Boosting the Speed of Real-Time Multi-Object Trackers

Xudong Zhang
Dept of Computer Science, The Graduate Center of CUNY
New York, NY USA
xzhang5@gradcenter.cuny.edu

Liang Zhao
Dept of Computer Science of Lehman College
New York, NY USA
Liang.Zhao1@lehman.cuny.edu

Abstract—**Object tracking is an important computer vision task that has drawn increasing attention because of its academic and commercial potential. Although several trackers based on convolutional neural network (CNN) have shown strong performance in single object tracking and multiple object tracking, it remains challenging to perform real-time object tracking on videos with frame rate over 30 frame per second. For real-world applications, one of the necessary requirements is the trackers need to estimate the bounding boxes with a rate higher or equal to the video frame rate to avoid latency. In this study, we propose a hybrid tracker combining the high accuracy of CNN-based trackers and the superior processing speed of particle filtering algorithm to achieve real-time tracking. The experimental results indicate our tracker achieves a state-of-the-art performance on the benchmark multi-object tracking datasets with competitive accuracy.**

Index Terms—**Object Tracking, Particle Filters, Real-Time Tracking**

I. Introduction

Real-time object tracking has been a popular topic in computer vision and it has received lots of attention over the past decade. The application of object tracking can be found in many areas, including visual surveillance [1], aircraft tracking [2], multiple pedestrian tracking [3], unmanned aerial vehicles (UAV) [4], augmented reality [5], human-computer interactions [6], intelligent driving [7], and pedestrian behavior analysis [8]. Among the performance metrics of tracking models, the tracking speed is one of the most critical factors that determines the whether a tracking model has realistic applications. In real-world applications, the trackers need to estimate the bounding boxes for each video/image frame at a frequency higher than or equal to the video frame rate to achieve real-time tracking. For example, a tracker with low tracking speed cannot be adequately used on a video with a high frame rate due to the processing latency. Videos nowadays have framerates of at least 24 frames per second (FPS), which is much higher than the tracking speeds of most of the existing tracking methods [9]–[12].

According to the number of targets, the tracking tasks can be classified into single object tracking tasks and multiple object tracking tasks. In single object tracking (SOT) problems, the trackers are focusing on estimating the coordinates of one object in the video/image frames. In multiple object tracking (MOT) tasks, the goal is to estimate the locations and scales of multiple objects in a video/image sequence. Both SOT trackers and MOT trackers need to handle the challenging scenes with illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, and low-resolution [13]. The MOT tasks are considered to be one of the most challenging tasks in computer vision [14], [15]. This is because the MOT trackers not only detect multiple objects in complicated environments but also identify the same objects, new objects, and vanished objects between any two consecutive frames in a video/image sequence, which refers to object re-identification (ReID) [16]. Due to these reasons, the existing MOT trackers, especially the convolutional neural network (CNN) based MOT trackers, usually have much lower tracking efficiency than SOT trackers. Therefore, improving the tracking speeds of MOT trackers should receive extensive attention.

In this study, we aim to improve the tracking speed of slow-but-accurate tracking models to meet the demands of real-time tracking. Based on the object re-identification scheme, the MOT trackers can be classified into two categories, including the two-stage scheme and the one-stage scheme. In the two-stage scheme, the MOT trackers detect the objects in the first stage and then associate the detected objects in the second stage. The first stage can be handled by existing models, such as YOLO [17], DPM [18], Fast R-CNN [19], etc., which usually use several bounding boxes to swipe the whole image to identify the potential objects. In the second stage, the trackers associate the identified objects with the existent objects. These two stages cannot be completed simultaneously and thus the two-stage scheme suffers from high computational cost. Another scheme is the one-stage re-identification, in which the detection and re-identification are completed simultaneously in one stage [20], [21]. Usually, the one-stage trackers handle video/image sequence faster than the two-stage trackers, due to the simultaneous reasoning structure. However, the one-stage trackers rarely have tracking speeds greater than 30 HZ to achieve real-time tracking. According to the released results from the MOT16 and MOT17 datasets [22], the top ten trackers have maximal tracking speed of 22.7 Hz (MOT16) and 17 Hz (MOT17), respectively. Therefore, it is crucial to develop a method to drastically accelerate the efficiency of these models without compromising other aspects of their performance.

In order to boost the tracking efficiency, we develop a fast particle filtering algorithm to improve the tracking speed of costly MOT tracking algorithms. Particle filtering is an efficient approach in identifying object movements in a sequence of frames [23], [24]. Particle filters, also called sequential Monte Carlo (SMC) methods, use Bayesian inference and stochastic sampling methods to estimate the posteriors of the states [25]. In a particle filters process, the particles are used to represent the posterior distribution of a stochastic process. Particle filters can provide a numerical approximation to the non-linear and/or non-Gaussian filtering problem without the assumptions of the properties of the system model [26], [27]. Due to those advantages, particle filters have been widely used in a variety of fields, such as positioning [28], navigating [25], visual tracking [29], modeling and simulation [30].

