









Fast Segmentation-Based Object Tracking Model for Autonomous Vehicles

Xiaoyun Dong^{1,2,3} , Jianwei Niu^{1,2,3,5} , Jiahe Cui^{1,2,3} , Zongkai Fu^{1,2,3} ,
and Zhenchao Ouyang^{1,2,4}  

¹ Hangzhou Innovation Institution, Beihang University, Chuanghui Street #18,
Binjiang, Hangzhou 310000, Zhejiang, China

ouyangkid@buaa.edu.cn

² State Key Laboratory of Virtual Reality Technology and Systems, Beijing, China

³ Beijing Advanced Innovation Center for Big Data and Brain Computing (BDBC)
Beihang University, Xueyuan Road #37, Haidian, Beijing 100191, China

⁴ Nanhu Laboratory, Jiaxin 314000, Zhejiang, China

⁵ Zhengzhou University Research Institute of Industrial Technology, Zhengzhou
University, Zhengzhou 450001, China

Abstract. On-road object tracking is a critical module for both Advanced Driving Assistant System (ADAS) and autonomous vehicles. Commonly, this function can be achieved through single vehicle sensors, such as a camera or LiDAR. Consider the low cost and wide application of optical cameras, a simple image segmentation-based on-road object tracking model is proposed. Different from the detection-based tracking with bounding box, our model improves tracking performance from the following three aspects: 1) the Positional Normalization (PONO) feature is used to enhance the target outline with common convolutional layers. 2) The inter-frame correlation of each target used for tracking relies on mask, this helps the model reducing the influences caused by the background around the targets. 3) By using a bidirectional LSTM module capable of capturing timing correlation information, the forward and reverse matching of the targets in consecutive frames is performed. We also evaluate the presented model on the KITTI MOTs (Multi-Object and Segmentation) task which collected from out door environment for autonomous vehicle. Results show that our model is three times faster than Track RCNN with slightly drop on sMOTSA, and is more suitable for deployment on vehicular low-power edge computing equipment.

Keywords: Object tracking · Segmentation · Bi-LSTM · Autonomous vehicle · Deep learning

1 Introduction

Multi-object tracking (MOT) in out door environment is a basic and challenging computer vision task [1–3]. It has a wide range of application scenarios and actual requirement, such as security monitoring, industrial production, precision

Supported by Hangzhou Innovation Institution, Beihang University.

© Springer Nature Switzerland AG 2020

M. Qiu (Ed.): ICA3PP 2020, LNCS 12453, pp. 259–273, 2020.

https://doi.org/10.1007/978-3-030-60239-0_18

guidance, aerial warning and transportation [4]. In the era of big data, related face recognition systems, satellites, surveillance cameras, vehicle-mounted systems, etc. provide rich data sources for researches [5] and also greatly promote the development of related technologies.

Due to the unclear definition of tracking tasks when dealing with partially visible, occluded, or cropped targets, and the lack of well defined and organized dataset and evaluation metrics, previous track systems lack comprehensive evaluation. Until the emergence of large-scale labeled datasets [6–8] in recent years, multi-target tracking algorithms and models have been able to develop rapidly [9]. However, most of the MOT are designed for data collected from fixed scene or with static sensors.

With the rapid development of sensor technology, big data processing and artificial intelligence, autonomous driving technology that can improve transportation efficiency, traffic safety, and travel convenience, has received widespread attention in academia and industry in recent years [10]. At the same time, complex road environments, limited vehicle platform and highly random obstacles also pose challenges to autonomous vehicle perception systems. The MOT is one of the most basic and critical perception modules for autonomous vehicle [11, 12].

The autonomous vehicle often equipped with rich sensors and can obtain information of the surrounding environment. Optical camera is the most commonly used sensors, and the vision-based MOT is widely studied [13]. Recent artificial intelligence solutions use deep learning modeling and big data-driven to build MOT model [14]. The convolutional neural network model relies on high-performance graphics cards, making the deployment of the model in a vehicle environment a challenge. Therefore, it is very necessary to study the tracking algorithm that balances performance and accuracy.

In this paper, a vision-based one stage MOT model is proposed for autonomous vehicle. Our tracking architecture is modified from Track R-CNN [3]. We first combine the Positional Normalization (PONO)[15] feature with a lightweight CNN backbone for refining object segmentation. The PONO combines first and second moments of features that can capture structural information. The bi-directional Long Short Term Memory (Bi-LSTM) module is then used to further enhance the association information of inter-frame targets. Finally, the segmentation masks of targets reduce the influence of the background, such as occluded targets and similar background. The final model can achieve better tracking performance while processing video stream data in a more efficient way. The contributions of this paper are summarized as follows¹:

- We present a light-weight segmentation-based multi-object tracking model for autonomous vehicular platform.
- The model combines PONO features and Bi-LSTM modules to capture the spatial-temporal features of targets from continues frames for better tracking.

