Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 7437289, 18 pages
https://doi.org/10.1155/2022/7437289

Research Article

Real-Time 3D Pedestrian Tracking with Monocular Camera

Peng Xiao,[1,2] **Fei Yan**,[1] **Jiannan Chi**,[2,3] **and Zhiliang Wang**[1]

[1]*School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing 100083, China*
[2]*Engineering Research Center of Intelligence Perception and Autonomous Control, Beijing University of Technology, Beijing 100124, China*
[3]*School of Automation and Electronic Engineering, University of Science and Technology Beijing, Beijing 100083, China*

Correspondence should be addressed to Jiannan Chi; ustbjnc@ustb.edu.cn

Received 3 October 2021; Revised 5 November 2021; Accepted 25 November 2021; Published 17 February 2022

Academic Editor: Chao-Yang Lee

Target tracking has always been a popular research area in computer vision, and many important methods have been proposed. However, most methods can only solve partial and slight occlusion. If the target is lost, a common solution is to keep detecting, reidentify the target when it reappears, and then link the broken tracks together, but this makes tracking discontinuous. There are two key points in this problem: continuous tracking and occlusion judgment. In this paper, we propose a target tracking method with a short-time prediction function to solve this problem. For continuous tracking, we establish a 3D dynamic model to estimate the motion state of the target in each frame. For occlusion judgment, we use a depth prediction network to estimate the depth of the target and then determine whether the target is occluded by the depth. Without relying on depth sensors or multiple cameras, we achieve depth estimation using only a single monocular image, which greatly expands the application of our method. Benefit from the introduction of motion estimation and depth prediction, the tracking accuracy of our method has been significantly improved, especially for better robustness to occlusion. Even when the target is completely occluded, it can be tracked for a short time without reidentification. In addition, we improve the speed of depth prediction through knowledge distillation by 2.08 times, and the final tracking speed reaches 52.6 Hz on GPU, which meets the real-time tracking requirements.

1. Introduction

Modern society produces a large number of videos every day. As an important means of video analysis, video object tracking has a wide range of applications, such as autonomous driving [1], robotics [2], and augmented reality [3]. Although great progress has been made over these years, most methods are based on the assumption that the target is visible. Therefore, these methods can only solve partial and slight occlusion problems. However, in daily life, the target is often completely obscured. There are some ways to solve this problem, and the mostly used one is reidentification, but this breaks the continuity of the tracking. Even if the target is not visible, it still exists, and we should speculate its location based on our experience. And in some cases, we cannot pay the consequences of ignoring the completely missing target, such as online intelligent driving. To solve this problem, we can start from two aspects: continuous tracking and occlusion judgment.

The key to continuous tracking is that the tracker should be able to use the historical information of the target's motion to estimate the current motion state of the target. Usually we assume that the motion of the target conforms to certain rules, so we can establish its motion model. Even if the target is lost, we can also infer the target's state according to the model. Regression model is a common prediction model in engineering, but regression model is not applicable to state prediction in tracking problem. Because the regression model requires that all points in state spaces roughly conform to a known relationship, such as a linear relationship or a polynomial relationship, which is almost impossible to satisfy in visual tracking. In addition, the regression model reflects the overall trend, and the prediction error of a single point may be very large, which may even directly

lead to tracking failure. A more practical prediction method is Kalman filtering (KF) [4], which has been widely used in tracking problems. It first predicts the ideal state value through the state transition equation and then corrects the predicted state value and the model based on the actual measured value. Compared with regression models, KF has three advantages. The first is low computation cost. Only five formulas are needed to perform in each a time step. The second is flexible. We can customize the target's state values that need to be estimated, such as coordinates, aspect ratio, speed, and acceleration, and the relationship between state values can also be defined by ourselves. The third is strong adjustment ability. Some parameters in KF model can be adjusted over time to ensure that the model conforms to the current motion state as much as possible. After comprehensive consideration, KF will be used for state estimation in this paper.

Occlusion judgment is a very challenging problem, especially for 2D tracking. In the visual tracking problem, occlusion occurs frame by frame, and the target and the block are gradually mixed together. There is no general and effective way to separate the target and the block or to measure the level of occlusion. In 3D tracking, the tracker usually calculates the distance between the target and the camera, that is, the depth. It is easy to determine whether and how much the target is occluded by simply checking the depth value of key points on the target. Since the block is nearer to the camera than the target, the depth value of the target region will suddenly increase once occlusion occurs. The only problem is how to obtain the depth value. The most direct way to obtain depth information is to use a depth sensor, such as Kinect and laser scanner. Although the hardware-based approaches can obtain accurate depths, the dependence on hardware also greatly limits the applicability of these approaches, and they cannot handle images that are already captured without depth information. For most tracking tasks, we only need to distinguish the relative position of each object in space, not the absolute position, so it is a more practical solution to estimate the relative position of the object through a model. This has been a hot topic in 3D tracking in recent years, and we will discuss this in detail in Section 2. In this paper, we will introduce a practical neural network [5] to predict the depth due to its excellent performance and published depth dataset. To speed up depth prediction, we carry out a knowledge distillation to the original net [5] and get great improvement.

Another point to focus on is target detection. Most of the state-of-the-art (SOTA) tracking methods are tracking-by-detection (TBD) framework, and the performance of the detector directly affects the accuracy of target localization. Before the rise of deep learning, landmark target detection methods include Viola Jones Detectors [6], HOG Detector [7], and Deformable Part-based Model (DPM) [8]. These well-designed artificial feature detectors often achieve good results on specific tasks but had poor generalization capabilities. Since 2014, various deep learning-based target detection methods having continuously refreshed the record of target detection and many classic methods have also been proposed, such as RCNN [9], Faster RCNN [10], YOLO [11], SSD [12], RetinaNet [13], and Fast YOLO [14]. Here, we select Fast YOLO [14] as private target detector because of its high speed.

In this paper, we study the pedestrian tracking problem in video surveillance and aim to achieve real-time continuous 3D tracking using a single camera. Inspired by SORT [15], we propose a multipedestrian tracking method incorporating depth information. The basic idea is to use a Kalman filter to continuously estimate the motion state parameters for each target. To ensure the continuity of tracking, we do prediction when occlusion happens or the track does not match any target. The measurement of a KF consists of two parts: plane information and depth information, which are obtained by a target detection neural network and depth prediction network, respectively. To ensure real-time tracking, the neural networks used are trained in advance and are not updated online. Figure 1 shows the tracking effect of our method. When pedestrians are partially or completely occluded, our tracker still keeps tracking. In contrast, most trackers give up tracking once they cannot detect the targets.

To summarize, this paper presents a method for 3D tracking with a fixed monocular camera. The contributions of our work are summarized as follows:

(1) We utilize KF's short-term prediction capabilities to achieve continuous tracking. The original SORT method cannot detect occlusion, so when the target is occluded, its trajectory has to be ended. While we introduce depth information to make occlusion detection very easy

(2) We fuse the uncalibrated depth information to achieve 3D tracking on 2D images. We use a neural network to estimate the depth information of the target in real time, thereby turning the original 2D tracking into a 3D tracking

(3) We take a series of measures to speed up tracking. The most important strategy is knowledge distillation of the deep prediction network. Beside this, we select a faster target detector and remove reidentification step that is usually used in MOT methods

The remainder of this paper is organized as follows. Section 2 introduces some related studies about multitarget detection, tracking, and depth prediction. Section 3 presents the overall architecture of our method and details its improved parts. In Section 4, the experimental results verify the effectiveness of the proposed method. In Section 5, we provide a comprehensive summary of this study, and a future research direction is presented.