In this study, we propose a hybrid tracker by executing a hybrid strategy that combines a CNN-based MOT model and a particle filtering procedure, so that the tracking speed can be accelerated without damaging its performance. The particle filtering algorithm does not require complex matrix operations and thus they consume much less time than the CNN-based trackers. Also, the CNN-based tracking can provide high tracking accuracy. Our empirical results confirm that the proposed tracker is capable of boosting the tracking speed of several state-of-the-art tracking model by as much as 141.8% on MOT16 dataset and 132.3% on MOT17, while keeping competitive tracking accuracy compared with the original CNN-based tracker.

The rest of the paper is organized as follows. Section II introduces the related work in object tracking. Section III presents the proposed tracking model. Section IV describes the experiments and achieved results. Section V provides the conclusions and future work.

II. Related work

Various MOT trackers have been successfully applied to multiple object tracking tasks without considering the tracking efficiency. Shyamgopal et al. proposed a tracker, which first generated tracking labels and trained a re-identification network to predict the generated labels by using cross-entropy loss. The experiments indicated that the proposed tracker outperformed the state-of-the-art performance on public datasets with a tracking speed of 1.9 Hz [9]. Zhang et al. proposed a graph-based method for near-online tracking tasks. A detection multiplexing method was introduced for tracking in monocular images and a multiplex labeling graph (MLG) model was proposed, in which each node in MLG can represent multiple targets. The experimental results indicated that the proposed near-online tracker achieved satisfactory efficiency without additional private detection on public tracking challenge with a tracking speed of 5.9 Hz [10]. Liang et al. studied the reasoning process of the object detection and identification-based tracking strategy and they revealed that the competition of those two tasks inevitably impeded the tracking performance. The authors proposed a novel cross-correlation network that effectively improved the learning of task-dependent representations. The proposed tracking model achieved new state-of-the-art performances on MOT16 and MOT17 with a tracking speed of 15.8 Hz [31]. Although high tracking speed has been achieved, it still needs to be improved to satisfy realistic applications.

As a tracking technique with low computational cost, particle filters are widely used in object tracking and detection tasks due to the low computation cost and competitive accuracy. Okuma et al. applied particle filters and machine learning algorithms to detect and track moving objects. A varying number of objects can be tracked from a large number of overlapped objects [32]. Yan et al. reconstructed the traffic jams from the deployed cameras by using particle filters. The proposed method can detect the slow-moving vehicles in the road networks, which caused the traffic jams [33]. Hwang and Sung used parallel particle filters with a centralized resampling technique to improve the estimation performance in tracking problems. The results achieved the constant execution speed, which is better than the conventional network based algorithm [34]. In order to furtherly improve the performance of particle filters, some studies focusing on the resampling [35] and parallel scheme [36] had been carried out, which significantly shortened the computation time and improved the speedup factors. In this work, we take the advantages of both CNN-based trackers and particle filters to boost the tracking speed without losing tracking accuracy.

III. The Hybrid Model

In this section, we present the CNN-based model, the particle filtering algorithm, and the proposed model combining those two models used in multiple object tracking.

A. The Particle Filtering Algorithm

In particle filtering algorithms, sequential particle filters are one of the widely used filtering algorithms. The sequential particle filtering algorithm usually runs on a single processing unit. The mechanism of sequential particle filters is the sequential importance sampling and resampling (SISR), as shown in Algorithm 1. The SISR has two basic stages, sampling and resampling. Initially, the number of particles is set to be K and these particles, $x_0^{(k)}$ are spread in the space by following the initial distribution $X_0^{(k)}$. Each particle has an equivalent weight $1/K$. In the sampling stage, at time step t, a set of particles, $\overline{x}_t^{(k)}$, is sampled from $x_{t-1}^{(k)}$. The set of particles, $\overline{x}_t^{(k)}$, is used to represent the states of interest. The weights of this set of particles representing the posteriors of the state are calculated based on the observation(s). The weights are normalized between 0 and 1 for the following resampling stage. In the resampling stage, offspring particles are obtained according to the normalized weights by following the probability $Pr(\widetilde{X}_t^{(k)} = \overline{x}_t^{(v)}) = \overline{w}_t^{(v)} / \sum_{k=1}^{K} \overline{w}_t$. Usually, the particles with higher weights will be copied more times and those with lower weights will be copied fewer times or discarded. The resampling procedures improve the robustness of the particle filters by updating the particles. At each time step, sampling and resampling are executed and the resampled particles will be the input of the sampling of the next time step. The estimates are the expectation of the all particles after resampling stages as $\pi_t^N(x_t) = \sum_{k=1}^K w_t^{(k)} \delta(x_t = x_t^{(k)})$. This procedure continues until the observations or the time steps are unavailable.