¹ We provide code online at <https://github.com/XYunaaa/Fast-Segmentation-based-Object-Tracking-Model>.

- We evaluate the presented model on KITTI Multi-Object and Segmentation (MOTS) dataset.

The rest of this paper organized as follows. The related work of deep learning based tracking models are summarized in Sect. 2. We present the segmentation-based tracking model architecture in Sect. 3, and evaluation results on KITTI MOTs dataset in Sect. 4. The final Section concludes the current model and possible improvements.

2 Related Works

Most of current object tracking algorithms follow the tracking-by-detection fashion which detect the targets frame by frame, and combine with matching algorithms to assign track ID for the same objects. The whole process mainly consists of three steps: detection, feature extraction/motion prediction and association. The detection module first find all possible targets in each frame. And then, the second module extracts a feature set for each target based on detection results. Finally, the association module matches targets with similar features or performs motion estimation based on inter-frame information.

The deep learning-based model mainly borrowed from object detection task directly [16–18], because the convolutional neural network models can automatically extract robust features for detection task instead of hand-made-craft features or simple statistical features. And then, the detection results from continuous frames are sent to different trackers, such as correlation filter [19–21], Kalman filter [22–24], clustering algorithm [25, 26] or Hungarian algorithm [27], optical flow [28], etc., for ID matching.

As mentioned before, the deep learning models are mainly used for solve the detection process, both one-stage and two-stage models are used. The 2D bounding box (BBox) [22, 23, 29] is a common and efficient way to describe the results. In static scenes and top view applications, such as the security monitoring [30], face recognition, etc., the 2D BBox can capture the main outline of the targets. However, when dealing with occlusion targets and close-range targets, such as following a bus or waiting for traffic lights in a crowded traffic environment, the BBox may contain other objects in the background. This has a little effect on classification and detection tasks, but it will affect the characterization of the target, resulting in matching failure [3]. Segmentation-based tracking use masks for each target, and can significantly reduce the interference caused by the background. Moreover, splitting detection and feature extraction also affects the overall efficiency of the model.

The current solution avoids the usage of segmentation networks mainly for the following reasons. Segmentation models are usually very large, especially for the multi-channel results, and will affect processing efficiency. The segmentation-based annotation datasets are also very limited. Traditional features, such as HOG, SIFT, Daisy, etc., cannot generate from irregular masks.

To overcome the above problems, several studies perform end-to-end tracking, and use convolutional features for both detection and target encoding [3, 8, 31] to

speed up the model efficiency. Moreover, combining with segmentation model, the convolutional encoding of the target mask is also more accurate.

The Siamese network is another CNN-based tracking solution, which consists of two parallel branches for different purposes. Namely, a searching branch response for high-level semantic representation at global area, and a detecting branch for low-level fine-grained representation at local area. The later branch is often offline trained for the predefined targets, and shares the weights with its twin branch. Fully convolutional Siamese trackers can achieve real-time performance, but due to not fully exploit semantic and objective information, it is often less accurate than other state-of-the-art works. Subsequent researches modified the Siamese network from different aspects, such as constructing cascaded region proposals based on convolution features from different layers of Siamese network [32], combining with Graph Convolutional Network (GCN) [33], increasing network depth [34] and refining match at different stage [35].

It is also important to consider temporal information in tracking tasks, and the most simple way is to combine with LSTM module [36]. TrackR-CNN [3] also deployed LSTM in their model, but found that there is no essential improvement. Other works [30, 37] introduce the Bi-LSTM to explore target matching from both forward and reverse image sequences. Ning Wang et al. [38] even designed a novel unsupervised deep tracking framework to compute the consistency loss for network training from forward and backward image sequences.

Base on previous works, our model tries to optimize the data and detection results through PONO features and segmentation, and combine Bi-LSTM to capture temporal features.

3 Proposed Model

The architecture of the proposed model is presented in Fig. 1, the main workflow is borrowed from TrackR-CNN [3]. We also embed the whole detection, segmentation and target feature extraction and inter-frame matching into a whole CNN workflow. We first use a lightweight ResNet50 [39] backbone combining with the profile enhanced module of PONO to get the spatial association of the target from consecutive frames. Different lengths of subset sequences are also evaluated. A Bi-LSTM module that can extract temporal information is attached behind. This module learns the inter-frame relationship from frontward and backward, and each gate state exchanges information between the two parallel branches. The whole backbone combines spatio-temporal feature together for region proposal and generates mask, classification score and association vector for each target. All information is used for linking the inter-frame targets into tracks over time.