2. Related Work

2.1. Multiple Pedestrian Tracking. Compared with single object tracking (SOT), multiple object tracking (MOT) is more complicated because there are extra issues to be considered, such as the matching of trajectory and target and target reidentification. Multiple pedestrian tracking (MPT)

Figure 1: Visualization of tracking when occlusion happens. Red bounding boxes represent occluded pedestrians that are tracked by our method. The figures below are partial enlargement of the above ones.

has become the main research direction of MOT. The related research work focuses on the following four areas for improvement: (a) design the association methods, (b) joint other vision tasks, (c) apply deep learning to MPT, and (d) multi-modality-based MPT. The core of TBD framework is data association. Some classical association methods are still used as basic algorithms. Hungarian method is a classic algorithm for solving the minimum weight matching problem of bipartite graph and is introduced into MPT by Singh et al. [16]. Although the algorithm is fast, the accuracy is not high due to local optimal nature. NF [17] extends local optimization to global optimization, and CRF [18] further considers association dependencies. GMMCP [19] are proposed to solve the problem of high computation complexity. Different from these graph-based methods, MCSM [20] formulates the association as a minimum cost subgraph multicut problem that links and performs clustering for the multiple plausible person detection jointly over time and space. Several researchers have leveraged other vision tasks to improve the tracking performance. One approach is to treat MOT as an extension of SOT. For example, Hu et al. [21] use Siamese-RPN to locate the target location. The other is to combine with image segmentation. Voigtlaender et al. [22] propose a new baseline method which jointly addresses detection, tracking, and segmentation with a single convolutional network. Since Kim et al. [23] first utilize CNN to extract 4096-dimensional features for each detection box, deep neural network models have been widely used in MPT, such as VGGNet [24, 25] and GoogleNet [21, 26]. When a single type of data is unreliable, people try to use multimodal data, such as Zhang et al. [27] using image and point cloud features and Gautam et al. [28] using image and radar features.

2.2. Depth Prediction. Scene depth estimation is an old-standing problem in vision. The most direct way to obtain depth information is to use a depth sensor, such as Kinect and laser scanner. Although sensors can capture accurate depth information, they are only applicable to specific scenes and cannot process existing RGB images. In recent years, using deep neural networks to estimate scene depth has become a mainstream research direction. Various DCNN-based methods focus on designing structural features, especially in depth prediction. Fu et al. [29] propose an encoder-decoder network, which extracts multiscale features from the encoder and is trained in an end-to-end manner without iterative refinement. Jiao et al. [30] propose an attention-driven loss, which merges the semantic priors to improve the prediction precision on unbalanced distribution datasets. Chen et al. [31] apply the generative adversarial training to lead the network to learn a context-aware and patch-level loss automatically.

The basic idea of these methods is to train a deep network model with a RGBD dataset and then reconstruct the 3D structure of the target from the RGB image to predict the depth information. Deep network models are data-driven, and the models achieve good results when trained with large amounts of samples, but it is still not an easy task to obtain large amounts of RGBD data. Silberman et al. [32] construct the NYU dataset using Kinect, but it was limited to indoors. Although the datasets Make3D [33] and KITTI [34] built with laser scanners can be used outdoors, they are collected in specific scenarios (a university campus and atop a car, respectively). Another way to collect depth data for training is to ask people to manually annotate depth in images, but it is not only time-consuming and laborious but also can only give the relative position of the object. Estimating geometry from Internet photo collections has been an active research area for a decade. Li et al. [5] propose a method to generate an infinite dataset. First, a large number of images are collected from the Internet, and then, the structure-from-motion (SfM) and multiview stereo (MVS) method are used to generate depth maps, and these depth maps are further processed to form a large-scale depth data set MegaDepth (MD).

2.3. Pedestrian Detection. The deep feature-based detection approaches have become the main direction of pedestrian detection research because of their SOTA performance. In fact, many general detection methods are also suitable for

pedestrian detection, so we do not distinguish between them. We strongly recommend researchers to refer the surveys [35, 36], which give detailed summary about target detection. Here, we only review some classical algorithms. Ren et al. [37] propose a Recurrent Rolling Convolution (RRC) architecture, which can selectively integrate contextual information into the bounding box regressor. Liu et al. [12] propose SSD, the first deep network based object detector that does not resample pixels or features for bounding box hypotheses and is as accurate as approaches that do. Liu et al. [38] propose the ALFNet, which trains the SSD in multisteps and significantly improves the accuracy of pedestrian detection while maintaining the efficiency of the single-stage detector. Most detection methods are trained on datasets without occlusion or with reasonable occlusion. Once heavy occlusion occurs, the performance will decrease significantly. Therefore, the recent benchmark test pays special attention to pedestrian detection with heavy occlusion. Zhang et al. [39] design a new regression loss and introduce a part occlusion aware region of interest (PORoI) pooling unit to solve the problem of occluded pedestrian detection in crowded scenes. Tian et al. [40] design a set of component detectors; each component is designed to handle a specific occlusion mode. Zhou and Yuan [41] use a neural network to locate the full body and visible part of a pedestrian, respectively.

3. Method

The flowchart of the proposed multipedestrian tracking algorithm is shown in Figure 2. First camera motion is calculated to reduce the global error. Then, each KF predicts its trajectory at time k, denoted as $\tilde{\mathbf{S}}_k$, according to the former estimated trajectory $\mathbf{S}_{k-1}$. Next, a target detection network and a depth prediction network, respectively, extract the plane information and depth information of the target from the image at current time k, which constitute the measurement $\mathbf{O}_k$ for KF. After that, the tracks and detected targets are matched with Hungarian method. There are three results for matching, and each result corresponds to a processing way. (1) For unmatched target, if a target does not match any track, the target will be treated as the starting point of a new track and a new KF will be initialized. (2) Unmatched track, if a track has not matched any target for several consecutive frames, the track is considered to have lost the target completely, and keep tracking is meaningless, so the track will be delete from tracking system. Otherwise, we treat the track as a temporary disappearance and update KF. (3) For matched track, if a track matches a target, the system will detect occlusion and then update KF. Finally, the added new tracks and the updated tracks compose the tracking results at time k. In the following subsections, we will elaborate on several key points in the algorithm.

3.1. State Estimation

3.1.1. KF Prediction. Kalman filter is a classic time series estimation method, which is very suitable for estimating the motion state of the target. The state can be any parameter related to the target, such as center position, height, aspect ratio, and their respective velocities in image coordinates. According to the state of the target at the previous time, KF calculates a prediction value at time k by $\tilde{\mathbf{S}}_k = g(\mathbf{S}_{k-1}, \mathbf{Q})$, where $\mathbf{Q}$ is a Gaussian noise that reflects the accuracy of the process model and g is a function that describes the state change law of the target in two adjacent frames. In multitarget tracking, g is usually assumed to be a constant noise model.

3.1.2. Observation Calculation. A target detection network and a depth prediction network, respectively, process the image at time k and obtain the plan and depth map of the targets, which constitute the measurement value $\mathbf{O}_k$. We adopt fast-YOLO [14] for target detection, which greatly reduces the number of deep inferences and speeds up the detection. For depth prediction, we follow the monocular depth estimation method open-sourced in paper [5]. To reduce the calculation, we use the trick of knowledge distilling to transfer the original depth prediction network to a 5-layer student CNN. The teacher network is trained on MD dataset [5]. Then, the student network is further trained to predict semantic segmentation maps from depth image. Eventually, the accuracy of the student network can be close to that of the teacher network, but the model size is much smaller, thus achieving accelerated computation. Both the two networks are trained offline and are not updated online when tracking.