For the application of particle filters on object tracking, the input videos are handled frame by frame and the estimates are the coordinate of the object in each frame. The observations are the objects labeled in the first frames, which are cropped images with edges as the ground-truth bounding boxes. Each particle stores coordinates and this particle can be represented by a cropped image with the center equal to stored coordinates and size equal to the observations. In the sampling stages, the calculated weight of each particle is proportional to the similarity between the observation and the cropped image of this particle. The similarity is calculated by using the similarity function, which is critical in weight calculation and the following particle selection. In the experimental section, we evaluate the performance of particle filters with different similarity functions. In the resampling stages, the particles are copied and discarded according to their weights. The final estimates are obtained by calculating the expectation of all particles after the resampling stages.

B. The Multiple Object Tracking Model

The MOT tracker is a cross-correlation network based on joint detection and embedding (JDE) structure [37], which improves the collaborative learning capability between detection and re-identification tasks in one-shot MOT methods [31]. The network structure of cross-correlation network is shown in Fig. 1. In the figure, F represents the features provided by YOLO [38], where $F \in R^3$. F' is the statistical information obtained by feeding the features into avg-pooling layer. The convolution layers decoupled the statistical information F' to $T1$ and $T2$ corresponding to detection tasks and re-identification tasks respectively. Then, 4 soft-max layers are applied on the reshaped $M1$ and $M2$ to generate the self-relation weight maps WT_1, WT_2, WS_1, WS_2 for each task. The enhanced feature map F_{T1} and F_{T2} are obtained by adding the original feature F to matrix multiplication results. F_{T1} and F_{T2} are feature maps for detection tasks and re-identification tasks, respectively. In summary, the original feature map F performs the matrix multiplication between the reshaped feature and the learned weight maps to obtain an enhanced representation for each image frame. The enhanced representation is fused with original feature map F and residual attention to prevent information loss. The ReID module handles the enhanced feature maps for ID re-identification and then output the information about the dimension and coordinates of bounding boxes. The loss function for training the network is shown in Equation 1, which is an accumulated loss of the classification loss, L_α, and the bounding box regression loss, L_β. L_α is a cross-entropy loss and L_β is formulated as a smooth-L_1 loss, as shown in Equation 2 and 3, respectively.

$$L_{total} = \sum_{j=\alpha,\beta} L_j^i \tag{1}$$

Algorithm 1 The particle filtering algorithm.

```
1:  **Initialization** $(t = 0)$:
2:      Set the number of particles as $K$.
3:      Sample $x_0^{(k)}$ for every $k$, as $X_0^{(k)} \sim \tau_0(dx_0)$.
4:      Set the weight of each particle: $\omega_0^{(k)} = \frac{1}{K}$
5:  **while** $0 < t \leqslant T$ **do**
6:      **Sampling:**
7:          Sample $\overline{x}_t^{(k)}$, as $\overline{X}_t^{(k)} \sim \tau_t(\overline{x}_t | x_{t-1}^{(k)})$.
8:          Calculate the weights $\overline{\omega}_t^{(k)}$.
9:      **Resampling:**
10:         Sample $\widetilde{x}_t^{(k)}$ from $\left\{\overline{x}_t^{(k)} | 1 \leq k \leq K\right\}$ with probability
            $Pr(\widetilde{X}_t^{(k)} = \overline{x}_t^{(v)}) = \frac{\overline{w}_t^{(v)}}{\sum_{k=1}^K \overline{w}_t}$
11:         Set $\widetilde{\omega}_t^{(k)} = \frac{1}{K}$
12:     **Update:**
13:         $x_t^{(k)} = \widetilde{x}_t^{(k)}$
14:         $w_t^{(k)} = \widetilde{w}_t^{(k)}$
15:         Estimation: $\pi_t^N(x_t) = \sum_{k=1}^K w_t^{(k)} \delta(x_t = x_t^{(k)})$
16:         $t = t + 1$
17: **end while**
```

Where, t is the index of frame, T is the total number of frames, k is the index of particle, K is the total number of particles, $x_0^{(k)}$ is the initial states of K particles, X_0 is the initial distribution, $\tau_t(\overline{x}_t | \pi_{t-1})$ is the Markov kernel that determines the probability distribution of X_t given π_{t-1}. $\overline{x}_t^{(k)}$, $\widetilde{x}_t^{(k)}$ are the states of kth particles after importance sampling and resampling, respectively. $\overline{w}_t^{(k)}$, $\widetilde{w}_t^{(k)}$ are the weights of kth particles after importance sampling and resampling, respectively. $w_t^{(k)}$ is the weight of kth particle at time step t. $\delta(\cdot)$ is Dirac delta function. $\pi_t^K(x_t)$ is approximated by particle filters with K particles.

