

# CDT: Cooperative Detection and Tracking for Tracing Multiple Objects in Video Sequences

Han-Ul Kim and Chang-Su Kim

School of Electrical Engineering, Korea University  
hanulkim@mcl.korea.ac.kr, changsukim@korea.ac.kr

**Abstract.** A cooperative detection and model-free tracking algorithm, referred to as CDT, for multiple object tracking is proposed in this work. The proposed CDT algorithm has three components: object detector, forward tracker, and backward tracker. First, the object detector detects targets with high confidence levels only to reduce spurious detection and achieve a high precision rate. Then, each detected target is traced by the forward tracker and then by the backward tracker to restore undetected states. In the tracking processes, the object detector cooperates with the trackers to handle appearing or disappearing targets and to refine inaccurate state estimates. With this detection guidance, the model-free tracking can trace multiple objects reliably and accurately. Experimental results show that the proposed CDT algorithm provides excellent performance on a recent benchmark. Furthermore, an online version of the proposed algorithm also excels in the benchmark.

**Keywords:** Joint detection and tracking, multiple object tracking, object detection, model-free tracking, and online multi-object tracking.

## 1 Introduction

The objective of multiple object tracking (MOT) is to estimate the states (or bounding boxes) of as many objects as possible in a video sequence and trace them temporally. Especially, tracking specific objects, such as pedestrians and cars, has drawn attention for its various applications, including surveillance systems and self-driving cars. For this purpose, many tracking-by-detection algorithms [1–19] have been proposed to yield promising performance. The tracking-by-detection approach decomposes MOT into two subproblems: object detection and data association. It first detects objects in each frame and then links the detection results to form trajectories across frames. With the recent success of object detection techniques [20–23], this approach has several advantages over model-free tracking, which does not assume a specific object and instead traces the bounding box of an arbitrary object, manually annotated in the first frame. Specifically, the tracking-by-detection approach is more robust against object appearance variation and model drift, and it can identify emerging or disappearing objects in a video sequence more easily.

MOT, however, still remains a challenging problem in case of crowded or cluttered scenes. A complicated scene causes more detection failures, which are

either undetected objects (false negatives) or spurious detection (false positives). The poor detection, in turn, decreases the accuracy of data association. To compensate for detection failures, many MOT algorithms [1–12, 14] focus on the global data association. Given detection results in all frames, they design a cost function to formulate the data association as an optimization problem and then determine optimal trajectories by minimizing the cost function. By considering detection results in all frames simultaneously, they can alleviate adverse effects of detection failures. Notice that an alternative approach for achieving accurate MOT is to improve the quality of object detection directly. But, contrary to the data association that has been investigated intensively, relatively little efforts have been made for this straightforward approach in the MOT community.

In this work, we attempt to improve the detection quality, by combining an object detector with a model-free tracker. We first collect detection results with high confidence levels only to decrease the number of false positives. However, there is a trade-off between precision and recall, and reducing false positives increases undetected objects. To restore the undetected objects, we conduct model-free tracking in the forward and backward directions sequentially. In general, a model-free tracker and an object detector have different strengths and weaknesses. For instance, a model-free tracker can temporally trace missing states of a target object from its initial state, but it is vulnerable to model drift and may fail to identify the appearance or disappearance of a target reliably, which can be easily handled by an object detector. Therefore, we propose the cooperative detection and tracking (CDT) algorithm, in which an object detector and a model-free tracker cooperate to complement each other. Specifically, the detector initiates the tracker, by providing initial states of targets, and informs of the termination conditions for the tracking. Also, the detector is utilized to refine the tracking results. Experimental results demonstrate that the proposed CDT algorithm improves the quality of object detection and excels on a recent MOT benchmark [24]. Moreover, by omitting the backward tracking, the proposed algorithm can operate online.

The rest of this paper is organized as follows: Section 2 reviews related work. Section 3 presents the proposed CDT algorithm. Section 4 analyzes the performance of the proposed algorithm. Finally, Section 5 draws conclusions.

## 2 Related Work

### 2.1 Multiple Object Tracking

Many MOT algorithms, including [1–12], adopt global (or batch) data association techniques. Specifically, a batch of frames are taken as input, objects are detected in these frames, and the association among the detected results is formulated as the minimization of a cost function. Then, the optimal trajectories are determined to minimize the cost function. Some algorithms formulate the data association as a relatively simple problem, such as linear programming relaxation [1, 7] and minimum-cost flow [3–5, 11], for which the global minimum can be computed plausibly. Other algorithms consider more complicated

problems to represent real world scenarios more faithfully, which are however too complex to find the global optimum. Thus, they instead find locally optimal solutions, by employing quadratic boolean programming [2], continuous or discrete-continuous energy minimization [9, 12], generalized clique graph [8, 10], and maximum weight-independent set [6].

Recently, Bae and Yoon [13] proposed the notion of tracklet confidence to handle fragmented trajectories and adopted online learning to discriminate target appearance during the data association. To reduce the dependency on erroneous detection results, Leal-Taixé *et al.* [14] proposed interaction feature strings, which encode pedestrians’ interactions. Milan *et al.* [15] developed a unified framework of tracking and segmentation to exploit low-level image features more effectively. Xiang *et al.* [16] formulated the online MOT problem as decision making in a Markov decision process. Rezatofighi *et al.* [17] reformulated the joint probabilistic data association (JPDA) [25] technique to make it computationally tractable. Similarly, Kim *et al.* [18] revisited another classic solution, multiple hypothesis tracker (MHT) [26], and adopted an online discriminative appearance model using a deep convolutional neural network. Choi [19] introduced the aggregated local flow descriptor to encode the relative motion pattern between two objects and proposed a near-online MOT algorithm. Also, Wang *et al.* [27] proposed the target-specific metric learning to represent target appearance faithfully. Moreover, their algorithm utilizes a motion cue for accurate tracking and shows excellent results in the MOT benchmark [24].