3.1.3. KF Update. The final state value $\mathbf{S}_k$ is composed of two parts: the predicted value and the corrected term,

$$\mathbf{S}_k = \tilde{\mathbf{S}}_k + K\left(\mathbf{O}_k - \mathbf{T}\tilde{\mathbf{S}}_k\right), \tag{1}$$

where K is the Kalman gain and $\mathbf{T}$ is the state transition matrix. For multitarget tracking, we match tracks with targets and get three results: a target does not match any track, a track matches a target, and a track does not match any target. The first case indicates that the target emerges lately, so a new KF needs to be instantiated. Both the second and third cases can be calculated by Equation (1), which will be detailed in Section 3.3.

3.2. Constructing 3D Motion Model. Equipped with depth estimates, we construct a 3D linear motion model with a constant velocity assumption. The state of each target is modelled on the ten dimensional state space $(X, Y, Z, A, H, \dot{X}, \dot{Y}, \dot{Z}, \dot{A}, \dot{H})^T$ that contains the bounding box center position (X, Y, Z), aspect ratio A, height H, and their respective velocities. The observations of the target are (X, Y, Z, A, H).

The state change of the target at two adjacent moments can be denoted by the state equation:

$$\mathbf{S}_t = \mathbf{S}_{t-1} + \dot{\mathbf{S}}_{t-1} + \mathbf{q}_t, \tag{2}$$

where $\mathbf{q}_t \in N(0, \mathbf{Q})$ is process noise and $\mathbf{Q}$ is the process noise covariance matrix. And we can derive the following formulas for X,

$$X_t = X_{t-1} + \dot{X}_{t-1} + r_X. \tag{3}$$

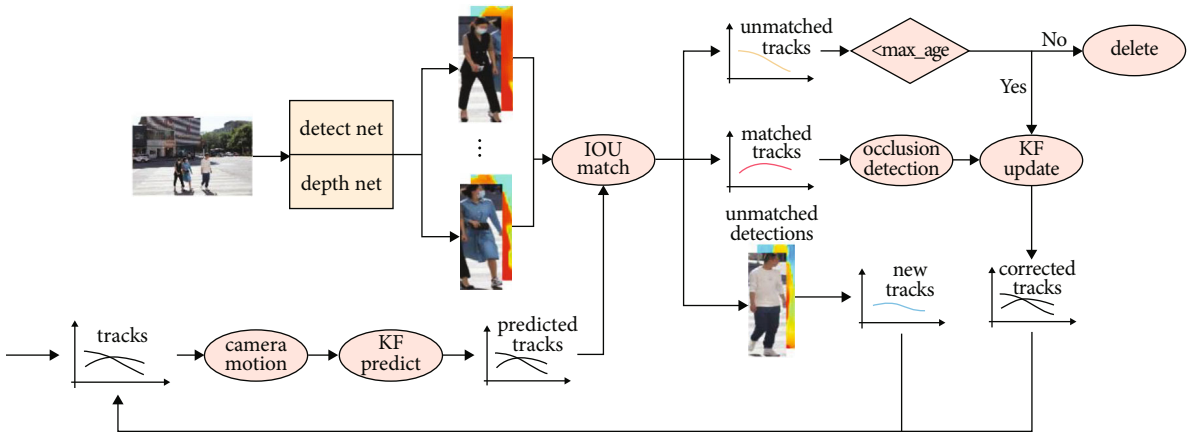


Figure 2: The flowchart of our tracking method.

Similarly, we can get the equations of Y, Z, A, and H. The intrinsic relationship between state $\mathbf{S}_t$ and observation $\mathbf{O}_t$ can be denoted by the measurement equation:

$$\mathbf{O}_{t-1} = \mathbf{S}_{t-1} + \mathbf{r}_{t-1}, \tag{4}$$

where $\mathbf{r}_t \in N(0, \mathbf{R})$ is observation noise and $\mathbf{R}$ is the observation noise covariance matrix.

Inverse depth is a commonly used representation predicted due to the ability to represent points at infinity and to model uncertainty in pixel disparity space. So the depth used is $1/Z$. In order to simplify notation, we assume the camera focal length f is a constant. In fact, f can be folded into a motion noise parameter and can be easily tuned on a training set. Then, the adjusted parameters of the bounding boxes are as follows:

$$x = f\frac{X}{Z}, y = f\frac{Y}{Z}, z = \frac{1}{Z}, a = A, h = f\frac{H}{Z}. \tag{5}$$

This means that we dynamically scale the object with inverse depth. If depths are smooth over time, we can take Z_{t-1} as an approximation of Z_t, so we derive the following formulas from Equation (3),

$$f\frac{X_t}{Z_t} \approx f\frac{X_t}{Z_{t-1}} = f\frac{X_{t-1}}{Z_{t-1}} + f\frac{\dot{X}_{t-1}}{Z_{t-1}} + f\frac{r_X}{Z_{t-1}}, \tag{6}$$

which is

$$x_t = x_{t-1} + \dot{x}_{t-1} + f\frac{r_X}{Z_{t-1}}. \tag{7}$$

The equation suggests that one can approximately apply a Kalman filter on 2D image measurements augmented with a temporal noise model that is scaled by the estimated inverse-depth of the object.

3.3. Occlusion and Unmatched Tracks. Due to the use of depth information, we can easily determine whether the target is occluded by comparing the depth values in $\tilde{\mathbf{S}}_k$ and $\mathbf{O}_k$, because objects with smaller depth are always in front of objects with larger depth. To avoid accidental errors, we take the average depth of all points near the predicted location as the observed depth. The area size is 1/4 of the predicted bounding box. When $z_k^{(\tilde{S})} < z_k^{(O)}$, we consider that the target is occluded, and $\mathbf{O}_k$ is the information of the occluder rather than the target, so $\mathbf{S}_k$ cannot be directly calculated by Equation (1). In this case, we take the KF prediction $\tilde{\mathbf{S}}_k$ as the approximate state value $\mathbf{S}_k$, which means the observation value $\mathbf{O}_k$ is completely accurate without error.

If a track does not match any target, it may because the target moves out of the image, not be detected, or not matched. To reduce the impact due to missed detections or matching errors, we introduce a counter for each track to count the number of frames since the last successful measurement association, denoted as c_k. When the track matches a target, the counter is reset to 0. When the track does not match any target, KF continues to predict the state of the target while c_k increases. If the track does not match any target for several consecutive frames $c_k\_\max$, there is a high probability that the target is lost, and the tracking of this track should be ended. It can be noticed that measurement association also works as identification. In most MOT methods, once a target is lost, its track will break off. So, when a new target appears, we have to judge whether the target is a new one or a lost one through an extra reidentification step. However, benefit from KF and occlusion detection, the track will not be interrupted in our method, so the reidentification is omitted.

In short, there are two ways to update $\mathbf{S}_k$ according to different situations,

$$\mathbf{S}_k = \begin{cases} \tilde{\mathbf{S}}_k & z_k^{(\tilde{S})} < z_k^{(O)} \text{ or } c_k < c_k\_\max \\ \tilde{\mathbf{S}}_k + K\left(\mathbf{O}_k - \mathbf{T}\tilde{\mathbf{S}}_k\right) & \text{others} \end{cases}. \tag{8}$$

In our experiment, we set $c_{k\max} = 26$.

