respectively. To limit size of the graph, we adopt a sliding window approach and build a separate CRF graph for each window; there are overlaps between neighboring windows so that trajectories can grow across different windows. Similar to [13], we make a multi-level association to progressively link tracklets so that we can limit the maximum frame gap and maximum speed between linkable tracklets at early levels, and loosen the constraint at later levels. In early levels, there are more tracklets and the constraint significantly reduce the graph size; in later levels, the number of tracklets is much less, and the constraint is loosened to avoid missing possible connections.

Note that all components introduced in this section are modular and can be easily replaced by others. For example, a better energy minimization algorithm or a better evaluation function would help improve the performance of the whole system.

## 5. Experiments

We applied the CRF based tracking approach to human tracking, and evaluated the performance on the public Trecvid08 data set [15], which captures several indoor scenes of a busy airport and contains crowds of persons and frequent occlusions. This data set is much more complex than the commonly used CAVIAR data set [1]; as the the state-of-art approaches produce almost perfect performance on the latter, we do not use it in our experiments.

### 5.1. Analysis of the Training Process

In the training process, the first five features selected are  $P_2, U_7, P_5, U_8, U_1$ , where  $U_i$  or  $P_i$  denote the  $i$ -th feature in the unary feature pool (Table 3) or in the pairwise feature pool (Table 4). We see that motion dependency and occlusion dependency are as important as motion smoothness or number of frames in the gap. Note that all features in [13]

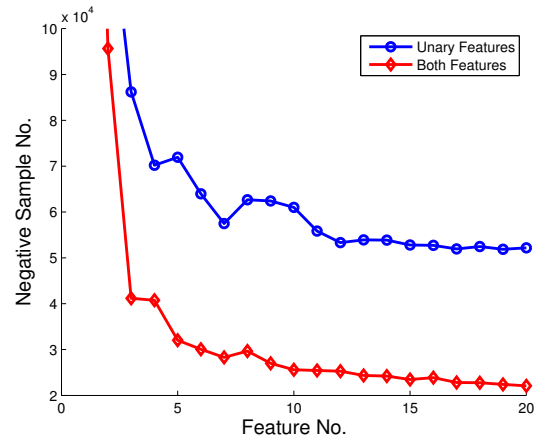


Figure 5. Comparison of negative sample numbers under different feature numbers. The blue curve shows results using only unary features; the red one shows results using both unary and pairwise features.

are used for modeling linkable tracklet pairs; however, the  $U_1$  selected here is for modeling false alarm tracks.  $P_2, P_5$ , and  $U_1$  are all features not contained in the feature pool in [13]. This indicates the effectiveness of the new features as well as the importance of modeling dependencies.

Figure 5 shows the trend of negative sample numbers in the first 20 rounds of training. We see that using both unary and pairwise features is more discriminative than using only unary features. This indicates that association dependencies do exist frequently in real cases, and the proposed pairwise features are effective for modeling these dependencies.

### 5.2. Evaluation Metrics

As it is difficult to produce a single score for tracking result evaluation, we use multiple scores like most previous work, including

- **GT:** The number of trajectories in ground truth.
- **Recall:** correctly matched detections / total detections in ground truth.
- **Precision:** correctly matched detections / total detections in the tracking result.
- **FAF:** Average false alarms per frame.
- **MT:** The ratio of mostly tracked trajectories, which are successfully tracked for more than 80%.
- **ML:** The ratio of mostly lost trajectories, which are successfully tracked for less than 20%.
- **PT:** The ratio of partially tracked trajectories, i.e.,  $1 - MT - ML$ .
- **Frag:** fragments, the number of times that a ground truth trajectory is interrupted.
- **IDS:** id switch, the number of times that a tracked trajectory changes its matched id.

Method	Recall	Precision	GT	MT	PT	ML	Frag	IDS
Huang <i>et al.</i> [9]	71.6%	80.8%	919	57.0%	28.1%	14.9%	487	278
Li <i>et al.</i> [13]	80.0%	83.5%	919	77.5%	17.6%	4.9%	310	288
CRF Unary Only	80.1%	84.3%	919	78.0%	17.0%	5.0%	309	278
CRF Tracking	79.2%	85.8%	919	78.2%	16.9%	4.9%	319	253

Table 5. Comparison of results on TRECVID 2008 dataset.

For fragments and id switch, we adopt the definition in [13], which is more strict but more clearly defined than previous definitions.

### 5.3. Tracking Performance

We evaluated our approach on the Trecvid08 data set, which includes three different scenes with more than 10 persons per frame on average. People are frequently occluded or interact with each other, making the tracking problem very challenging. We select six video clips for each scene with a length of 5,000 frames each. We use a total of nine videos selected evenly from each scene as the training set, so that it has enough variety; the other nine videos are used for testing. As only a few have reported performance on this challenging data set, we compare our results with [9] and [13]. Quantitative results are shown in Table 5.