## 2.2 Object Detection

Recently, deep convolutional neural networks have made impressive progress in object detection. Girshick *et al.* [20] proposed the R-CNN detector using a deep convolutional neural network to classify object proposals. To prevent the overfit due to a small dataset, they introduced a domain-specific fine-tuning method, improving the detection performance dramatically. To accelerate the processing speed of R-CNN, He *et al.* [21] presented the architecture to compute a convolutional feature map of an entire image prior to the spatial pyramid pooling. However, it cannot fine-tune the convolutional layers. Girshick [22] proposed the Fast R-CNN, which is an end-to-end trainable system with shared convolutional layers. In these detectors [20–22], separate region proposal methods [28–30] are required, which cause computational bottlenecks. To overcome this issue, Shaoqing [23] introduced the region proposal network to extract proposals from a convolutional feature map directly.

## 2.3 Model Free Tracking

For model-free tracking, we adopt a discriminative approach that uses a classifier to estimate targets states. Thus, we briefly review discriminative trackers only. Avidan [31] used an offline-trained classifier to track a target, but the classifier may provide wrong results when an object changes its appearance. In [32, 33], online learning techniques have been developed to adjust to variations in object

appearance. These techniques, however, may update appearance models unreliably with falsely labeled samples. Grabner *et al.* [34] attempted to reduce the impacts of false labels, by training a classifier with labeled samples in the first frame and unlabeled samples in subsequent frames. Babenko *et al.* [35] adopted the multiple instance learning to deal with the ambiguity in the foreground labeling. Hare *et al.* [36] employed the structured support vector machine [37] to avoid a heuristic for assigning binary labels to samples. Henriques *et al.* [38] introduced a correlation filter to track an object and performed the filtering efficiently in the Fourier domain.

### 3 Proposed Algorithm

The proposed CDT algorithm consists of three components: object detector, forward tracker, and backward tracker. The first component detects targets in a video using a conventional detector [22]. It selects only the detection results with high scores to provide reliable information to the other components. The second component traces the detection results forwardly in the time domain to restore undetected states of the targets. To this end, we adopt a model-free tracker that is guided by the object detector. The third component conducts the backward tracking to recover more missing states and refine the target trajectories.

#### 3.1 Object Detection

We adopt an end-to-end trainable object detector, called Fast R-CNN [22]. However, other detectors, *e.g.* [20, 21, 23], also can be used instead of Fast R-CNN.

**Training:** We employ the pre-trained Fast R-CNN detector with the VGG16 model [39], trained with the PASCAL VOC dataset. Since the MOT challenge dataset [24] is for pedestrian detection, we replace the softmax layer to consider only two classes (pedestrian or non-pedestrian). We use the selective search [29] to generate region proposals. To fine-tune the detector on the MOT challenge dataset, we adopt the mini-batch sampling in [22].

**Detection:** Given a video, we generate region proposals using [29]. Then, the detector measures the score for each proposal and chooses only the proposals whose scores are greater than a high threshold  $\theta_{\text{high}} = 0.99$ . A lot of objects remain undetected due to the high threshold  $\theta_{\text{high}}$ . However, the impacts of undetected objects are less severe than those of false positives, since our CDT system includes a model-free tracker to trace temporally the states of a target object from its initial state. When a target is detected, we regard it as an initial state. Then, we can restore undetected target states by performing the model-free tracking. To summarize, since precision is more important than recall, we adopt the high threshold  $\theta_{\text{high}}$  to provide reliable information to the tracker.

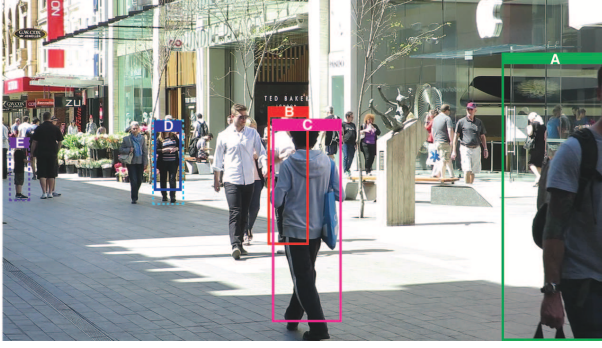


Fig. 1. Illustration of the forward tracking process.

### 3.2 Forward Tracking

As mentioned above, many target states remain undetected. To restore these missing states, we carry out model-free tracking forwardly in the time domain. During the tracking, the object detector cooperates with the model-free tracker to achieve accurate tracking. Fig. 1 illustrates the forward tracking. Let  $\mathcal{A}_t$  be the active target list to record detected target states and their appearance models in frame  $t$ . Given active targets in  $\mathcal{A}_{t-1}$  in the previous frame  $t-1$ , the forward tracker estimates their states in the current frame  $t$ . Then, guided by the object detector, the tracker checks the visibility of each active target to remove disappearing or occluded targets. The detector also helps to improve the initial state estimation of the tracker. To this end, we perform the matching between tracked target states and detection results, and update a target state when its corresponding detection result is more reliable. Also, an unmatched detection result is regarded as a new target and added to the active target list  $\mathcal{A}_t$ . Finally, the appearance model of each active target is updated.