Input: The state vector $\boldsymbol{S}_{k-1}$, image $\boldsymbol{img}_k$, max unmatched consecutive frames c_k_max.
Output: The state vector $\boldsymbol{S}_k$
1: estimate camera motion;
2: calculate the predicted state vector $\tilde{\boldsymbol{S}}_k$;
3: detect all pedestrian on $\boldsymbol{img}_k$;
4: predict the depth of $\boldsymbol{img}_k$;
5: match each track $\boldsymbol{track}_i$ with targets $\boldsymbol{target}_j$;
6: if $\boldsymbol{target}_j$ does not match any $\boldsymbol{track}_i$
go step 7;
elseif $\boldsymbol{track}_i$ does not match any $\boldsymbol{target}_j$
count number of unmatched frames c_k, and go step 8
else
go step 9
7: initialize a new KF tracker with $\boldsymbol{target}_j$;
8: if $c_k < c_k$_max
$\boldsymbol{S}_k = \tilde{\boldsymbol{S}}_k$else
delete $\boldsymbol{target}_j$
9: if $z_k^{(\tilde{S})} < z_k^{(O)}$
$\boldsymbol{S}_k = \tilde{\boldsymbol{S}}_k$else
$\boldsymbol{S}_k = \tilde{\boldsymbol{S}}_k + K(\boldsymbol{O}_k - \boldsymbol{T}\tilde{\boldsymbol{S}}_k)$
10: update KF

Algorithm 1: The overall algorithm for the proposed tracking method.

Table 1: Pedestrian detection results of different detectors on MOT17.

Detector	AP↑	MODA↑	MODP↑	FAF↓	TP↑	FP↓	FN↓	Hz↑
SDP [47]	**0.81**	**76.9**	**78.0**	**1.3**	**95699**	**7599**	**18865**	0.6
FRCNN [15]	0.72	68.5	78.0	1.7	88601	10081	25963	5.1
DPM [48]	0.61	31.2	75.8	7.1	78007	42308	36557	19.7
Fast YOLO [49]	0.77	73.1	77.8	1.5	92436	8694	20185	**147.2**

3.4. Camera Motion. Camera motion is an important factor to vision tracking, which not only changes the coordinates of the object but may also blur the image. The motion of dynamic objects is assumed to be small relative to the scene motion in most videos, so we use image alignment algorithm to approximate camera motion estimation. Philipp et al. proposed a practical work [42]. We first estimating a nonlinear pixel warp **W** between neighbouring frames which maps pixel coordinates (x_{t-1}, y_{t-1}) in one frame to the next (x_t, y_t) and then use this wrap to align boxes forecasted using frames up to $t-1$ with frame t.

At last, we summarize the algorithm proposed in this article in pseudo-code as follows (Algorithm 1).

4. Results and Discussion

We conducted tracking experiments on the popular multi-target tracking datasets, and the results will be shown in Section 4.3. In addition, we also discuss the effect of depth prediction in Section 4.2.

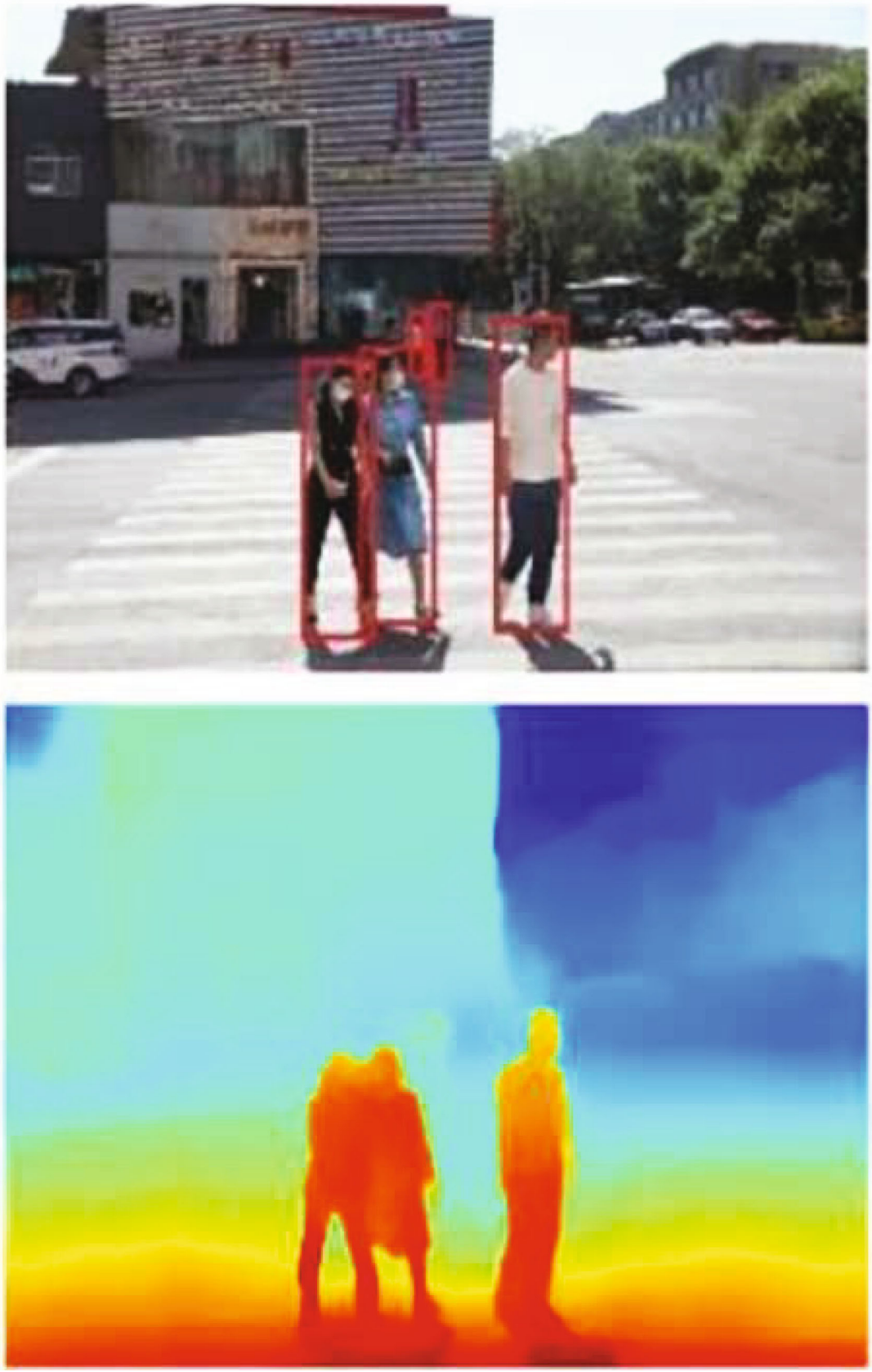
4.1. Experiment Setting. We evaluated our method on two popular MOT datasets: MOT2017 [43] and MOT2020 [44]. MOT17 contains 14 videos, 7 for training and 7 for testing. Faster R-CNN [10], SDP [45], and DPM [8] are provided as public target detector. MOT20 contains 8 videos, 4 for training and 4 for testing, and only Faster R-CNN [10] is provided. All of the datasets are very challenging including crowded scenes with heavy occlusions, camera motion, and both day and night sequences.

To evaluate the performance of the tracking methods, we adopted the widely used CLEAR MOT metrics [46]. Multiple object tracking accuracy (MOTA) evaluates accuracy in the presence of false positives (FP), false negatives (FN), and identity switches (IDS). IDS counts the total number of identity switches. At the same time, IDF1, MT, ML, and Hz are also considered. IDF1 is the ratio of correctly identified detections over the average number of ground-truth and computed detections, and it indicates the average maximum consistent tracking rate. MT evaluates the mostly tracked trajectories that are successfully tracked at least 80%. ML evaluates the mostly lost trajectories that are successfully tracked at most 20%. Hz indicates the processing speed (in frames per second). Among these metrics, MOTA and IDF1 are usually considered the most important.