Where, α and β represent the foreground/background classification loss and the bounding box regression loss, respectively.

$$L_\alpha = \sum_k^K -(y_i log\hat{y}_i + (1 - y_i) log(1 - \hat{y}_i)) \tag{2}$$

Where, y_i and $\hat{y}_i$ denote the ground-truth and prediction, respectively. k represents the index of the frame.

$$L_\beta = \sum_k^K \sum_i^N \|x_i^{(k)} - \hat{x_i}^{(k)}\|_1 \tag{3}$$

Where, x_i and $\hat{x_i}$ denote the bounding box of ground-truth and prediction, respectively. k represents the index of the frame.

The MOT tracker is initialized with pre-trained parameters based on the COCO dataset [39]. Then, the network is trained with additional datasets including ETH [40], CityPerson [41], CalTech [42], MOT17 [22], CUDK-SYSU [20], and PRW [43]. The training mini-batch size is set to be 20. The initial learning rate is 110^{-3}, and it is decayed to 510^{-5} after the twentieth epoch. Adam algorithm is used as a replacement optimization algorithm for stochastic gradient descent for training the network.

C. The Proposed Tracking Model

We proposed a hybrid tracking model, namely HTracker, which alternatively executes the MOT tracker and the particle filters to improve the tracking speed. Because of the low computational time consumption of particle filters, we proposed to alternatively executive the CNN-based tracker and the particle filtering algorithm to track the object(s) as shown in Fig. 2. In the figure, t_1, t_2, t_3 and t_4 represent the time steps. A person is working from the bottom left to the top right. (x_i, y_i) represents the coordinate of the person in the frame. For t_1 and t_3, the coordinates are estimated by CNN-based tracker. For t_2 and t_4, the coordinates are estimated by particle filters. Usually, the CNN-based trackers provide relatively accurate tracking results compared with particle filters. If the person (targets) are assumed to move uniformly in the scene. The coordinates of this person can be roughly pre-estimated from previous tracking results. Since the CNN-based tracker can provide accuracy estimation, we used the estimated coordinates from the CNN-based tracker to guide the particle distribution in the following particle filter step. For example, in Fig. 2, (x_4, y_4) can be roughly estimated from (x_1, y_1) and (x_3, y_3). The frame labeled with t_4 are handled by particle filers and the particles are spread with the Gaussian distribution with a mean equal to the pre-estimated coordinates (x_4, y_4). In time step t_4, the observation for the particle filer is the coordinate (x_3, y_3), which can be represented by a cropped image with a black bounding box in time step t_3.

IV. Experiments and Results

We evaluate the proposed tracker on public datasets, MOT16, and MOT17 [22]. The MOT16 dataset and MOT17 dataset are used for multiple object tracking evaluation. The MOT16 dataset consists of seven training sequences and seven testing sequences. Some scenes include crowded pedestrians with obstacles and sometimes the targets are partially or fully occluded, which is considered to be challenging for object tracking. The MOT17 includes the same video sequences as the MOT16 dataset but has more accurate ground truth. Each sequences is provided with 3 sets of detections, including DPM [18], Faster-RCNN [44], and SDP [45] for more comprehensive comparisons. For a fair comparison, all the evaluation results are obtained from the public platform, "The Multiple Object Tracking Benchmark" at https://motchallenge.net/, with a similar computer configuration. The proposed model is compared with the start-of-the-art tracking models. Our computation platform includes a RTX 2080 Ti GPU and a Intel i7-9700 CPU.

A. Evaluation on Different Similarity Functions

The weights of particles are calculated by using similarity functions in the sampling stage of particle filters and the weights decide the particle selection in the following resampling stage. The estimated coordinates of objects are the expectation of all remaining particles after the resampling stage. It is interesting to observe the difference of estimate accuracy caused by different choice of the similarity function. Particles with high weights are considered to be good particles since they contain more information about the target. For the object tracking task, the weight of a particle can be calculated by comparing the similarity between the cropped image (from this particle) and the observation. Usually, a particle with the corresponding bounding box having more overlapped areas with the observation image has a higher weight and this particle is considered to be a good particle.

An ideal similarity function can differentiate good particles from bad particles by assign weights with a large difference and those good particles will be selected and copied to estimate the coordinates of objects in the resampling stage. Otherwise, both good particles and bad particles will be kept in resampling stages and thus the resultant estimates have low accuracy. Another criterion of selecting a similarity function is the computation cost. Only the high-efficient similarity functions can be used in particle filters in order to boost the tracking speed.