**State Estimation:** Let  $\mathbf{x}_{i,t-1} = (\mathbf{c}, w, h) \in \mathcal{A}_{t-1}$  denote the state of the  $i$ th active target in frame  $t-1$ , where  $\mathbf{c}$ ,  $w$ , and  $h$  are the location, width, and height of the target. Given the previous state  $\mathbf{x}_{i,t-1}$ , we estimate its current state  $\mathbf{x}_{i,t}$  using a discriminative tracker and put it into the active target list  $\mathcal{A}_t$ . We set a square search region with center  $\mathbf{c}$  and side length  $\sqrt{wh}$ . Then, we sample candidate states within the search region using the sliding window method. To describe the contents in each candidate  $\mathbf{x}$ , we encode it into a feature vector  $\phi(\mathbf{x})$ . Similarly to [40], the candidate window is decomposed into 64 non-overlapping patches and each patch is described by a 24-dimensional RGB histogram and a 31-dimensional HOG histogram [41]. We then concatenate all patch features to



**Fig. 2.** Examples of the detection guidance. Target ‘A’ disappears from the view. Target ‘B’ is occluded by target ‘C.’ The initial state of target ‘D’ is refined into the cyan box. Target ‘E’ newly appears. All these disappearance, occlusion, refinement, and appearance cases are handled using the guidance information from the object detector.

construct  $\phi(\mathbf{x})$ . We determine the current state  $\mathbf{x}_{i,t}$  to yield the highest score,

$$\mathbf{x}_{i,t} = \arg \max_{\mathbf{x}} \mathbf{w}_i^T \phi(\mathbf{x}) \quad (1)$$

where  $\mathbf{w}_i$  is the appearance model of the  $i$ th target.

**Detection Guidance:** During the tracking, the tracker is guided by the object detector. Let us consider target ‘A’ in Fig. 2, which is disappearing from the view. Note that the goal of object detection is to identify the existence of an object. Thus, we can easily handle the disappearing case by computing the detection score. Specifically, we check the detection score for each active target in  $\mathcal{A}_t$  and remove targets, whose scores are lower than a threshold  $\theta_{\text{low}}$ , from  $\mathcal{A}_t$ . The low threshold  $\theta_{\text{low}}$  is fixed to 0.5, since we consider only two classes and a lower score than 0.5 indicates that the estimated state contains no pedestrian.

In Fig. 2, targets ‘B’ and ‘C’ illustrate an occlusion case, which also causes an invisible target. To find occlusion, we compute the intersection-over-union (IoU) overlap ratio between each pair of active targets. We declare that an occlusion case occurs, when the overlap ratio is larger than a threshold  $\theta_{\text{iou}} = 0.3$ . Then, we compare the detection scores of the two targets and determine that the target with a lower score is occluded. We exclude the occluded target from  $\mathcal{A}_t$ .

Another task of the detection guidance is to refine the initial estimation of the tracker for each active target. An object may experience scale variation. In this case, the tracker may provide an inaccurate result, since it does not consider scale variation in this work. To correct such inaccuracy, we utilize the detection results in Section 3.1. More specifically, we first determine which detection result should be used for the target state refinement. The matching cost  $c(\mathbf{x}, \mathbf{z})$  between

a target state  $\mathbf{x}$  and a detection result  $\mathbf{z}$  is defined as

$$c(\mathbf{x}, \mathbf{z}) = \begin{cases} \|\phi(\mathbf{x}) - \phi(\mathbf{z})\|^2 & \text{if } \Delta(\mathbf{x}, \mathbf{z}) \geq \theta_{\text{iou}}, \\ \infty & \text{otherwise,} \end{cases} \quad (2)$$

where  $\Delta(\mathbf{x}, \mathbf{z})$  denotes the IoU overlap ratio. We determine the optimal matching between the initial tracking results and the detection results using the Hungarian algorithm [42]. For instance, target ‘D’ becomes closer to the camera in Fig. 2, in which the initial estimation of the tracker and the corresponding detection result are depicted by blue and cyan boxes, respectively. In this example, the detection result represents the target more faithfully. In general, given an initial target state and the matching detection result, we compare their detection scores. If the detection result yields a higher score, then it replaces the initial state. Finally, when a detection result is unmatched, *e.g.* target ‘E’ in Fig. 2, we regard it as a new target and insert it into the active target list  $\mathcal{A}_t$ .

**Appearance Model Update:** After determining the state of each target in  $\mathcal{A}_t$ , we update its appearance model. We model the target appearance using the structured support vector machine (SSVM) [36,37], which yields excellent performance in a recent model-free tracking benchmark [43]. To update the appearance model of the  $i$ th target, we sample 81 bounding boxes around the current object location and extract their feature vectors. SSVM constrains that the bounding box  $\mathbf{x}_{i,t}$  should yield a larger score than a nearby box  $\mathbf{x}$  by a margin, which decreases as the IoU overlap ratio between the two boxes increases. Specifically, the appearance model  $\mathbf{w}_i$  is trained to minimize an objective function,

$$\mathbf{w}_i = \arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|^2 + \beta \sum_{\mathbf{x}} \max\{0, (1 - \Delta(\mathbf{x}_{i,t}, \mathbf{x})) - \mathbf{w}^T(\phi(\mathbf{x}_{i,t}) - \phi(\mathbf{x}))\} \quad (3)$$

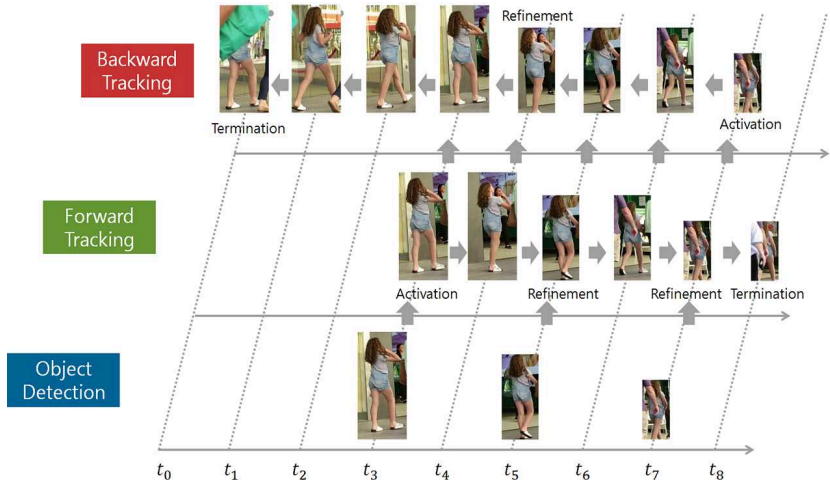
where  $\beta$  is 10. We use the LaRank algorithm [36,44] for the minimization.