For target detection, public detectors provided by the authors and private detector (Fast YOLO [14]) are all used for sufficient comparison. The private detector is trained

(a)

Figure 3: Continued.



(b)

Figure 3: Continued.

(c)

Figure 3: Continued.

(d)

Figure 3: Continued.

(e)

Figure 3: Result of pedestrian detection and depth prediction. The upper and under figures are the results of pedestrian detection and depth prediction, respectively.

Table 2: Tracking results using public detectors on MOT17.

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑
IOU [47]	45.5	39.4	15.7	40.5	**19993**	281643	5988	**1523.0**
SORT [15]	43.1	39.8	12.5	42.3	28398	287582	4852	143.3
GMPHD_Rd17 [48]	46.8	54.1	19.7	33.3	38452	257678	3865	30.8
FlowTracker [49]	40.4	38.0	14.0	36.6	60962	269136	5927	61.2
ours_pub	**51.3**	**55.2**	**20.8**	**32.5**	26236	**247437**	**3734**	40.4

Table 3: Tracking results using public detectors on MOT20.

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑
IOU_KMM [50]	46.5	49.4	**29.9**	19.6	57517	214777	4509	30.3
SORT [15]	42.4	45.1	16.7	26.2	27521	264694	4470	**57.3**
GMPHD_Rd20 [48]	44.7	43.5	23.6	22.1	42778	236116	7492	25.2
FlowTracker [49]	46.7	42.4	27.8	20.0	54732	217371	**3532**	19.2
ours_pub	**52.6**	**53.8**	29.3	**19.5**	**26891**	**214278**	3619	25.3

with the 7 training videos in MOT17. All models are retrained in advance and not updated when running. Our experiment was conduct in PyTorch and runs on a desktop with a CPU of Intel(R) Xeon(R) E5-2680@2.80GHz and a 1080Ti GPU.

4.2. Pedestrian Detection and Depth Prediction

4.2.1. Pedestrian Detection. To evaluate the effects of pedestrian detection, we tested Faster R-CNN [10], SDP [45], DPM [8], and Fast YOLO [14] on MOT17. Since there are many well-trained network models, we just fine-tuned them on the train set and then detect the pedestrians on the test set. Table 1 shows the result. The up arrow and down arrow, respectively, indicate that the larger the value, the better and the smaller the better, and the italic value indicates the best result. AP, MODA, MODP, and FAF mean average precision taken over a set of reference recall values, multiobject detection accuracy, multiobject detection precision, and the average number of false alarms per frame, respectively. As we can see, SDP has the best detection performance in most metrics; however, the speed is as slow as 0.6 FPS. Although Fast YOLO gets the second-best detection results, it achieves amazing speed, which is 245 times faster than SDP. There is no doubt that it is worthwhile to trade extremely small detection accuracy for extremely high detection speed for target tracking problem.

4.2.2. Depth Prediction. To evaluate the effects of depth prediction, we selected many pictures with different backgrounds for experiments, and five representative results are shown in Figure 3. It can be seen that pedestrians on all images are correctly detected. The effect of depth prediction is satisfactory, which can distinguish people from background. After careful observation, we found that depth prediction has the following two characteristics: One is that the estimation of the big target is more accurate. If the size of the target is small, the depth difference between the target and the background is smaller, and the target will not be well-marked on the depth map. As shown in Figure 3(d), the depth of the car on the right is similar to that of the person on the left, but the outline of the car is obviously clearer. The other is that the estimation is more accurate under a simple background. It can be seen that the depth distinction between pedestrians and background is obvious in Figures 3(a) and 3(b), while in the depth maps in Figures 3(c)–3(e), some people and background are difficult to distinguish. In Figure 3(a), it is easy to see the difference in depth between the pedestrians and the car, which reflects the advantage of using depth to judge the occlusion. In Figure 3(b), the background can be divided into three parts, the ground, the building on the left rear, and the trees on the right rear. Because the building and trees are far away from people, they have little influence on the depth prediction of the pedestrians, so depth image is also outstanding. As a comparison, we can be seen that the depth of the people nearby in Figure 3(c) is obvious, while the people faraway are difficult to distinguish from the background. Interestingly, in the depth map of Figure 3(d), the pedestrian in the middle seems to have blended into the background, while the rear car is vaguely discernible, which seems to be related to the contrast of pixels. In addition, the words on the image will also have a significant impact. In the depth map in Figure 3(e), the silhouette of human is not clear, probably because the interference from the ground is very serious. Although sometimes the result of depth estimation is not very accurate, it can help us judge occlusion as long as it can be distinguished from the background depth.

4.3. Pedestrian Tracking

4.3.1. Results on MOT17 and MOT20. Our goal is to build a real-time tracking system, so the algorithms that we select from MOT17 and MOT20 for comparison all reach a speed of 25 frames per second. Meanwhile, all these algorithms have been published in papers.

Table 2 shows the results on MOT17. IOU [47], SORT [15], GMPHD_Rd [48], and FlowTracker [49] are select as the baseline, because most of them also appear in MOT20,

Table 4: Tracking results of all real-time trackers on MOT17.

Tracker	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑	Det.
TrTrack [51]	**75.2**	63.5	**55.3**	**10.2**	50157	**86442**	3603	59.2	1
Fair [52]	73.7	72.3	43.2	17.3	27507	117477	3303	25.9	1
RekTCL [53]	73.3	**73.2**	41.3	18.7	22944	124980	**2790**	88.8	1
TraDeS [54]	69.1	63.9	36.4	21.5	20892	150060	3555	66.9	1
ours_pri	65.1	64.8	34.2	25.8	22065	171873	2976	52.6	1
ours_pub	51.3	55.2	20.8	32.5	26236	247437	3734	40.4	0
GMPHDOGM [55]	49.9	47.1	19.7	38.0	24024	255277	3125	30.7	0
GMPHD_Rd17 [48]	46.8	54.1	19.7	33.3	38452	257678	3865	30.8	0
PHD_LMP [56]	45.9	42.5	15.5	37.9	27946	272196	4977	29.4	0
IOU17 [47]	45.5	39.4	15.7	40.5	**19993**	281643	5988	**1522.9**	0
SORT [15]	43.1	39.8	12.5	42.3	28398	287582	4852	143.3	0
FlowTracker [49]	40.4	38.0	14.0	36.6	60962	269136	5927	61.2	0

and this is helpful for comparison. All trackers use the same public detectors (Faster R-CNN [10], SDP [45], and DPM [8]). We can see that our method achieves the best performance on MOTA, IDF1, MT, ML, FN, and IDS. IOU [47] gets the smallest FP score and amazing tracking speed, and our method gets the second smallest FP score and the third fastest speed. IDS scores largely reflect the ability of continuous tracking. Our method uses different strategies for different matching results, so we obtain better IDS score.

The results on MOT20 are shown in Table 3. Compared with MOT17, MOT20 contains less frames and less trajectories. Due to more crowdedness and more pedestrians, the detection task is much more challenging. On the whole, tracking is significantly slower, and MT and ML are better, but there is no clear trend in the other metrics. Our method gets the best scores in most metrics except for MT, IDS, and Hz.