Based on those two criteria, we selected some similarity functions including the histogram on gray images, the histogram on RGB images, the perceptual hash (pHash), and the cosine similarity. The histogram is the statistical information about the distribution of pixel values. The gray image is an image with one layer and the pixel value is from 0 to 255. The RGB image is an image with three layers corresponding to red, green, and blue color layers. A pHash value represents the image fingerprint. An image is initially resized to 16 by 16 and then converted to a gray image. A discrete cosine transform (DCT) is applied to the gray image to form a coefficient matrix, which is the fingerprint of the original image. The Hamming distance of pHash values from two images can be used to decide whether the two images are similar. If the Hamming distance from two images is short, the content features of these two images can be considered to be similar. The cosine distance measures the image similarity by measuring the angle between two vectors formed from two images as shown in Equation 4. In the equation, the vectors A and B are formed by stacking the pixel values of two images. The nominator is the dot product of vectors and the denominator is the product of the L2 norm of these two vectors.

$$cosine\_similarity = cos(\theta) = \frac{A \cdot B}{\|A\| \, \|B\|} \tag{4}$$

We evaluate different similarity functions in the proposed tracker on MOT16 and MOT17 datasets. The experimental results are shown in Table I and II. In the tables, the measurements with red color and blue color represent the 1st and 2nd best results. respectively. Among the similarity functions, the pHash obtains the best estimation accuracy, but the lowest tracking speed. In order to balance the estimation accuracy and tracking efficiency, we use the Histogram (RGB) as the similarity function in the proposed tracker.

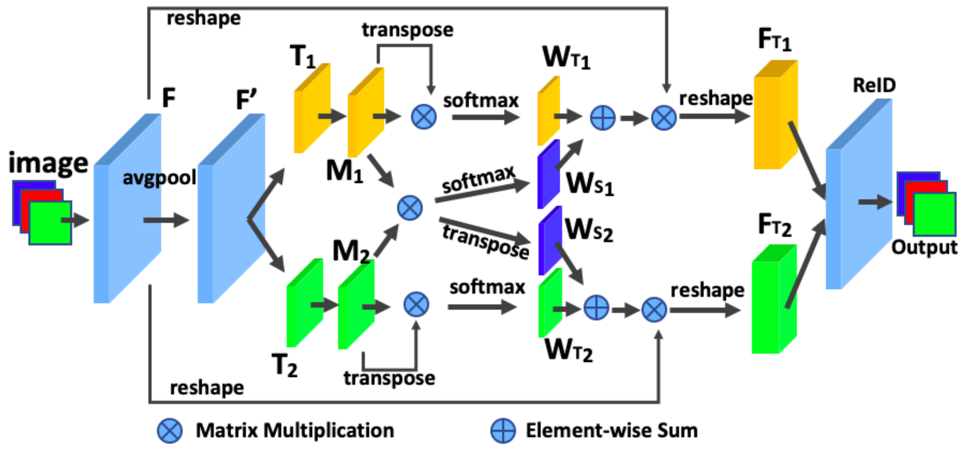

Figure 1. The network structure of MOT tracker.

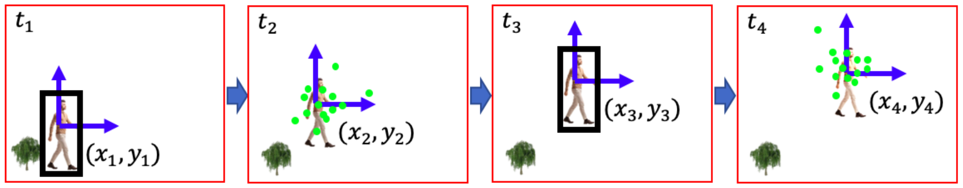

Figure 2. The workflow of the proposed HTracker. t_1, t_2, t_3, and t_4 are the time steps.(x_i, y_i) is the estimated coordinate of the target (person). The green dots are the particles. The black bounding boxes represent the estimated coordinates of targets from the CNN-based tracker.

B. Evaluation of Proposed Tracker

We evaluate the proposed tracker on the MOT16 and MOT17 datasets. The evaluation metrics include MOTA, IDF1, MT, ML, and Hz (tracking speed). The detailed explanation of these metrics is as follows:

- MOTA: Multi-object tracking accuracy. This measure combines three error sources: false positives, missed targets, and identity switches.

$$MOTA = 1 - \frac{\sum_t (FN_t + FP_t + IDSW_t)}{\sum_t GT_t} \quad (5)$$

- IDF1: ID F1 score. The ratio of correctly identified detection over the average number of ground-truth and computed detection.
- MT: Mostly tracked targets. The ratio of ground-truth trajectories that are covered by a track hypothesis for at least 80% of their respective life span.
- ML: Mostly lost targets. The ratio of ground-truth trajectories that are covered by a track hypothesis for at most 20% of their respective life span.
- Hz: Processing speed in frames per second.