### 3.3 Backward Tracking

Fig. 3 illustrates the entire process of the proposed CDT algorithm. First, the object detector collects reliable detection results. Once an object is detected, it becomes an active target and the forward tracker estimates its missing states after the activation. Moreover, the detector cooperates with the tracker to improve the estimation. For example, in Fig. 3, the estimated states are replaced by the detection results at  $t_5$  and  $t_7$  to localize the target more accurately. The forward tracking stops when the target disappears due to occlusion or out-of-view. After the forward tracking, the states in  $t_1 \sim t_2$  still remain undetected and should be estimated. Therefore, we further perform the backward tracking.

Suppose that the forward tracking starts at  $t_s$  and ends at  $t_{e+1}$ . Then, the backward tracking is activated at  $t_e$ . First, it refines the trajectory, obtained by the forward tracking, backwardly from  $t_e$  to  $t_s$ . Second, the backward tracking restores missing states backwardly from  $t_s$  until its termination.

**Existing State Refinement:** Given the target state  $\mathbf{x}_{i,t+1}$  in the subsequent frame  $t+1$ , the backward tracker traces its state  $\mathbf{x}_{i,t}$  in the current frame  $t$ .



**Fig. 3.** Illustration of the target tracking. A pedestrian is first detected at  $t_3$  and becomes an active target. From  $t_4$ , the forward tracker estimates its states. At  $t_5$  and  $t_7$ , the tracker refines the estimated states using the detection results. The forward tracking terminates at  $t_8$  due to the occlusion. Given the state at  $t_7$ , the backward tracker estimates new states at  $t_6$  and  $t_5$  to refine the trajectory. However, these states are rejected since the original states yield higher detection scores. On the contrary, a new state is accepted at  $t_4$ . Moreover, the backward tracking restores undetected states at  $t_2$  and  $t_1$ , and then stops at  $t_0$ .

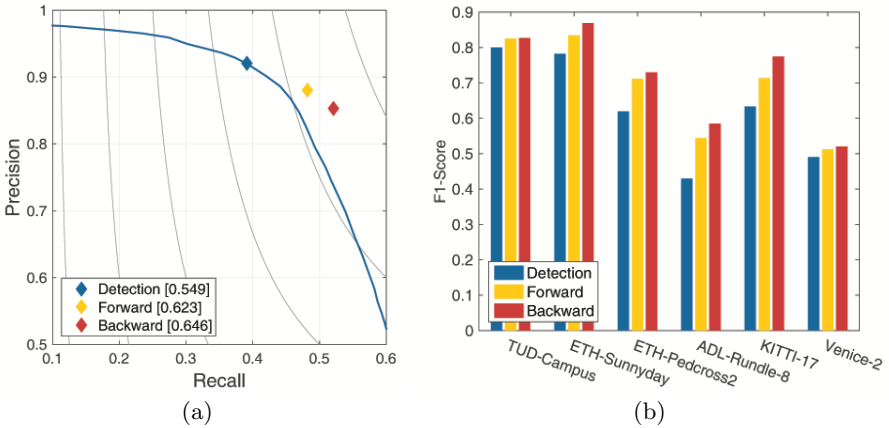
It compares the detection score of  $\mathbf{x}_{i,t}$  with that of the forward tracking result. Then, the state with a higher score is selected as the final tracking result. In Fig. 3, the forward result is replaced by the backward result at  $t_4$ .

**Missing State Restoration:** The backward tracking process from  $t_s$  is identical with the forward tracking process in Section 3.2, but the temporal direction is reversed. Moreover, we check the IoU overlap ratio of the estimated state with the other active states, which are already estimated by the forward tracker, to prevent duplicated estimation. Specifically, we only accept an estimated state  $\mathbf{x}_{i,t}$  and update its appearance model  $\mathbf{w}_i$ , when the maximum IoU overlap ratio between  $\mathbf{x}_{i,t}$  and the other active states is smaller than  $\theta_{\text{iou}}$ . Otherwise, we regard  $\mathbf{x}_{i,t}$  as duplication and terminate its backward tracking.

## 4 Experimental Results

We assess the proposed CDT algorithm on the MOT challenge benchmark [24], which consists of 22 sequences with different view points, camera motions, and weather conditions. They are divided into 11 training sequences and 11 test sequences, and annotations are available for the training sequences only.

We use the F1-score to quantify the detection performance of the proposed algorithm, which is a harmonic mean between precision and recall. We also em-



**Fig. 4.** (a) The precision-recall curve of the object detection. The blue diamond point is selected with the high threshold  $\theta_{\text{high}} = 0.99$  by the object detector. Then, the detection performance is improved by the forward tracker and then by the backward tracker. A number within brackets is the F1-score after each component. (b) The F1-scores on each validation sequence after each component.