3.1 ResNet50 with PONO

To reduce total floating-point operations per second (Flops) of the model, we first use the ResNet50 as spatial-feature backbone. However, shallow networks

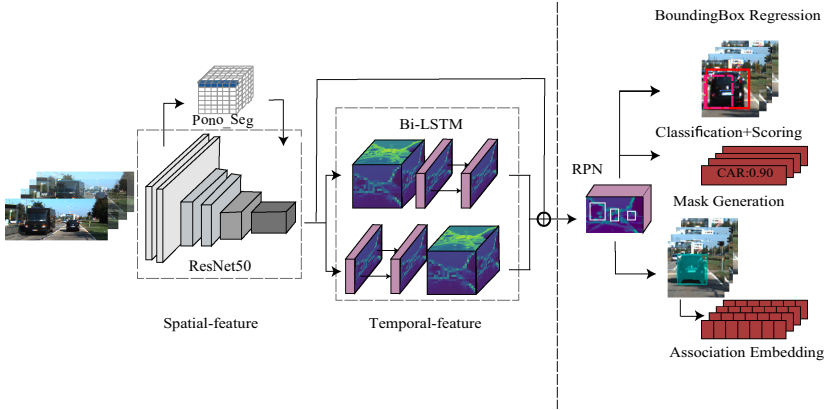


Fig. 1. The proposed segmentation-based on-road object tracking model.

will reduce the model fitting ability, especially on segmentation tasks. Therefore, we combine the Positional Normalization feature with ResNet50. Different from previous feature normalization (e.g., Batch, Group, Layer or Instance normalization), PONO not only benefit for network convergence, but also offer a clear structural information from the input images, as shown in Fig. 2(top). PONO achieves this by normalizing over the channels at any given pixel location, and the extracted statistics are position dependent and reveal structural information at this particular layer as Eq. 1.

$$\begin{aligned}
 u_{w,h,b} &= \frac{1}{C} \sum_{C=1}^{c=1} X_{w,h,c,b} \\
 \sigma_{w,h,b} &= \sqrt{\frac{1}{C} \sum_{C=1}^{c=1} (X_{w,h,c,b} - u_{w,h,b})^2 - \epsilon} \\
 PONO(X_{w,h,c}) &= \frac{X_{w,h,c} - u_{w,h,b}}{\sigma_{w,h,b}}
 \end{aligned} \tag{1}$$

Where $u_{w,h,b}$ and $\sigma_{w,h,b}$ are the mean and standard deviation or the first and second order extension, respectively. $\epsilon = 10^{-5}$ is a small constant coefficient. The w, h, c, b are the width, height, channel and batch of input feature map of current layer, and X is activation. Commonly, after the normalization operation is deployed on each layer, the two extracted statistics are discarded. PONO treats them as single channel feature maps with same scale of input, and combine them to convolutional feature together as final output. It can be seen that the added amount of calculation is very small, however, the outline information is very significant and can be used to assist segmentation. The ResNet50 can be treat as texture encoding, while the PONO can be treat as outline encoding. Figure 2 shows an example of how to combine the common convolutional layer and a 3-layer residual module with PONO layer together.

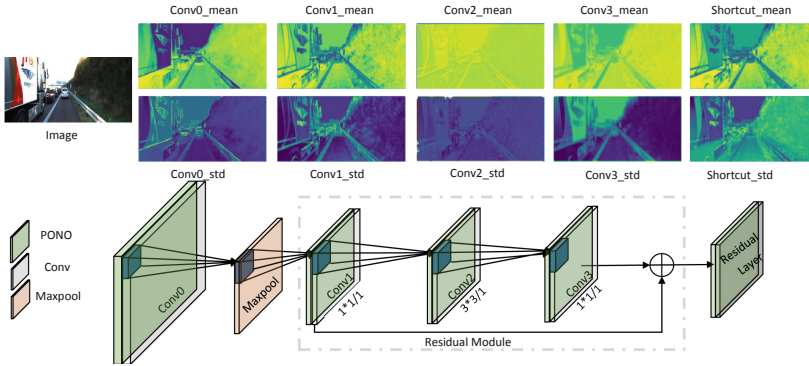


Fig. 2. Examples of the first (mean) and second (std) moments of the statistical PONO feature (top). The residual PONO module that combines residual convolutional and PONO layers (bottom).

3.2 Bi-LSTM

Building sequence model with RNN or LSTM module can capture temporal information, however, previous tracking tasks only considered forward order. However, if we treat the motion of vehicles and pedestrians as event, each frame is highly related with its context states (both forward and backward). Several human pose estimation and tracking tasks [37] use Bi-LSTM for encoding the temporal state of the target. Instead of using stacked LSTM, we use single Bi-LSTM unit to encode both forward and backward states of on-road objects as temporal-feature. We attached Bi-LSTM module behind the convolutional backbone, and to ensure the effectiveness of LSTM training, the nature order of frame sequence is used for training instead of random sampling.

3.3 Multi-task Region Proposal

As the previous two backbone modules generate spatio-temporal features, the final region proposal of our model directly borrow the multi-task network from TrackR-CNN [3, 30, 37]. It contains three individual branches, i.e., a mask-based segmentation, a classification and association vector generator. The fixed association vector is 128 generated from a fully connected layer from the mask, and is used for tracking with the Hungarian algorithm.