The experimental results are shown in Table III and IV, and Fig. 3 and 4, respectively. All the results are from the public benchmarks with the same hardware configurations to make a fair comparison. In Table III and IV, the results with the red color and the blue color indicate the best results and the second-best results, respectively. The results show that the proposed tracker has a high tracking speed, 38.2 Hz on MOT16 dataset and 36.7 Hz on MOT17 dataset, which realize a new state-of-the-art tracking speed to satisfy real-time usages. Compared with the state-of-the-art CSTracker [31], the tracking accuracy is lowered by 11.2% and 10.7%, but the tracking speed increased by 141.8% and 132.3% on the MOT16 and the MOT17, respectively. In summary, our proposed tracker achieves the best tracking speed among these top trackers and good MOTA, IDF1, MT, and ML scores to

Table I
EVALUATION OF HTRACKER WITH DIFFERENT SIMILARITY FUNCTION ON MOT16

Similarity Functions	MOTA↑	IDF1↑	MT↑	ML↓	Hz↑
Histogram (Gray)	66.9	69.7	219	149	38.8
Histogram (RGB)	67.1	71.1	221	146	38.2
pHash	67.8	72.4	232	139	34.2
Cosine	64.1	66.1	203	169	39.1

Table II
EVALUATION OF HTRACKER WITH DIFFERENT SIMILARITY FUNCTIONS ON MOT17

Similarity Functions	MOTA↑	IDF1↑	MT↑	ML↓	Hz↑
Histogram (Gray)	66.3	69.3	652	498	37.2
Histogram (RGB)	66.9	70.4	667	490	36.7
pHash	67.2	70.9	672	483	33.2
Cosine	62.5	65.6	637	529	38.3

lower the latency, while keeping good tracking accuracy.

Although, there exists some high speed trackers, their tracking accuracies are still low. We compare the proposed tracker with high speed trackers based on the public records from MOT16 and MOT17 challenges as shown in Table V and VI, respectively. For the MOT16 challenge, the tracking accuracy of the proposed tracker is 7.532% higher than that of the tracker named MCMOT_HDM. For the tracking speed, the proposed tracker is 22.5% lower than the tracker named LP2D. For the MOT17 challenge, the tracking accuracy of the proposed tracker is 47.0% higher than that of the tracker named EDA_GNN. For the tracking speed, the proposed tracker is 6.6% lower than the tracker named EDA_GNN. Our tracker provides higher tracking accuracy than these trackers mentioned above, and keeps tracking speed higher than 30 Hz, which enable the real-time application.

Table III
PERFORMANCE COMPARISON WITH HIGH-ACCURACY MODELS ON MOT16 DATASET

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	Hz↑
CSTracker [31]	75.6	73.3	325	125	15.8
UnsupTrack [9]	62.4	58.5	205	242	1.9
MLT [10]	52.8	62.6	160	322	5.9
MOTRF [11]	57.9	41.7	216	168	11.1
FairMOT_re [12]	60.0	62.1	191	522	22.1
HTracker(ours)	67.1	71.1	221	146	38.2

Table IV
PERFORMANCE COMPARISON WITH HIGH-ACCURACY MODELS ON MOT17 DATASET

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	Hz↑
CSTracker [31]	74.9	72.6	978	411	15.8
UnsupTrack [9]	61.7	58.1	640	762	2.0
CTTrackPub [46]	61.5	59.6	621	752	17.0
Lif_T [47]	60.5	65.6	637	1,189	0.5
Lif_TsimInt [47]	58.2	65.2	674	1,022	4.6
HTracker(ours)	66.9	70.4	667	490	36.7

Table V
PERFORMANCE COMPARISON WITH HIGH-SPEED MODELS ON MOT16 DATASET

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	Hz↑
MCMOT_HDM [22]	62.4	51.6	239	184	39.9
LP2D [22]	35.7	34.2	66	385	49.3
HTracker(ours)	67.1	71.1	221	146	38.2

Table VI
PERFORMANCE COMPARISON WITH HIGH-SPEED MODELS ON MOT17 DATASET

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	Hz↑
EDA_GNN [22]	45.5	40.5	368	955	39.3
GM_PHD [22]	36.4	33.9	97	1,349	38.4
HTracker(ours)	66.9	70.4	667	490	36.7

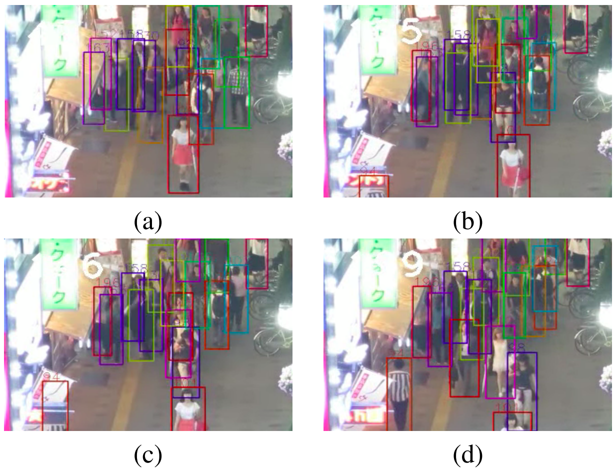

Figure 3. The tracking results by using the proposed tracker on a video sequence in MOT16 dataset.