**Table 1.** Training and validation sequences for performance analysis.

<b>Training</b>	TUD-Stadtmitte, ETH-Bahnhof, ADL-Rundle-6, PETS09-S2L1, KITTI-13
<b>Validation</b>	TUD-Campus, ETH-Sunnyday, ETH-Pedcross2, ADL-Rundle-8, KITTI-17, Venice-2

play various evaluation metrics in the benchmark [24] to compare the proposed and conventional algorithms: the number of false positives (FP), the number of false negatives (FN), the number of ID switches (IDS), multiple object tracking accuracy (MOTA), multiple object tracking precision (MOTP), mostly tracked targets (MT), and mostly lost targets (ML).

#### 4.1 Analysis on Validation Sequences

Unlike the training sequences, the annotations for the test sequences are not released. Therefore, as done in [16], we partition the 11 training sequences into two subsets ‘training’ and ‘validation’ to analyze the proposed algorithm. Table 1 lists the contents in each subset. We use the five ‘training’ sequences to fine-tune the object detector and the six ‘validation’ ones for the evaluation.

Fig. 4(a) is the precision-recall curve of the object detector. In the proposed MOT system, the high detection threshold  $\theta_{\text{high}} = 0.99$  is used to detect reliable objects only. Therefore, the operating point is selected from the high-precision, low-recall part of the curve, resulting in the F1-score of 0.549. In the forward tracking, the model-free tracker detects missing objects and increases the recall rate, at the cost of a relatively small reduction in the precision rate. Thus, after the forward tracking, the F1-score is increased to 0.623. Similarly, the

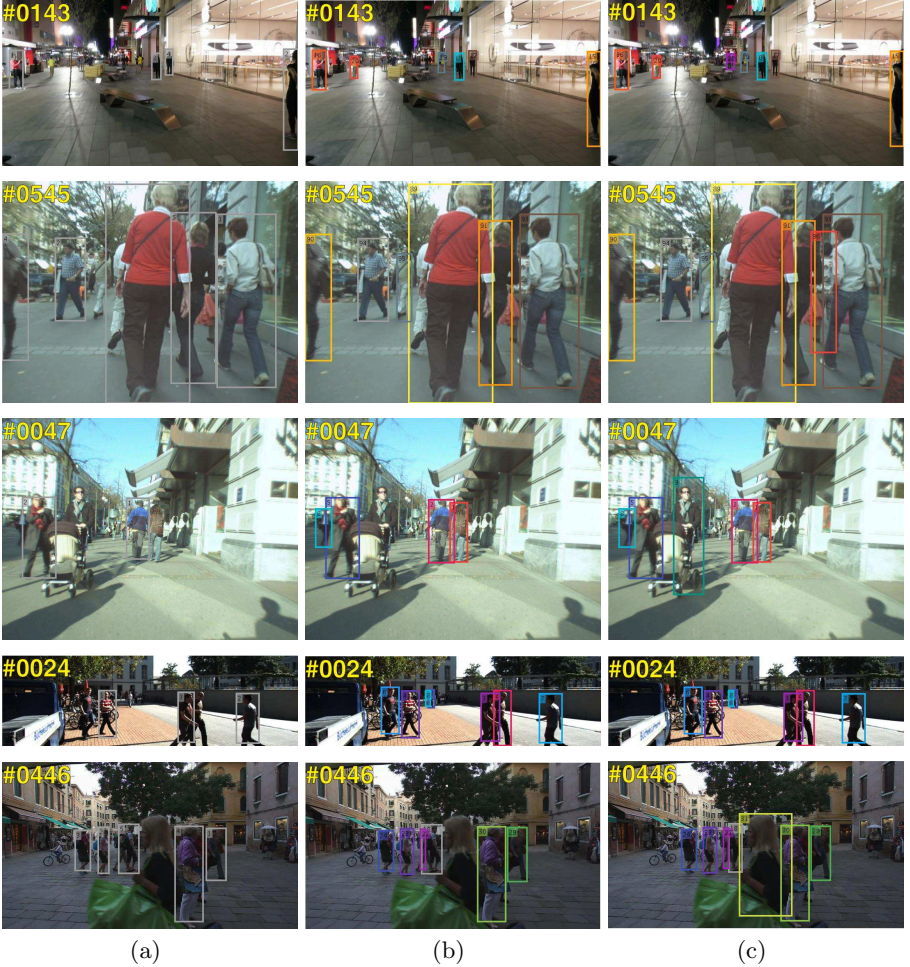
**Table 2.** MOT results of the proposed CDT algorithm on the validation sequences with various settings. Settings ‘A’ and ‘B’ denote the results after the forward tracking, and settings ‘C’ and ‘D’ after the backward tracking. Settings ‘A’ and ‘C’ do not perform the state refinement, while settings ‘B’ and ‘D’ do.

Setting	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓
A: Forward w/o	26.8	67.1	12.8	28.2	3,160	13,457	293
B: Forward	40.8	72.5	22.2	27.8	1,516	11,964	180
C: Backward w/o	42.1	72.1	32.1	26.5	2,119	11,072	184
D: Backward	42.4	72.2	32.5	26.5	2,077	11,050	178

backward tracking further improves the detection performance and the final F1-score becomes 0.646, which corresponds to 17.7% improvement in comparison with the original F1-score. Fig. 4(b) presents the F1-scores on each sequence after the object detection, the forward tracking, and the backward tracking. Each component of the proposed algorithm leads to better results on all sequences with no exception. As mentioned in Section 3.2, the tracker is guided by the detector in our CDT system. It is worth pointing out that the cooperation in the opposite direction is also carried out; the experimental results in Fig. 4 indicate that the detection performance itself is improved with the assistance of the tracker.