The whole model is trained on KITTI MOTs Challenge dataset². This dataset is extended from KITTI tracking dataset which contains 21 labeled sequences. Among them, the sequence {0, 1, 3, 4, 5, 9, 11, 12, 15, 17, 19, 20} for training and sequence {2, 6, 7, 8, 10, 13, 14, 16, 18} for validation. Each time we feed more than one frame as model input as a sliding window for training and validation.

² <https://www.vision.rwth-aachen.de/page/mots>.

4 Experiments

We first quantify the background noise introduced by Bounding box to the target detection process in an open road environment. Figure 3 shows two examples of the bounding box based detection and mask based detection, it is easy to see that the bounding box introduces background noises and overlapped vehicles.



Fig. 3. The bounding box vs. mask: bounding boxes involve invalid background.

Take the MOTs dataset of KITTI for example. We evaluate the pixel noise introduced by bounding boxes by comparing with mask labels in Table 1. We evaluate the pixels of mask and bounding box for cars and pedestrians according to Eq. 2, respectively. Where the cumulative pixel (CP) is calculated by adding all the $mask/bbox$ of class (car/ped) according to the tracking ID (t) in each sequence. The mask and bounding box donate incremental pixels (megapixel) of each category of targets in the 21 sequences. Seq 17 does not contain cars, and Seq 3, 5, 6, 8, 18 and 20 do not contain pedestrians.

$$\begin{aligned} CP_{car} &= \sum_{t \in T}^{t=1} f_{\{mask, bbox\}}^{c=car} \\ CP_{ped} &= \sum_{t \in T}^{t=1} f_{\{mask, bbox\}}^{c=ped} \end{aligned} \quad (2)$$

From Table 1, it is easy to see that the ratio of the mask/BBox for car ranges from 0.12 to 0.82, and the ratio of the mask/BBox for pedestrian ranges from 0.23 to 0.61. The average m/b for the 21 sequences is 0.28 for cars and 0.31 for pedestrians, respectively. The smaller the ratio, the greater the background noise introduced by BBox. And the pedestrians are affected more than cars, this may lead to poor tracking results for pedestrians when dealing with BBox. The overall ratio ranges from 0.13 to 0.65, and only 6 out of 21 sequences have a ratio larger than 50%.

We then train and evaluate our proposed model in the same way mentioned in Track R-CNN [3], the only difference is our hardware is a little different from theirs. All the testings are deployed on a GPU server with NVidia RTX2080ti@11.3 Gbps, with 32G memory and Intel Core i9-9900K CPU. During training, we can only set a maximum batch size of 7 due to the limitation of the GPU memory, the rest of the configuration is basically consistent with the original experiment [3].

The MOSTA (multi-object tracking and segmentation accuracy), MOTSP (mask-based multi-object tracking and segmentation precision) and sMOTSA

Table 1. Pixel compare: Bounding box vs. Mask (M=Million)

Seq	Cars			Pedestrian			Total		
	mask(M)	bbox(M)	m/b	mask(M)	bbox(M)	m/b	mask(M)	bbox(M)	m/b
0	2.15	2.89	0.75	0.06	0.10	0.56	10.08	25.01	0.40
1	21.00	28.71	0.73	0.21	0.39	0.54	30.43	63.15	0.48
2	2.14	2.91	0.74	0.17	0.33	0.52	7.41	24.09	0.31
3	2.69	3.32	0.81	-	-	-	3.56	6.07	0.59
4	2.57	3.62	0.71	0.12	0.22	0.55	6.31	16.99	0.37
5	3.07	3.90	0.79	-	-	-	5.69	12.88	0.44
6	3.15	4.15	0.76	-	-	-	5.86	11.55	0.51
7	28.56	36.55	0.78	0.09	0.19	0.49	38.51	59.06	0.65
8	2.48	3.20	0.77	-	-	-	4.54	8.86	0.51
9	20.54	27.65	0.74	0.03	0.06	0.57	33.01	67.40	0.49
10	1.71	2.28	0.75	0.05	0.07	0.61	7.00	12.88	0.54
11	16.45	22.06	0.75	0.10	0.20	0.51	19.99	3 5.88	0.56
12	0.13	0.16	0.78	0.01	0.02	0.53	0.61	1.51	0.41
13	0.20	0.24	0.82	2.14	4.12	0.52	14.01	36.30	0.39
14	2.60	10.46	0.25	0.21	0.56	0.38	3.74	29.52	0.13
15	12.28	46.73	0.26	1.97	8.43	0.23	17.76	130.93	0.14
16	2.53	21.11	0.12	6.33	23.93	0.26	13.48	94.41	0.14
17	-	-	-	4.40	13.80	0.32	10.92	67.46	0.16
18	12.04	42.21	0.29	-	-	-	13.75	65.25	0.21
19	11.81	38.42	0.31	16.73	54.19	0.31	77.66	352.53	0.22
20	26.35	96.36	0.27	-	-	-	51.41	229.78	0.22
All	50.19	176.99	0.28	16.73	54.19	0.31	142.81	647.55	0.22