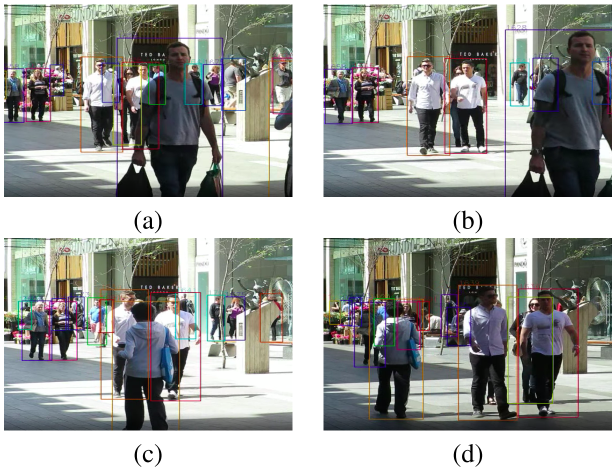

Figure 4. The tracking results by using the proposed tracker on a video sequence in MOT17 dataset.

V. CONCLUSIONS

In this study, we explore boosting the tracking speed by combining the CNN-based tracker and particle filters in MOT tasks to deal with the video sequence and image sequence with high frame rates for satisfying the real-word application. The experimental results indicate that the tracking speeds are improved significantly with competitive tracking accuracies. Our future work includes the following. First, we will explore the CNN-based resampling method in particle filters to improve the tracking accuracy, since the resampling procedures greatly influence the estimation quality. Second, we will explore lowering the computation consumption of CNN-based trackers to speed up the tracking.

REFERENCES

[1] J. Xing, H. Ai, and S. Lao, "Multiple human tracking based on multi-view upper-body detection and discriminative learning," in 2010 20th International Conference on Pattern Recognition. IEEE, 2010, pp. 1698–1701.

[2] S. Challa, M. R. Morelande, D. Mušicki, and R. J. Evans, Fundamentals of object tracking. Cambridge University Press, 2011.