Next, to quantify the performance gain achieved by each algorithmic part, Table 2 compares the MOT results using various settings. Settings ‘A’ and ‘B’ denote the results after the forward tracking: ‘A’ does not perform the state refinement using the detection guidance, while ‘B’ does. Setting ‘C’ and ‘D’ denote the results after the backward tracking: ‘C’ does not conduct the state refinement, while ‘D’ does. Comparing ‘A’ with ‘B,’ we see that the state refinement significantly increases the MOTA score by 14 points in the forward tracking. Notice that a refined state also helps to estimate the target states in subsequent frames. In other words, correcting a tracking error not only improves the state in the current frame but also prevents model-drift. Thus, it contributes to the extraction of a longer trajectory of a target, thereby increasing the MT score from 12.8 to 22.2. The comparison between ‘C’ and ‘D’ shows that the state refinement also improves the performance of the backward tracking. It improves the MOTA, MOTP, MT scores and reduces false positives, false negatives, ID switches. The comparison between ‘B’ and ‘C’ demonstrates the impacts of the state estimation in the backward tracking. It restores a lot of missing target states and improves the performance.

Figs. 5(a), (b), and (c) compare active targets after the object detection, the forward tracking, and the backward tracking, respectively. The number of states, provided by the object detector, is relatively small, but it is increased by the forward tracker and then by the backward tracker. For instance, the object with ID 19 in “ADL-Rundle-8” is missing in the detection stage. However, it is already active in the previous frame and its state is accurately estimated by the forward tracker. Similarly, the object with ID 22 in “ADL-Rundle-8” is restored by the backward tracker. The results on the other sequences also confirm that the forward and backward trackers restore missing states effectively.



**Fig. 5.** Comparison of active targets after (a) the object detection, (b) the forward tracking, and (c) the backward tracking. From top to bottom, the sequences “ADL-Rundle-8,” “ETH-Pedcross2,” “ETH-Sunnyday,” “KITTI-17,” and “Venice-2” are used.

## 4.2 Additional Tests on Validation Sequences

Table 3 shows the results of the proposed algorithm on the validation sequences with different object proposal methods. SS denotes the default mode, in which the detector uses the selective search [29] to generate proposals. On the other hand, ACF indicates that it employs the published detection results in the benchmark [24] as proposals. It can be observed that, even when we use the published detection results, the proposed algorithm provides similar performance. Compared with SS, ACF [45] provides more reliable results for small scale objects

**Table 3.** The results of the proposed algorithm on the validation sequences with different object proposals. SS denotes that the selective search [29] is used for generating object proposals, while ACF [45] means that the published detection results in the MOT benchmark [24] are used as the proposals.

Sequences	Proposal	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓
TUD-Campus	SS	67.1	73.8	62.5	0.0	27	87	4
	ACF	71.0	74.2	62.5	0.0	13	87	4
ETH-Sunnyday	SS	72.1	74.4	70.0	3.3	297	202	20
	ACF	66.0	77.6	40.0	23.3	129	494	9
ETH-Pedcross2	SS	53.6	74.3	25.5	32.3	402	2,435	69
	ACF	28.4	74.4	10.5	63.9	162	4,302	23
ADL-Rundle-8	SS	34.9	69.7	32.1	25.0	683	3,695	41
	ACF	38.2	74.0	28.5	28.5	449	3,714	30
KITTI-17	SS	58.3	70.8	11.1	0.0	64	211	10
	ACF	60.8	71.5	11.1	0.0	45	217	6
Venice-2	SS	29.2	70.8	23.1	42.3	604	4,420	34
	ACF	41.5	75.1	34.6	42.3	116	4,042	23
Average	SS	42.4	72.2	32.5	26.5	2,077	11,050	178
	ACF	39.9	74.8	20.9	47.4	914	12,856	95

**Table 4.** Performance comparison with the same detector.

Algorithm	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓
MHT [18]	39.9	74.7	18.8	38.9	616	13,180	70
JPDA [17]	27.2	72.2	14.1	42.3	2,970	13,625	220
Proposed	42.4	72.2	32.5	26.5	2,077	11,050	178

**Table 5.** Performance improvements when we use the MHT [18] algorithm for the post-processing.

Algorithm	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓
Proposed	42.4	72.2	32.5	26.5	2,077	11,050	178
Proposed+MHT	44.6	73.2	36.3	27.0	2,045	10,639	112

and improves the performance on the “Venice-2” sequence. In contrast, SS is more effective on the “ETH-Pedcross2” sequence, where ACF fails to detect objects that are very high and thus touch the top and bottom boundaries of images.

The performance of an MOT algorithm depends strongly on its object detector. Therefore, Table 4 compares the proposed algorithm with the conventional algorithms in [17, 18], by employing an identical detector. In this test, all algorithms employ the detector in Section 3.1. We obtain the results of the conventional algorithms using their published codes and default parameters. Overall, the proposed algorithm provides the best performance. JPDA [17] yields relatively poor performance, since it does not consider the appearance of targets. Thus, it cannot handle complex scenes, in which it is difficult to identify objects from the motion information only. The recent state-of-the-art MHT [18] uses the convolutional neural network (CNN) features to exploit the appearance information and outperforms JPDA. However, it suffers from missing states in the detection stage and produces a lot of false negatives. In contrast, the proposed algorithm restores undetected states effectively, reducing the false negatives.

As mentioned previously, the tracking-by-detection approach decomposes the MOT problem into two subproblems: object detection and data association. The proposed algorithm focuses on improving the detection quality, but its performance can be also improved by adopting a sophisticated data association technique. Table 5 shows the improvements on the validation sequences when we use MHT [18] for the data association. In this test, using the results of the proposed algorithm as initial detection results, we employ MHT as a post-processing scheme for the data association. Notice that the post-processing improves the MOTA score by 2.2 points, by reducing the number of tracking failures (FP, FN, and IDS). From a different point of view, we see in Tables 4 and 5 that the proposed algorithm increases the MOTA score of the MHT algorithm [18] by 4.7 points, by providing more reliable detection results.