Next, we compared our method and more published real-time trackers on MOT17. Ten trackers are selected, including 4 trackers using private detectors (TrTrack [51], Fair [52], RekTCL [53], TraDeS [54]) and 6 trackers using public detectors (GMPHDOGM [55], GMPHD_Rd [48], PHD_LMP [56], IOU17 [47], SORT [15], FlowTracker [49]). Table 4 shows the detail results. Benefit from better detectors and better training strategies, trackers using private detectors achieve high MOTA scores than trackers using public detectors. While using public detectors, ours_pub performs best. Although ours_pri ranks last in the trackers using private detectors, it has a relatively high speed. Because an additional network is introduced to calculate the depth, the speed of our method has to be encumbered. In the future, we plan to use light architectures to optimize the processing of depth prediction. Better detection and association methods are also considered to improve accuracy.

We demonstrate the relationship between tracker accuracy and speed in Figure 4 (excluding IOU17). The farther to the right, the higher the MOTA score, and the higher the upward, the faster the tracking speed. It can be clearly seen in the figure that the methods using private detectors have great advantages in tracking accuracy and not slow in speed. Our method is in the middle in speed and accuracy, and there is still much room for improvement.

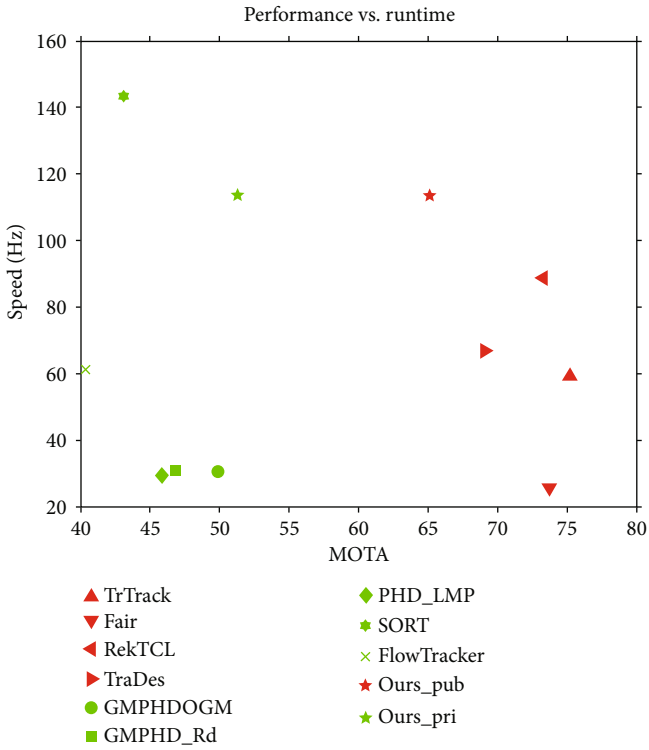


Figure 4: The performance compare between different trackers. Each marker denotes a tracker accuracy and speed. The red and green markers represent private and public detector, respectively.

4.3.2. Impact of Different Detectors. To explore the influence of different detectors on multiobject tracking, we ran our tracker with public and private detectors on MOT17. The results are shown in Table 5. Obviously, using better detector can improve the performance of the tracking algorithm, which is mainly reflected in three aspects: the total number of detected targets, the number of correctly recognized targets, and the detection precision. The MOTA scores from high to low are ours_pri, SDP, FrRCNN, and DPM, and it

Table 5: Tracking results of different detectors on MOT17.

Detector	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑
SDP	51.8	56.1	9.9	9.6	**2938**	68995	1096	43.2
FrRCNN	43.6	47.8	6.7	10.4	5725	84076	**956**	9.1
DPM	35.8	39.4	4.2	12.5	18573	94736	1682	48.4
ours_pri	**65.4**	**64.8**	**11.4**	**8.6**	7355	**57291**	992	**52.6**

Table 6: Tracking results of different depth predictors on MOT17.

Detector	MOTA↑	IDF1↑	MT↑	ML↓	FP↓	FN↓	IDS↓	Hz↑
HG	62.8	61.1	12.2	9.5	8573	64736	1082	17.1
Distilled-HG	**65.4**	**64.8**	**11.4**	**8.6**	**7355**	**57291**	**992**	**52.6**

directly reflects the performance of the detector. The private detector performs very well, especially for FN, which is 11704 lower than the best public detector. The tracking speed is not only related to the detection speed but also related to the number of targets detected, because the more the number of targets to be tracked, the greater the amount of calculation required to match the trajectory to the target. The private detector achieves 52.6 Hz, which is much higher than the public detectors.

4.3.3. Impact of Distillation. We compare the tracking performance with and without distilling the HG network, and the result is shown in Table 6. Accuracy of tracking improves as that of depth prediction improves after distillation. The most attractive change is speed. The tracking speed before distillation was only 17.1 Hz, and after distillation, it reached 52.6 Hz, increasing by 2.08 times.

4.3.4. Impact of Different Components. To get a deeper insight into our method, we run several experiments to test the effect of each component. The complete method with private detector is set as the baseline. We remove one component each time and then evaluate the method on MOT17. The result is shown in Table 7.

The function of KF is mainly reflected in two aspects: one is to predict the target search area, and the other is to smooth the trajectory. Since the detector is not updated online, its performance remains stable. But KF can affect the detection result by change the target search area. Therefore, the indicator related to accuracy (IDF1) has not changed much. When the occlusion or unmatched trajectory occurs, KF can continue to estimate the trajectory, effectively avoiding the interruption of the trajectory, so the impact on IDS is very obvious. Overall, MOTA drops by 6.2, mainly due to the increase in IDS. Since the computation of KF is very small, it has little influence on the tracking speed.

Occlusion detection involves depth prediction and occlusion judgment. Occlusion judgment will affect the processing method after IOU matching, so it has a greater impact on IDS but has almost no impact on IDF1. Since the depth prediction network is time-consuming, the tracking speed increases by 26.8 frame per second without occlusion detection, which is the largest increase amplitude in the three components.

Table 7: Result of ablation study on MOT17.

Components	MOTA↑	IDF1↑	IDS↓	Hz↑
Complete	65.1	64.8	2976	52.6
No KF	-6.2	-1.4	+127	+4.4
No occlusion detection	-4.5	-0.1	+54	+26.8
No motion estimation	-5.9	-1.1	+46	+11.3

The position of the target in the image is determined by the motion of the target and the camera. When using Kalman filtering, we assume that the camera is fixed. In fact, in the MOT17 dataset, three camera sequences are stationary, and four are captured from a moving camera. If we ignore the motion of the camera, it will inevitably affect the Kalman filter. It can be seen from Table 7 that motion estimation has an obvious impact on the overall performance. MOTA dropped by 5.9, IDF1 dropped by 1.1, IDS increased by 46, and tracking speed increased by 11.3.

Referring the results in Table 7 by column, we can get the impact of each component on a single indicator. Figure 5 shows this impact intuitively. KF, occlusion detection, and motion estimation all contribute to MOTA, IDF1, and IDS. For MOTA and IDF1, the effect of KF and motion estimation is slightly greater than that of occlusion detection, because KF and motion estimation have a greater impact on the target detection. For IDS, KF is particularly useful. This is due to the processing of missing values by KF, which results in a significant decrease in the number of reidentified targets. More components mean more computation, so the tracking speed inevitably decreases. Among them, occlusion detection has the greatest impact on the speed, because it calculates the depth through a deep neural network. Accuracy and speed are often difficult to compromise. Fortunately, our method improves accuracy while keeps high speed.