(soft multi-object tracking and segmentation accuracy)[3] are introduced to evaluate the tracking model and is calculated as Eq. 3. Where TP is comprised of hypothesized masks which are mapped to a ground truth mask, and \widehat{TP} is the number of true positives. FP are hypothesized masks that are not mapped to any ground truth mask. FN are the ground truth masks which are not covered by any hypothesized mask. M denote the latest tracked predecessor of a ground truth (mask), or \emptyset if no tracked predecessor exists. The set IDS is defined as the set of ground truth masks whose predecessor was tracked with a different id. MOTSA is the mask-based multi-target tracking accuracy metric that only consider successfully tracked targets. MOTSP is the mask-based multi-object tracking and segmentation precision. sMOTSA measures both segmentation as well as detection and tracking quality.

$$\begin{aligned}
 MOTSA &= \frac{|TP| - |FP| - |IDS|}{|M|} \\
 MOTSP &= \frac{TP}{|TP|} \\
 sMOTSA &= \frac{\widehat{TP} - |FP| - |IDS|}{|M|}
 \end{aligned} \tag{3}$$

The batch size of 5 is used during all training and testing, as we found the enlarged batch size can slightly increase the model performance and has no influence on FPS during testing. When the batch size is greater than 5, the model performance will not continue to improve, as shown Table 2.

Table 2. Comparison of different batch sizes for training.

Batch		2	3	4	5	6	7
sMOTSA	Car	65	70.8	72.3	72.6	72.6	72.1
	Ped	29.7	31.9	34	34.2	34.1	34
FPS		10.29					

Table 3 illustrates the comparison of different model settings and model performances, two different backbones (Resnet50 and Resnet101) are used during testing, + means the backbone is integrated with PONO features in each module. The backbone is designed for charging the spatial feature, and is attached with a temporal module. Three different temporal modules are compared in our work, 3DCNN, LSTM and Bi-LSTM. In the end of the model, two tracking vector generation mechanisms are considered, i.e., generating from the RPN and from the mask.

We only evaluate the sMOTSA and FPS (Frame per Seconds) in the current stage. The \checkmark donates the selected combination of modules for each setting, + donates the PONO feature is used, and $\checkmark\checkmark$ means two repeatedly stacked modules. As we focus on tracking task, we randomly select a sequence during training and testing each time, and use the default frame order as model input.

Table 3. Comparison of different model settings.

	Backbone		Temporal Module			Vector		Performance		
	Resnet50	Resnet101	3DCNN	LSTM	Bi-LSTM	RPN	Mask	sMOTSA		FPS
								Car	Ped	
1		\checkmark	\checkmark			\checkmark		76.2	46.8	3.12
2	\checkmark		\checkmark			\checkmark		68.1	37.3	5.79
3	$\checkmark+$		\checkmark			\checkmark		70.7	30.2	10.29
4	\checkmark			\checkmark		\checkmark		61.4	39.3	8.82
5	\checkmark			$\checkmark\checkmark$		\checkmark		61.4	38.4	6.61
6	\checkmark				\checkmark 0.05	\checkmark		68.5	42.4	13.37
7	$\checkmark+$				\checkmark 0.05	\checkmark		72.3	34	10.36
8	$\checkmark+$				\checkmark 0.8	\checkmark		73.7	37.3	10.36
9	$\checkmark+$				\checkmark 0.8		\checkmark	74.9	43.6	10.15

The configuration1 is Track R-CNN [3] with a combination of Resnet101, 3DCNN, RPN-based tracking vector, it achieves the highest sMOTSA on both car and pedestrian, but can only reach 3.12 FPS on NVidia Titan XP@12G. This is far cry from the basic requirement of 10 Hz in autonomous system. By replacing the backbone with Resnet50, the FPS increases to 5.79 (nearly twice the original value), however, the sMOTSA drop to 68.1 and 37.3, respectively. Then, we add

PONO feature with Resnet50 backbone, the model FPS is surprisingly reached 10.29, and the car sMOTSA increased by about 2%, but pedestrian sMOTSA drops to 30.2. Which means that the PONO can heavily increase the backbone speed, and has benefit for large targets of cars.