### 4.3 Comparative Evaluation on Test Sequences

For the evaluation on test sequences, we fine-tuned the object detector using all 11 training and validation sequences in Table 1, and submitted our results to the MOT challenge website [24]. Note that the post-processing in Section 4.2 was not employed. Table 6 lists the results of the proposed CDT algorithm on each test sequence.

Table 7 compares the proposed algorithm with conventional trackers. In terms of MOTA, the proposed algorithm outperforms the recent trackers in [16–19] significantly, while providing a comparable score to the state-of-the-art tracker TSML-CDE [27]. Furthermore, the proposed algorithm provides the best performance in terms of MT, ML, and FN scores. Note that a lot of target states remain undetected in the detection stage, since the object detector focus on decreasing the number of false positives. Nevertheless, the proposed algorithm successfully restores these missing states and reduces the FN score considerably. Also, the proposed algorithm yields the outstanding performance in the trajectory quality metrics (MT,ML). These results indicate the effectiveness of the detection guidance in the model-free tracking. Even though the proposed CDT algorithm does not perform explicit global data association, the cooperation between the object detector and the model-free tracker enables to build long reliable trajectories.

By removing the backward tracker, the proposed algorithm can operate as an online tracker, since the object detector and the forward tracker are causal components. Thus, we consider it as the online version of the proposed algorithm. Table 7 presents the performance of the online version as well. The online version still outperforms the recent trackers [16–19] by considerable margins.

## 5 Conclusions

We proposed a novel MOT algorithm, called CDT. The proposed CDT algorithm first collects detection results with high confidence levels only to reduce spurious detection. Then, the proposed algorithm conducts model-free tracking in the forward direction and then in the backward direction to restore undetected states.

**Table 6.** MOT results on each test sequence.

Sequences	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓
TUD-Crossing	70.0	73.3	61.5	0.0	73	229	29
PETS09-S2L2	47.7	70.4	21.4	19.0	502	4,429	113
ETH-Jelmoli	48.2	75.0	60.0	11.1	758	529	26
ETH-Linthescher	60.5	76.4	43.7	24.4	1,425	1,963	138
ETH-Crossing	59.0	81.5	30.8	30.8	169	221	21
AVG-TownCentre	45.5	68.5	28.3	27.4	653	3,127	117
ADL-Rundle-1	26.0	70.1	21.9	28.1	1,697	5,146	47
ADL-Rundle-3	43.3	75.4	29.5	13.6	1,168	4,517	84
KITTI-16	44.7	69.3	23.5	0.0	232	690	19
KITTI-19	45.7	72.8	30.6	9.7	884	1,946	72
Venice-1	32.3	70.9	29.4	41.2	527	2,538	18
Average	44.5	72.9	34.7	22.1	8,088	25,335	684

**Table 7.** Comparison of the proposed CDT algorithm with conventional algorithms on the test sequences. The best and the second best scores are boldfaced and underlined, respectively.

Algorithm	MOTA↑	MOTP↑	MT↑	ML↓	FP↓	FN↓	IDS↓
TSML-CDE [27]	<b>49.1</b>	<b>74.3</b>	30.4	26.4	<b>5,204</b>	25,460	637
NOMT [19]	33.7	71.9	12.2	44.0	7,762	32,547	442
MHT [18]	32.4	71.8	16.0	43.8	9,064	32,060	<u>435</u>
MDP [16]	30.3	71.3	13.0	38.4	9,717	32,422	680
JPDA [17]	23.8	68.2	5.0	58.1	6,373	40,084	<b>365</b>
MotiCon [14]	23.1	70.9	4.7	52.0	10,404	35,844	1,018
SegTrack [15]	22.5	71.7	5.8	63.9	7,890	39,020	697
DCO [12]	19.6	71.4	5.1	54.9	10,652	38,232	521
CEM [9]	19.3	70.7	8.5	46.5	14,180	34,591	813
RMOT [46]	18.6	69.6	5.3	53.3	12,473	36,835	684
SMOT [47]	18.2	71.2	2.8	54.8	8,780	40,310	1,148
TC-ODAL [13]	15.1	70.5	3.2	55.8	12,970	38,538	637
DP-NMS [5]	14.5	70.8	6.0	40.8	13,171	34,814	4,537
Proposed	<u>44.5</u>	72.9	<b>34.7</b>	<b>22.1</b>	8,088	<b>25,335</b>	684
Proposed-online	42.8	<u>73.3</u>	23.8	<u>25.1</u>	5,494	28,997	668

For accurate tracking, the model-free trackers are guided by the object detector, in order to identify emerging or disappearing objects in a video and refine inaccurate state estimates. Experimental results demonstrated that the proposed CDT algorithm provides excellent performance on the MOT benchmark [24].

## Acknowledgement

This work was supported partly by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2015R1A2A1 A10055037), and partly by the MSIP, Korea, under the ITRC support program supervised by the Institute for Information & communications Technology Promotion (No. IITP-2016-R2720-16-0007).