5. Discussion

In Section 4.2, we evaluate the performance of depth prediction. Experiment shows that depth prediction is more

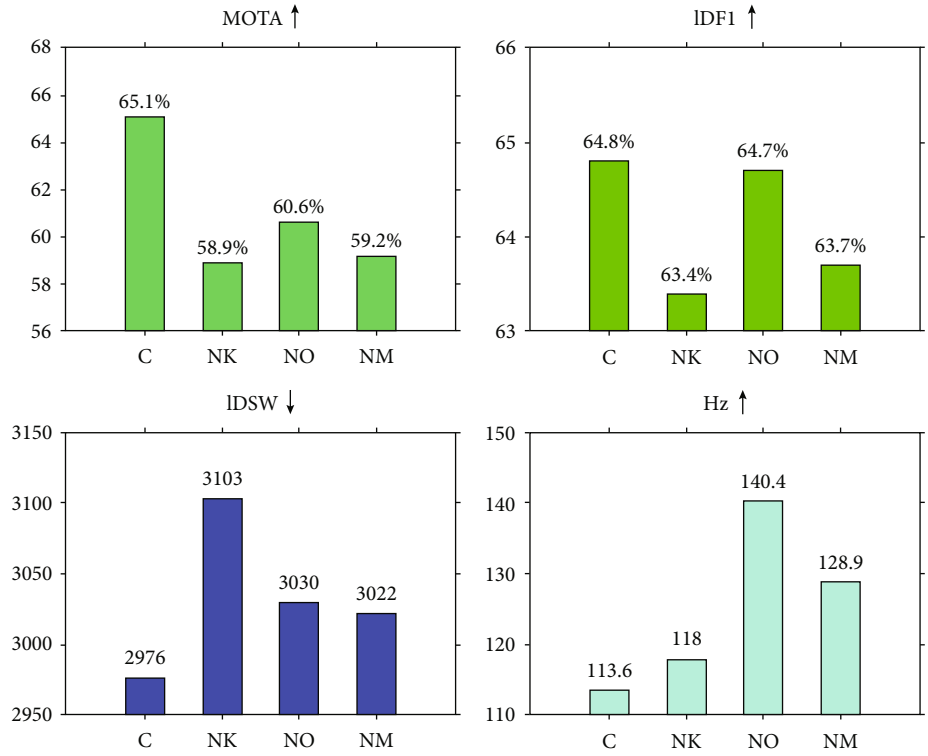


Figure 5: Impact of each component on indicators. C, NK, NO, and NM denote complete, no KF, no occlusion detection, and no motion estimation, respectively.

effective when the target is relatively large, or when it differs greatly from the background. Except for very few cases, the depth prediction is reliable. Although the estimated depth values cannot replace the true depth values, they completely reflect the sequential order of the objects, which is enough for us to judge the spatial location of the targets.

In Section 4.3, we evaluate our method with both public and private detectors on MOT17 and MOT20. Our method has achieved high accuracy while maintaining a high speed. It should be pointed out that the methods we selected are all real-time methods. In fact, more methods trade speed for accuracy. Through the ablation study, we show the impact of each component on the tracking results. KF framework has the greatest impact on IDS, as it is the key to maintaining continuous tracking. Occlusion detection plays the most important role in IDF1, because it determines the update method of KF, which in turn affects the final accuracy. For tracking speed, depth estimation involves neural network calculations and has the greatest impact on speed, while KF has relatively small impact. On the whole, each component has similar contributions to MOTA.

6. Conclusions

In this paper, we study pedestrian detection and tracking with a fixed monocular camera. To improve the robustness of tracking, we focus on continuous tracking and occlusion detection to deal with heavily occlusion. To keep continuous tracking, we introduce Kalman filter to estimate the motion state of the target in each frame. With the help of KF's predictive ability, we can continue to estimate the state of the target when the target is occluded. Once the target reappears, the tracker can quickly locate the target, thus avoiding reidentification and ensuring the continuity of the target's trace during the occlusion period. To detect occlusion, we introduce depth information in our tracking system. Once the depth of the target suddenly decreases a lot, we think the target is obscured by something, because the closer the distance to the camera the smaller the depth. Depth information is usually obtained through depth sensors or multiview images, which is not practical in daily life, because most cameras are monocular and cannot provide depth value. Here, we introduce a light depth prediction network, which is distilled from a large but well-performed network. Distillation can not only slightly improve the accuracy of depth prediction (2.6 MOTA score) but also greatly accelerate the speed of model inference (35.5 Hz). We evaluated our method on public object tracking dataset. The results show that our method can not only achieve high accuracy (65.1 MOTA score) on MOT 17 but also high tracking speed (52.6 Hz). In the future, we intend to introduce better association rules to improve the tracking accuracy, continuously optimize the target detection network and depth estimation network, and strive to achieve real-time tracking in a more universal environment.

Data Availability

The data used to support the findings of this study are included within the article. All datasets used in our research are publicly available and are cited in our article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Natural Science Foundation of Beijing Municipality (4212023), the Fundamental Research Funds for the Central Universities, USTB (FRF-GF-20-04A), the Major Program of National Social Science Foundation of China (No. 17ZDA331), the National Key Research and Development Program of China (2018YFC2001700), the Scientific and Technological Innovation Foundation of Shunde Graduate School, and the Engineering Research Center of Intelligence Perception and Autonomous Control, Ministry of Education, Beijing (100124).

References

[1] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, no. 3, pp. 362–386, 2020.

[2] S. James, P. Wohlhart, M. Kalakrishnan et al., "Sim-to-real via sim-to-sim: data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12627–12637, Long Bench, CA, USA, 2019.

[3] M. Gattullo, G. W. Scurati, M. Fiorentino, A. E. Uva, F. Ferrise, and M. Bordegoni, "Towards augmented reality manuals for industry 4.0: a methodology," *Robotics and Computer-Integrated Manufacturing*, vol. 56, pp. 276–286, 2019.

[4] G. Welch and G. Bishop, "An Introduction to the Kalman Filter," TR 95-041, Department of Computer Science, University of North Carolina at Chapel Hill, 1995.

[5] Z. Li and N. Snavely, "Megadepth: learning single-view depth prediction from internet photos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2041–2050, Salt Lake City, UT, USA, 2018.

[6] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition*, Kauai, HI, USA, 2001.

[7] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, pp. 886–893, San Diego, CA, USA, 2005.

[8] P. Felzenszwalb, D. McAllester, and D. Ramanan, "A discriminatively trained, multiscale, deformable part model," in *2008 IEEE conference on computer vision and pattern recognition*, pp. 1–8, Anchorage, AK, USA, 2008.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 580–587, Columbus, OH, USA, 2014.

[10] 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, 2017.

[11] 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 (CVPR)*, pp. 779–788, Las Vegas, NV, USA, 2016.

[12] W. Liu, D. Anguelov, D. Erhan et al., "Ssd: single shot multibox detector," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 21–37, Cham, Springer, 2016.

[13] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

[14] M. J. Shafiee, B. Chywl, F. Li, and A. Wong, "Fast YOLO: a fast you only look once system for real-time embedded object detection in video," 2017, https://arxiv.org/abs/1709.05943.

[15] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*, pp. 3464–3468, Phoenix, AZ, USA, 2016.

[16] V. K. Singh, B. Wu, and R. Nevatia, "Pedestrian tracking by associating tracklets using detection residuals," in *2008 IEEE workshop on motion and video computing*, pp. 1–8, Copper Mountain, CO, USA, 2008.

[17] H. Shen, L. Huang, C. Huang, and W. Xu, "Tracklet association tracker: an end-to-end learning-based association approach for multi-object tracking," 2018, https://arxiv.org/abs/1808.01562.