Configuration 4, 5 and 6 show the performances of different temporal modules, except for the 3DCNN and LSTM, we also added a Bi-LSTM to capture the patterns from both toward and backward. Compare to configuration2, LSTM module shows higher processing efficiency for extracting temporal features than 3DCNN, but the sMOTSA is much lower than 3DCNN. The stack of two LSTM shows no better than single LSTM module, and increased processing time from 8.82 FPS to 6.61 FPS. Moreover, the Bi-LSTM shows similar performance on car, but much higher sMOTSA on pedestrian (42.4), and configuration6 can achieve the highest FPS at 13.37 among all configurations. This also donates that convolution-based Bi-LSTM can confirm the timing characteristics of the target through reverse information and greatly improve the efficiency of feature extraction and later tracking.

We further adjust the weight of the temporal module, as this is not a main module, we only consider two different conditions, i.e., 0.05 and 0.8, for Bi-LSTM. Considering the configuration6 and 7, when using Resnet50 with PONO, the sMOTSA of car further improves to 72.3 while the sMOTSA of pedestrian drops to 34. We then enlarged the weight of temporal module to 0.8 as in configuration8, both the sMOTSA of the car and pedestrian have increased significantly, and this process has not affect on FPS. In configuration9, the tracking vector is generated from mask instead of RPN, the tracking performance of sMOTSA is raises to 74.9 for car and 43.6 for pedestrian, and the FPS only drops 0.21. This is very close to Track R-CNN with a deeper backbone of Resnet101, but the FPS has increased more than three times, and can meet the needs of autonomous driving systems. Table 4 lists the detailed tracking indicators of the above-mentioned models with different configurations on cars and pedestrians.

Table 4. Details of tracking performances.

	Cars			Pedestrians		
	sMOTSA	MOTSA	MOTSP	sMOTSA	MOTSA	MOTSP
1	76.2	87.8	87.2	46.8	65.1	75.7
2	68.1	79.8	86.3	37.3	56.3	73
3	70.7	82.1	87	30.2	52.9	66.5
4	64.1	76	86.4	39.3	57.8	73.2
5	61.4	73.6	86.2	38.4	57.2	73.2
6	68.5	80.2	87	42.4	62.2	73.2
7	72.3	83.2	87.5	34	56.4	67
8	73.7	84.9	87.4	37.3	59.5	68.2
9	74.9	85.6	87.7	43.6	63.4	73.4

Figure 4 illustrates an example of ID matching based on the mask-based tracking vectors, the upper figure donates segmentation results at time T , while the bottom figure donates segmentation results at time $T+1$. After segmentation, we use the tracking vectors in each frame for matching track IDs with the famous Hungary algorithm. Figure 4(a) is the segmentation results in frame T and $T+1$. While Fig. 4(b), 4(c), 4(d), 4(e) and 4(f) donate the matching process for each targets. Due to the van and bus are ignored classes in KITTI tracking dataset, the model does not tracking on them during both training and testing. We also illustrated the calculated distances between the current target vectors (in $T+1$ frame) and last target vectors (in T frame), each target uses the same color of number and BBox. Our model successfully tracked four cars and a pedestrian, i.e., Car34, Ped83, Car44, Car52 and Car53, between the two frames.