“CRF unary only” row in Table 5 shows the performance of our approach without using pairwise features, while the last row shows the results using all listed features. We see that even using unary features only, our approach achieves better performance than of [13] and [9]. By using both unary and pairwise features, our approach achieves the best precision and mostly tracked rate; the recall rate is comparable to [13]. With 9 more fragments, we reduce 35 id switches compared with [13]. This demonstrates that the CRF model is able to consider one step further than using only affinities which may be heavily affected by dependencies with other local associations. Note that in [10], a lower id switch is reported; however, [10] focuses on producing a more discriminative appearance model, while we aim at building a more powerful framework for integrating multiple cues, and the model in [10] can be integrated into our system as a more powerful feature. Our framework does not have limitations on models for affinities and dependencies, and can easily absorb any new models.

Figure 6 shows some sample tracking results. In the first row, three persons (No. 19, No. 25, No. 26) come very close to each other; our tracker is able to differentiate them with each other and keep their identities. In frame 447 and 469, there is a similar case; all three involved persons are tracked correctly and successfully. Figure 6(b) shows an occlusion case, where person 12 and 13 are occluded by person 32 and 35. We see that the occlusion lasts for more than fifty frames, and sometimes these persons are fully overlapped. However, our tracker is able to provide cor-

rect associations by considering association affinities and dependencies. In Figure 6(c), a man is bending in front of the door for more than 200 frames, and multiple persons pass through the door and have overlaps with this man. The tracklet for the man does not drift to other persons and all persons are tracked correctly.

The training for the CRF energy function takes about 12 hours. Testing time is tightly related with the number of tracklets. As the nodes in the CRF graph builds on tracklet pairs and edges builds on node pairs, the time cost is polynomial to number of tracklets, which is often proportional to the number of persons. We implement our system using C on a PC with 3.0G Hz CPU and 8GB memory. The average testing time is around ten minutes for a video clip with 5000 frames and around 10 persons per frame.

## 6. Conclusion

We propose a CRF model to transform the problem of multi-target tracking into an energy minimization problem. The proposed RankBoost algorithm is able to learn cost functions for modeling both affinities between tracklets and local association dependencies. Experimental results have demonstrate effectiveness of our approach in crowded scenes as well as the importance of local association dependencies. Comparisons with up-to-date methods are given. Future work would be focused on integrating online information into the framework.

## Acknowledgments

This paper is based upon work supported in part by Office of Naval Research under grant number N00014-10-1-0517.

## References

- [1] Caviar dataset. <http://homepages.inf.ed.ac.uk/rbf/CAVIAR/>.
- [2] S. Avidan. Ensemble tracking. In *CVPR*, 2005.
- [3] J. Berclaz, F. Fleuret, and P. Fua. Robust people tracking with global trajectory optimization. In *CVPR*, 2006.
- [4] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool. Robust tracking-by-detection using a detector confidence particle filter. In *ICCV*, 2009.
- [5] K. J. Cannons, J. M. Gryn, and R. P. Wildes. Visual tracking using a pixelwise spatiotemporal oriented energy representation. In *ECCV*, 2010.