[18] H. Zhou, W. Ouyang, J. Cheng, X. Wang, and H. Li, "Deep continuous conditional random fields with asymmetric inter-object constraints for online multi-object tracking," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 4, pp. 1011–1022, 2019.

[19] A. Dehghan, S. Modiri Assari, and M. Shah, "Gmmcp tracker: globally optimal generalized maximum multi clique problem for multiple object tracking," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 4091–4099, Boston, MA, USA, 2015.

[20] S. Tang, B. Andres, M. Andriluka, and B. Schiele, "Subgraph decomposition for multi-target tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5033–5041, Boston, MA, USA, 2015.

[21] W. Feng, Z. Hu, W. Wu, J. Yan, and W. Ouyang, "Multi-object tracking with multiple cues and switcher-aware classification," 2019, https://arxiv.org/abs/1901.06129.

[22] P. Voigtlaender, M. Krause, A. Osep et al., "MOTS: multi-object tracking and segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7942–7951, Long Beach, CA, USA, 2019.

[23] C. Kim, F. Li, A. Ciptadi, and J. M. Rehg, "Multiple hypothesis tracking revisited," in *Proceedings of the IEEE international conference on computer vision (ICCV)*, pp. 4696–4704, Santiago, Chile, 2015.

[24] S. Tang, M. Andriluka, B. Andres, and B. Schiele, "Multiple people tracking by lifted multicut and person re-identification," in *Proceedings of the IEEE Conference on Computer*

Vision and Pattern Recognition (CVPR), pp. 3539–3548, Honolulu, HI, USA, 2017.

[25] Y. Xu, L. Qin, X. Liu, J. Xie, and S. C. Zhu, "A causal and-or graph model for visibility fluent reasoning in tracking interacting objects," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2178–2187, Salt Lake City, UT, USA, 2018.

[26] L. Chen, H. Ai, Z. Zhuang, and C. Shang, "Real-time multiple people tracking with deeply learned candidate selection and person re-identification," in *2018 IEEE international conference on multimedia and expo (ICME)*, pp. 1–6, San Diego, CA, USA, 2018.

[27] W. Zhang, H. Zhou, S. Sun, Z. Wang, J. Shi, and C. C. Loy, "Robust multi-modality multi-object tracking," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2365–2374, Seoul, Korea (South), 2019.

[28] S. Gautam, G. P. Meyer, C. Vallespi-Gonzalez, and B. C. Becker, "SDVTracker: real-time multi-sensor association and tracking for self-driving vehicles," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pp. 3012–3021, Montreal, Canada, 2021.

[29] F. Huan, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2002–2011, Salt Lake City, UT, USA, 2018.

[30] J. Jiao, Y. Cao, Y. Song, and R. Lau, "Look deeper into depth: monocular depth estimation with semantic booster and attention-driven loss," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 53–69, Munich, Germany, 2018.

[31] R. Chen, F. Mahmood, A. Yuille, and N. J. Durr, "Rethinking monocular depth estimation with adversarial training," 2018, https://arxiv.org/abs/1808.07528.

[32] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGBD images," in *Computer Vision – ECCV 2012. ECCV 2012*, A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, Eds., vol. 7576 of Lecture Notes in Computer Science, pp. 746–760, Springer, Berlin, Heidelberg, 2012.

[33] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: learning 3D scene structure from a single still image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824–840, 2009.

[34] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 3061–3070, Boston, MA, USA, 2015.

[35] J. Cao, Y. Pang, J. Xie, F. S. Khan, and L. Shao, "From handcrafted to deep features for pedestrian detection: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, p. 1, 2021.

[36] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: a survey," 2019, https://arxiv.org/abs/1905.05055.

[37] J. Ren, X. Chen, J. Liu et al., "Accurate single stage detector using recurrent rolling convolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5420–5428, Honolulu, HI, USA, 2017.

[38] W. Liu, S. Liao, W. Hu, X. Liang, and X. Chen, "Learning efficient single-stage pedestrian detectors by asymptotic localization fitting," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 618–634, Munich, Germany, 2018.

[39] S. Zhang, L. Wen, X. Bian, Z. Lei, and S. Z. Li, "Occlusion-aware R-CNN: detecting pedestrians in a crowd," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 637–653, Munich, Germany, 2018.

[40] Y. Tian, P. Luo, X. Wang, and X. Tang, "Deep learning strong parts for pedestrian detection," in *Proceedings of the IEEE international conference on computer vision (ICCV)*, pp. 1904–1912, Santiago, Chile, 2015.

[41] C. Zhou and J. Yuan, "Bi-box regression for pedestrian detection and occlusion estimation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 135–151, Munich, Germany, 2018.

[42] P. Bergmann, T. Meinhardt, and L. Leal-Taixe, "Tracking without bells and whistles," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICVV)*, pp. 941–951, Seoul, Korea (South), 2019.

[43] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "MOT16: a benchmark for multi-object tracking," 2016, https://arxiv.org/abs/1603.00831.

[44] P. Dendorfer, H. Rezatofighi, A. Milan et al., "Mot20: a benchmark for multi object tracking in crowded scenes," 2020, https://arxiv.org/abs/2003.09003.

[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 (CVPR)*, pp. 2129–2137, Las Vegas, NV, USA, 2016.

[46] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, Article ID 246309, 10 pages, 2008.

[47] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, Lecce, Italy, 2017.

[48] N. L. Baisa, "Occlusion-robust online multi-object visual tracking using a GM-PHD filter with CNN-based re-identification," *Journal of Visual Communication and Image Representation*, vol. 80, article 103279, 2021.

[49] H. Nishimura, S. Komorita, Y. Kawanishi, and H. Murase, "SDOF-Tracker: fast and accurate multiple human tracking by skipped-detection and optical-flow," 2021, https://arxiv.org/abs/2106.14259.

[50] O. Urbann, O. Bredtmann, M. Otten, J.-P. Richter, T. Bauer, and D. Zibriczky, "Online and real-time tracking in a surveillance scenario," 2021, https://arxiv.org/abs/2106.01153.

[51] P. Sun, Y. Jiang, R. Zhang et al., "Transtrack: multiple-object tracking with transformer," 2020, https://arxiv.org/abs/2012.15460.

[52] Y. Zhang, C. Wang, X. Wang, W. Zeng, and W. Liu, "Fairmot: on the fairness of detection and re-identification in multiple object tracking," *International Journal of Computer Vision*, vol. 129, no. 11, pp. 3069–3087, 2021.

[53] W. Li, Y. Xiong, S. Yang, M. Xu, Y. Wang, and W. Xia, "Semi-TCL: semi-supervised track contrastive representation learning," 2021, https://arxiv.org/abs/2107.02396.

[54] J. Wu, J. Cao, L. Song, Y. Wang, M. Yang, and J. Yuan, "Track to detect and segment: an online multi-object tracker," in

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 12352–12361, 2021.

[55] Y. Song, K. Yoon, Y. C. Yoon, K. C. Yow, and M. Jeon, "Online multi-object tracking with GMPHD filter and occlusion group management," *IEEE Access*, vol. 7, pp. 165103–165121, 2019.

[56] R. Sanchez-Matilla and A. Cavallaro, "Motion prediction for first-person vision multi-object tracking," in *Computer Vision – ECCV 2020 Workshops. ECCV 2020*, A. Bartoli and A. Fusiello, Eds., vol. 12538 of Lecture Notes in Computer Science, pp. 485–499, Springer, Cham, 2020.