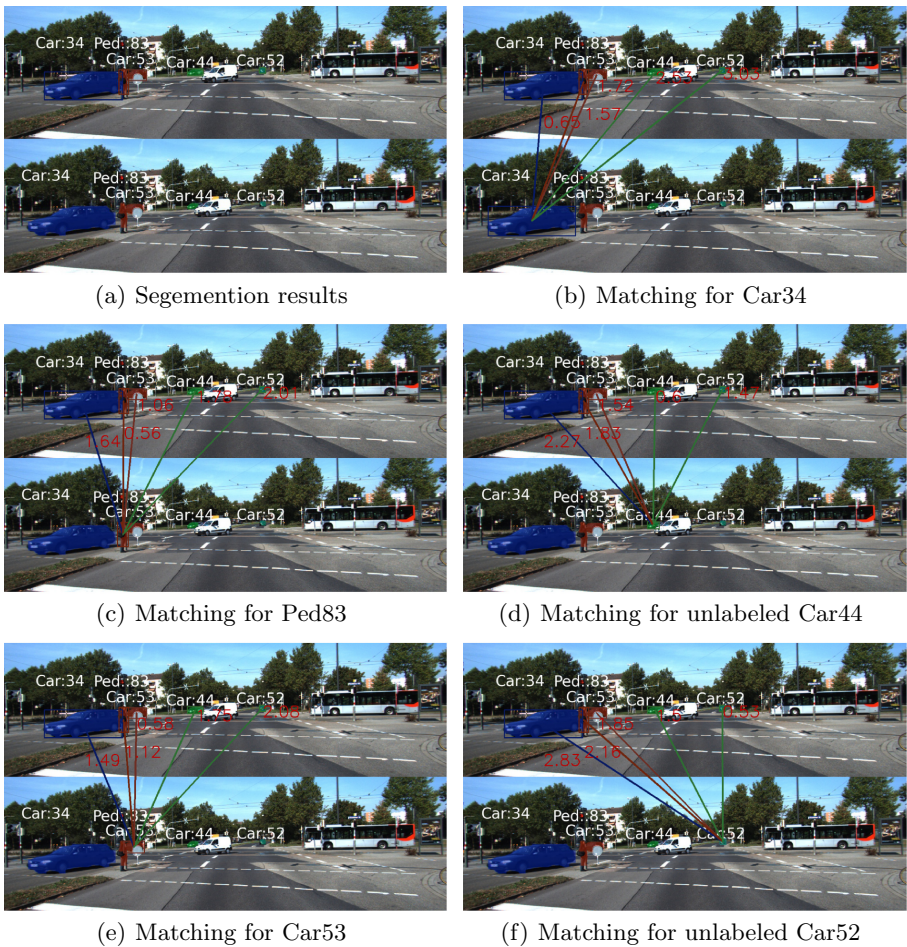


Fig. 4. Evaluation of inter-frame ID matching.

This process can also be presented as matching matrix with distances information as shown in Fig. 5. When the element on the main diagonal is the smallest, and the distance of the same target (track ID) in the adjacent frames has the shortest distance, the model can achieve the best matching result based on the tracking vector distances. The vector distance between all correctly matched target frames is less than 1 (around 0.5).

Figure 6 illustrates four examples of 2D tracking results of our final model, the gray trajectory lines are ground truth calculated according to BBox center, while the trajectory line consistent with the same color of the mask is the predicted result of our model. It is easy to see that the predicted trajectories of the final model are very close to the ground truth in ideal occasions.



Fig. 5. The matching distance matrix of inter-frame ID matching.



(a) Scenario 1



(b) Scenario 2

Fig. 6. Examples of 2D tracking results of the final model.

5 Conclusion

The cars and pedestrians are two main moving targets in open street environment. Accurate tracking of dynamic targets is an important module for perception system of autonomous vehicles, and can provide powerful information support for subsequent driving decision making and motion planning. To address accurate and efficient dynamic target tracking in dynamic open street area, this paper presented a novel segmentation-based object tracking model. We use light-weight backbone of Resnet50 with PONO module for generating the spatial features, and then combined with a Bi-LSTM for generating temporal feature from continuous inter-frame information in forward and reverse orders. The final tracking vector generated from segmentation mask also helps increase the tracking performance. Testing results on the public KITTI MOTs dataset show that the tracking performance of our model is slightly lower than Resnet101 based Track R-CNN on MOTSA, MOTSP and sMOTSA, but is three times faster and can be deployed for autonomous vehicles.

The current model still has some drawbacks, such as the tracking is relied on segmentation results, when the segmentation fails or is wrong, the entire system will fail. Moreover, the current tracking results on pedestrian is also not remarkable and safe enough for usage. We plan to solve those problems with re-identification strategies in the future.

Acknowledgment. This work has been supported by National Natural Science Foundation of China (61772060, 61976012), Qianjiang Postdoctoral Foundation (2020-Y4-A-001), and CERNET Innovation Project (NGII20170315).

References

1. Porzi, L., Hofinger, M., Ruiz, I., Serrat, J., Bulò, S.R., Kotschieder, P.: Learning multi-object tracking and segmentation from automatic annotations. arXiv preprint [arXiv:1912.02096](https://arxiv.org/abs/1912.02096) (2019)
2. Osep, A., Voigtlaender, P., Weber, M., Luiten, J., Leibe, B.: 4D generic video object proposals. arXiv preprint [arXiv:1901.09260](https://arxiv.org/abs/1901.09260) (2019)
3. Voigtlaender, P., et al.: MotS: multi-object tracking and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7942–7951 (2019)
4. Li, X., Weiming, H., Shen, C., Zhang, Z., Dick, A., Van Den Hengel, A.: A survey of appearance models in visual object tracking. *ACM Trans. Intell. Syst. Technol. (TIST)* **4**(4), 1–48 (2013)
5. Ward, J.S., Barker, A.: Undefined by data: a survey of big data definitions. arXiv preprint [arXiv:1309.5821](https://arxiv.org/abs/1309.5821) (2013)
6. Leal-Taixé, L., Milan, A., Reid, I., Roth, S., Schindler, K.: Motchallenge 2015: towards a benchmark for multi-target tracking. arXiv preprint [arXiv:1504.01942](https://arxiv.org/abs/1504.01942) (2015)
7. Milan, A., Leal-Taixé, L., Reid, I., Roth, S., Schindler, K.: Mot16: a benchmark for multi-object tracking. arXiv preprint [arXiv:1603.00831](https://arxiv.org/abs/1603.00831), 2016