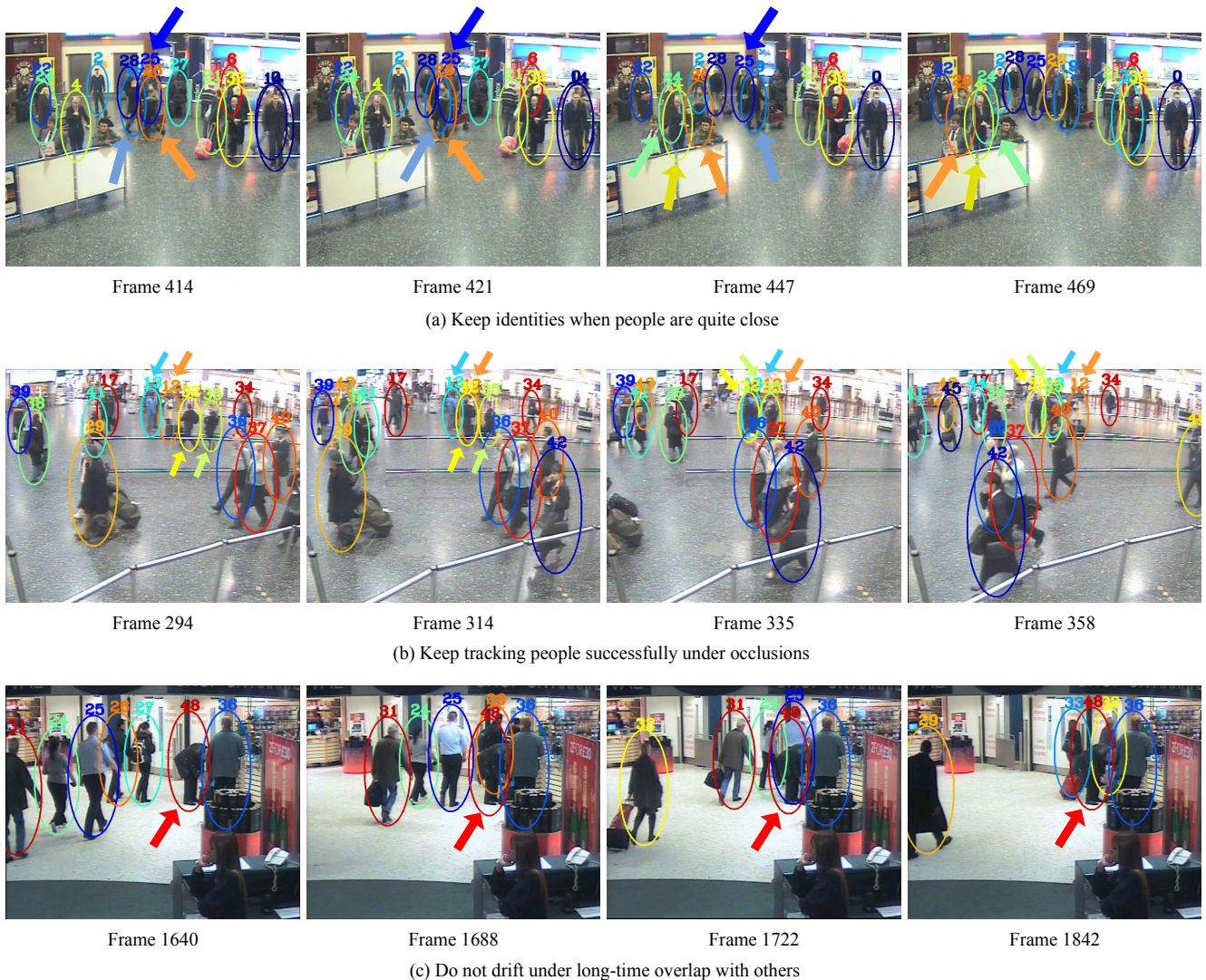


Figure 6. Sample tracking results of our approach on different scenes in Trecvid08 data set [15].

- [6] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *The Journal of Machine Learning Research*, 4:933–969, 2003.
- [7] H. Grabner and H. Bischof. On-line boosting and vision. In *CVPR*, 2006.
- [8] H. Grabner, J. Matas, L. V. Gool, and P. Cattin. Tracking the invisible: Learning where the object might be. In *CVPR*, 2010.
- [9] C. Huang, B. Wu, and R. Nevatia. Robust object tracking by hierarchical association of detection responses. In *ECCV*, 2008.
- [10] C.-H. Kuo, C. Huang, and R. Nevatia. Multi-target tracking by on-line learned discriminative appearance models. In *CVPR*, 2010.
- [11] B. Leibe, K. Schindler, and L. V. Gool. Coupled detection and trajectory estimation for multi-object tracking. In *ICCV*, 2007.
- [12] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different lifespans. In *CVPR*, 2007.
- [13] Y. Li, C. Huang, and R. Nevatia. Learning to associate: Hybrid-boosted multi-target tracker for crowded scene. In *CVPR*, 2009.
- [14] A. G. A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu. Multi-object tracking through simultaneous long occlusions and split-merge conditions. In *CVPR*, 2006.
- [15] A. F. Smeaton, P. Over, and W. Kraaij. Evaluation campaigns and trecvid. In *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, 2006.
- [16] B. Song, T.-Y. Jeng, E. Staudt, and A. K. Roy-Chowdhury. A stochastic graph evolution framework for robust multi-target tracking. In *ECCV*, 2010.
- [17] J. Xing, H. Ai, and S. Lao. Multi-object tracking through occlusions by local tracklets filtering and global tracklets association with detection responses. In *CVPR*, 2009.
- [18] M. Yang, F. Lv, W. Xu, and Y. Gong. Detection driven adaptive multi-cue integration for multiple human tracking. In *ICCV*, 2009.
- [19] Q. Yu, G. Medioni, and I. Cohen. Multiple-target tracking by spatiotemporal monte carlo markov chain data association. In *CVPR*, 2007.
- [20] L. Zhang, Y. Li, and R. Nevatia. Global data association for multi-object tracking using network flows. In *CVPR*, 2008.