[3] E. Foxlin, “Pedestrian tracking with shoe-mounted inertial sensors,” IEEE Computer graphics and applications, vol. 25, no. 6, pp. 38–46, 2005.
[4] S. Kamate and N. Yilmazer, “Application of object detection and tracking techniques for unmanned aerial vehicles,” Procedia Computer Science, vol. 61, pp. 436–441, 2015.
[5] G. Zhang and P. A. Vela, “Good features to track for visual slam,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1373–1382.
[6] L. Liu, J. Xing, H. Ai, and X. Ruan, “Hand posture recognition using finger geometric feature,” in Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012). IEEE, 2012, pp. 565–568.
[7] A. Petrovskaya and S. Thrun, “Model based vehicle detection and tracking for autonomous urban driving,” Autonomous Robots, vol. 26, no. 2-3, pp. 123–139, 2009.
[8] M. Dimitrievski, P. Veelaert, and W. Philips, “Behavioral pedestrian tracking using a camera and lidar sensors on a moving vehicle,” Sensors, vol. 19, no. 2, p. 391, 2019.
[9] S. Karthik, A. Prabhu, and V. Gandhi, “Simple unsupervised multi-object tracking,” arXiv preprint arXiv:2006.02609, 2020.
[10] Y. Zhang, H. Sheng, Y. Wu, S. Wang, W. Ke, and Z. Xiong, “Multiplex labeling graph for near-online tracking in crowded scenes,” IEEE Internet of Things Journal, vol. 7, no. 9, pp. 7892–7902, 2020.
[11] J. Lee, S. Kim, and B. C. Ko, “Online multiple object tracking using rule distillated siamese random forest,” IEEE Access, vol. 8, pp. 182 828–182 841, 2020.
[12] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, “Fairmot: On the fairness of detection and re-identification in multiple object tracking,” arXiv: 2004.01888, 2020.
[13] Y. Wu, J. Lim, and M.-H. Yang, “Online object tracking: A benchmark,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2013, pp. 2411–2418.
[14] W. Luo, J. Xing, A. Milan, X. Zhang, W. Liu, and T.-K. Kim, “Multiple object tracking: A literature review,” Artificial Intelligence, p. 103448, 2020.
[15] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: the clear mot metrics,” EURASIP Journal on Image and Video Processing, vol. 2008, pp. 1–10, 2008.
[16] L. Zheng, Y. Yang, and A. G. Hauptmann, “Person re-identification: Past, present and future,” arXiv preprint arXiv:1610.02984, 2016.
[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 779–788.
[18] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, “Object detection with discriminatively trained part-based models,” IEEE transactions on pattern analysis and machine intelligence, vol. 32, no. 9, pp. 1627–1645, 2009.
[19] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” IEEE transactions on pattern analysis and machine intelligence, vol. 39, no. 6, pp. 1137–1149, 2016.
[20] T. Xiao, S. Li, B. Wang, L. Lin, and X. Wang, “Joint detection and identification feature learning for person search,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3415–3424.
[21] Y. Wang, X. Weng, and K. Kitani, “Joint detection and multi-object tracking with graph neural networks,” arXiv preprint arXiv:2006.13164, 2020.
[22] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, “Mot16: A benchmark for multi-object tracking,” arXiv preprint arXiv:1603.00831, 2016.
[23] C. Shan, T. Tan, and Y. Wei, “Real-time hand tracking using a mean shift embedded particle filter,” Pattern recognition, vol. 40, no. 7, pp. 1958–1970, 2007.
[24] C. Chang and R. Ansari, “Kernel particle filter for visual tracking,” IEEE signal processing letters, vol. 12, no. 3, pp. 242–245, 2005.
[25] F. Gustafsson, “Particle filter theory and practice with positioning applications,” IEEE Aerosp. Electron. Syst. Mag., vol. 25, no. 7, pp. 53–82, 2010.
[26] P. Del Moral, “Non-linear filtering: interacting particle resolution,” Markov Process. Related Fields, vol. 2, no. 4, pp. 555–581, 1996.
[27] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, “A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking,” IEEE Trans. Signal Process., vol. 50, no. 2, pp. 174–188, 2002.
[28] L. Kocarev and U. Parlitz, “General approach for chaotic synchronization with applications to communication,” Phys. Rev. Lett., vol. 74, no. 25, p. 5028, 1995.
[29] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, “Robust tracking-by-detection using a detector confidence particle filter,” in 2009 IEEE 12th ICCV. IEEE, 2009, pp. 1515–1522.
[30] T. Bokareva, W. Hu, S. Kanhere, B. Ristic, N. Gordon, T. Bessell, M. Rutten, and S. Jha, “Wireless sensor networks for battlefield surveillance,” in Process. land warfare Conf., 2006, pp. 1–8.
[31] C. Liang, Z. Zhang, Y. Lu, X. Zhou, B. Li, X. Ye, and J. Zou, “Rethinking the competition between detection and reid in multi-object tracking,” arXiv preprint arXiv:2010.12138, 2020.
[32] K. Okuma, A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe, “A boosted particle filter: Multitarget detection and tracking,” in ECCV. Springer, 2004, pp. 28–39.
[33] B. S. Kerner, H. Rehborn, R.-P. Schäfer, S. L. Klenov, J. Palmer, S. Lorkowski, and N. Witte, “Traffic dynamics in empirical probe vehicle data studied with three-phase theory: Spatiotemporal reconstruction of traffic phases and generation of jam warning messages,” Physica a: statistical mechanics and its applications, vol. 392, no. 1, pp. 221–251, 2013.
[34] K. Hwang and W. Sung, “Load balanced resampling for real-time particle filtering on graphics processing units,” IEEE Trans. Signal Process., vol. 61, no. 2, pp. 411–419, 2012.
[35] X. Zhang, L. Huang, E. Ferguson-Hull, and F. Gu, “Adaptive particle routing in parallel/distributed particle filters,” in Proc. of the 25th High Performance Computing Symposium. Society for Computer Simulation International, 2017, p. 10.
[36] X. Zhang, L. Zhao, W. Zhong, and F. Gu, “Performance analysis of resampling algorithms of parallel/distributed particle filters,” IEEE Access, 2020.
[37] Z. Wang, L. Zheng, Y. Liu, and S. Wang, “Towards real-time multi-object tracking,” arXiv preprint arXiv:1909.12605, 2019.
[38] J. Redmon and A. Farhadi, “Yolov3: An incremental improvement,” arXiv preprint arXiv:1804.02767, 2018.
[39] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in European conference on computer vision. Springer, 2014, pp. 740–755.
[40] A. Ess, B. Leibe, K. Schindler, and L. Van Gool, “A mobile vision system for robust multi-person tracking,” in 2008 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2008, pp. 1–8.
[41] S. Zhang, R. Benenson, and B. Schiele, “Citypersons: A diverse dataset for pedestrian detection,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 3213–3221.
[42] P. Dollár, C. Wojek, B. Schiele, and P. Perona, “Pedestrian detection: A benchmark,” in 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2009, pp. 304–311.
[43] L. Zheng, H. Zhang, S. Sun, M. Chandraker, Y. Yang, and Q. Tian, “Person re-identification in the wild,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1367–1376.
[44] R. Girshick, “Fast r-cnn,” in Proceedings of the IEEE international conference on computer vision, 2015, pp. 1440–1448.
[45] F. Yang, W. Choi, and Y. Lin, “Exploit all the layers: Fast and accurate cnn object detector with scale dependent pooling and cascaded rejection classifiers,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 2129–2137.
[46] X. Zhou, V. Koltun, and P. Krähenbühl, “Tracking objects as points,” arXiv preprint arXiv:2004.01177, 2020.
[47] A. Hornakova, R. Henschel, B. Rosenhahn, and P. Swoboda, “Lifted disjoint paths with application in multiple object tracking,” in International Conference on Machine Learning. PMLR, 2020, pp. 4364–4375.