8. Chen, Y., Jing, L., Vahdani, E., Zhang, L., He, M., Tian, Y.: Multi-camera vehicle tracking and re-identification on AI city challenge 2019. In: Proceedings of CVPR Workshops (2019)
9. Milan, A., Schindler, K., Roth, S.: Challenges of ground truth evaluation of multi-target tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 735–742 (2013)
10. Shi, W., Alawieh, M.B., Li, X., Yu, H.: Algorithm and hardware implementation for visual perception system in autonomous vehicle: a survey. *Integr.* **59**, 148–156 (2017)
11. Leal-Taixé, L., Milan, A., Schindler, K., Cremers, D., Reid, I., Roth, S.: Tracking the trackers: an analysis of the state of the art in multiple object tracking. arXiv preprint [arXiv:1704.02781](https://arxiv.org/abs/1704.02781) (2017)
12. Ouyang, Z., Niu, J., Liu, Y., Guizani, M.: Deep CNN-based real-time traffic light detector for self-driving vehicles. *IEEE Trans. Mob. Comput.* **19**(2), 300–313 (2019)
13. Luo, W., et al.: Multiple object tracking: a literature review. arXiv preprint [arXiv:1409.7618](https://arxiv.org/abs/1409.7618) (2014)
14. Ciaparrone, G., Sánchez, F.L., Tabik, S., Troiano, L., Tagliaferri, R., Herrera, F.: Deep learning in video multi-object tracking: a survey. *Neurocomputing* **381**, 61–88 (2020)
15. Li, B., Wu, F., Weinberger, K.Q., Belongie, S.: Positional normalization. In: Advances in Neural Information Processing Systems, pp. 1620–1632 (2019)
16. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
17. Redmon, J., Farhadi, A.: Yolov3: an incremental improvement. arXiv preprint [arXiv:1804.02767](https://arxiv.org/abs/1804.02767) (2018)
18. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-ResNet and the impact of residual connections on learning. In: 31st AAAI Conference on Artificial Intelligence (2017)
19. Wang, L., Xu, L., Kim, M.Y., Rigazico, L., Yang, M.H.: Online multiple object tracking via flow and convolutional features. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3630–3634. IEEE (2017)
20. Wang, Q., Zhang, L., Bertinetto, L., Hu, W., Torr, P.H.: Fast online object tracking and segmentation: a unifying approach. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1328–1338 (2019)
21. Zhao, D., Hao, F., Xiao, L., Tao, W., Dai, B.: Multi-object tracking with correlation filter for autonomous vehicle. *Sensors* **18**(7), 2004 (2018)
22. Bewley, A., Ge, Z., Ott, L., Ramos, F., Upcroft, B.: Simple online and realtime tracking. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3464–3468. IEEE (2016)
23. Yu, F., Li, W., Li, Q., Liu, Y., Shi, X., Yan, J.: POI: multiple object tracking with high performance detection and appearance feature. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9914, pp. 36–42. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-48881-3_3
24. Chen, J., Sheng, H., Zhang, Y., Xiong, Z.: Enhancing detection model for multiple hypothesis tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 18–27 (2017)
25. Tan, X., et al.: Multi-camera vehicle tracking and re-identification based on visual and spatial-temporal features. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 275–284 (2019)

26. Zhang, S., et al.: Tracking persons-of-interest via adaptive discriminative features. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9909, pp. 415–433. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46454-1_26
27. Kieritz, H., Hubner, W., Arens, M.: Joint detection and online multi-object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1459–1467 (2018)
28. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: Deepflow: large displacement optical flow with deep matching. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1385–1392 (2013)
29. Danelljan, M., Bhat, G., Khan, F.S., Felsberg, M.: Atom: accurate tracking by overlap maximization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4660–4669 (2019)
30. Maksai, A., Fua, P.: Eliminating exposure bias and metric mismatch in multiple object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4639–4648 (2019)
31. Li, X., Ma, C., Wu, B., He, Z., Yang, M.H.: Target-aware deep tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1369–1378 (2019)
32. Fan, H., Ling, H.: Siamese cascaded region proposal networks for real-time visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 7952–7961 (2019)
33. Gao, J., Zhang, T., Xu, C.: Graph convolutional tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4649–4659 (2019)
34. Li, B., Wu, W., Wang, Q., Zhang, F., Xing, J., Yan, J.: Siamrpn++: evolution of siamese visual tracking with very deep networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4282–4291 (2019)
35. Wang, G., Luo, C., Xiong, Z., Zeng, W.: SPM-tracker: series-parallel matching for real-time visual object tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3643–3652 (2019)
36. Milan, A., Rezatofighi, S.H., Dick, A., Reid, I., Schindler, K.: Online multi-target tracking using recurrent neural networks. In: 31st AAAI Conference on Artificial Intelligence (2017)
37. Kim, C., Li, F., Rehg, J.M.: Multi-object tracking with neural gating using bilinear LSTM. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 200–215 (2018)
38. Wang, N., Song, Y., Ma, C., Zhou, W., Liu, W., Li, H.: Unsupervised deep tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1308–1317 (2019)
39. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)