## References

1. Jiang, H., Fels, S., Little, J.J.: A linear programming approach for multiple object tracking. In: CVPR. (2007)
2. Leibe, B., Schindler, K., Gool, L.V.: Coupled detection and trajectory estimation for multi-object tracking. In: ICCV. (2007)
3. Zhang, L., Li, Y., Nevatia, R.: Global data association for multi-object tracking using network flows. In: CVPR. (2008)
4. Berclaz, J., Fleuret, F., Turetken, E., Fua, P.: Multiple object tracking using k-shortest paths optimization. TPAMI **33**(9) (Sep. 2011) 1806–1819
5. Pirsivash, H., Ramanan, D., Fowlkes, C.C.: Globally-optimal greedy algorithms for tracking a variable number of objects. In: CVPR. (2011)
6. Brendel, W., Amer, M., Todorovic, S.: Multiobject tracking as maximum weight independent set. In: CVPR. (2011)
7. Wu, Z., Thangali, A., Sclaroff, S., Betke, M.: Coupling detection and data association for multiple object tracking. In: CVPR. (2012)
8. Roshan Zamir, A., Dehghan, A., Shah, M.: GMCP-Tracker: Global multi-object tracking using generalized minimum clique graphs. In: ECCV. (2012)
9. Milan, A., Roth, S., Schindler, K.: Continuous energy minimization for multitarget tracking. TPAMI **36**(1) (Jan. 2014) 58–72
10. Dehghan, A., Assari, S.M., Shah, M.: GMMCP tracker: Globally optimal generalized maximum multi clique problem for multiple object tracking. In: CVPR. (2015)
11. Lenz, P., Geiger, A., Urtasun, R.: Followme: Efficient online min-cost flow tracking with bounded memory and computation. In: ICCV. (2015)
12. Milan, A., Schindler, K., Roth, S.: Multi-target tracking by discrete-continuous energy minimization. TPAMI (2016)
13. Bae, S.H., Yoon, K.J.: Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning. In: CVPR. (2014)
14. Leal-Taixé, L., Fenzi, M., Kuznetsova, A., Rosenhahn, B., Savarese, S.: Learning an image-based motion context for multiple people tracking. In: CVPR. (2014)
15. Milan, A., Leal-Taixé, L., Schindler, K., Reid, I.: Joint tracking and segmentation of multiple targets. In: CVPR. (2015)
16. Xiang, Y., Alahi, A., Savarese, S.: Learning to track: Online multi-object tracking by decision making. In: ICCV. (2015)
17. Rezaatofghi, S.H., Milan, A., Zhang, Z., Shi, Q., Dick, A., Reid, I.: Joint probabilistic data association revisited. In: ICCV. (2015)
18. Kim, C., Li, F., Ciptadi, A., Rehg, J.M.: Multiple hypothesis tracking revisited. In: ICCV. (2015)
19. Choi, W.: Near-online multi-target tracking with aggregated local flow descriptor. In: ICCV. (2015)
20. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014)
21. He, K.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ECCV. (2014)
22. Girshick, R.: Fast R-CNN. In: ICCV. (2015)
23. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS. (2015)
24. Leal-Taixé, L., Milan, A., Reid, I., Roth, S., Schindler, K.: MOTChallenge 2015: Towards a benchmark for multi-target tracking. <https://motchallenge.net> Accessed: 2016-03-08.

25. Fortmann, T., Bar-Shalom, Y., Scheffe, M.: Sonar tracking of multiple targets using joint probabilistic data association. *IEEE J. Ocean. Eng.* **8**(3) (Jul. 1983) 173–184
26. Reid, D.: An algorithm for tracking multiple targets. *TAC* **24**(6) (Dec. 1979) 843–854
27. Wang, B., Wang, G., Chan, K.L., Wang, L.: Tracklet association by online target-specific metric learning and coherent dynamics estimation. *TPAMI* **PP**(99) (2016) 1–1
28. Zitnick, C., Dollár, P.: Edge boxes: Locating object proposals from edges. In: *ECCV*. (2014)
29. Uijlings, J., van de Sande, K., Gevers, T., Smeulders, A.: Selective search for object recognition. *IJCV* **104**(2) (2013) 154–171
30. Arbelaez, P., Pont-Tuset, J., Barron, J., Marques, F., Malik, J.: Multiscale combinatorial grouping. In: *CVPR*. (2014)
31. Avidan, S.: Support vector tracking. *TPAMI* **26**(8) (Aug. 2004) 1064–1072
32. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: *BMVC*. (2006)
33. Avidan, S.: Ensemble tracking. *TPAMI* **29**(2) (Feb. 2007) 261–271
34. Grabner, H., Grabner, M., Bischof, H.: Semi-supervised on-line boosting for robust tracking. In: *ECCV*. (2008)
35. Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. *TPAMI* **33**(8) (Aug. 2011) 1619–1632
36. Hare, S., Saffari, A., Torr, P.H.S.: Struck: Structured output tracking with kernels. In: *ICCV*. (2011)
37. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large margin methods for structured and interdependent output variables. *JMLR* **6** (Sep. 2005) 1453–1484
38. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. *TPAMI* **37**(3) (Mar. 2015) 583–596
39. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: *ICLR*. (2015)
40. Kim, H.U., Lee, D.Y., Sim, J.Y., Kim, C.S.: SOWP: spatially ordered and weighted patch descriptor for visual tracking. In: *ICCV*. (2015)
41. Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. *TPAMI* **32**(9) (Sep. 2010) 1627–1645
42. Kuhn, H.W.: The hungarian method for the assignment problem. *Nav. Res. Logist. Quart.* **2**(1-2) (Mar. 1955) 83–97
43. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. *TPAMI* **37**(9) (Sep. 2015) 1834–1848
44. Bordes, A., Bottou, L., Gallinari, P., Weston, J.: Solving multiclass support vector machines with larank. In: *ICML*. (2007)
45. Dollar, P., Appel, R., Belongie, S., Perona, P.: Fast feature pyramids for object detection. *TPAMI* **36**(8) (Aug. 2014) 1532–1545
46. Yoon, J.H., Yang, M.H., Lim, J., Yoon, K.J.: Bayesian multi-object tracking using motion context from multiple objects. In: *WACV*. (2015)
47. Dicle, C., Camps, O.I., Szaier, M.: The way they move: Tracking multiple targets with similar appearance. In: *ICCV*. (2013)