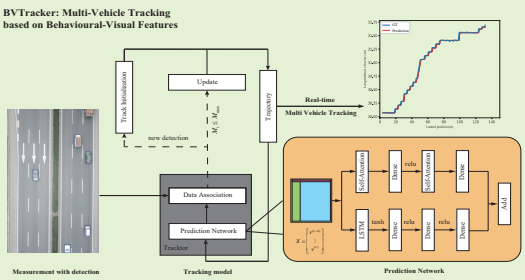
BVTracker: Multi-Vehicle Tracking based on Behavioural-Visual Features

Mingxu Li, Donghai Zhai, *Member, IEEE*, Da Yang, Lihua Xu

***Abstract*— In recent years, Unmanned Aerial Vehicle (UAV) is widely used in vehicle tracking. However, the objects in drone pictures always consist of fewer pixels due to their flying height. The lacking of visual information results in unreliable tracking results. Meanwhile, higher flying heights capture more targets, which makes it difficult to perform real-time inference with existing methods. To solve the above problems, we present a lightweight tracking model which employed behavioural information in the multi-vehicle tracking problem. Besides, BVTracker utilizes visual information as well as behavioural information independently and consists of two branches. One is a trajectory prediction model based on Long Short-Term Memory (LSTM) network and self-attention network. The other one is an association model base on the Hungarian Algorithm. To enable multi-vehicle tracking, the trajectory of each vehicle is predicted by the trajectory prediction model and the prediction results are associated with the detection findings. Moreover, a new dataset collected by UAVs for vehicles on freeways is applied to train and validate BVTracker. Compared with the state-of-art tracking algorithms, the experimental results show that the proposed method is superior to the existing algorithms at different frame rates and can achieve both robustness and real-time requirements, which profits the fast and effective traffic data analysis.**



***Index Terms*— multi-vehicle tracking, trajectory extraction, trajectory prediction, UAV**

I. Introduction

MULTI-VEHICLE TRACKING (MVT), which aims at identifying and tracking vehicles in image sequences, provides a reliable data source like velocity, attitude and direction for applications such as traffic flow analysis and security monitoring.

At present, many studies on the tracking problem have been conducted [1]–[7]. The idea behind the existing tracking algorithms is to find out and utilize more effective features to improve the detection and re-ID processes. However, due to the fact that from a bird's-eye view, the targets are similar to each other and lack sufficient visual information, the visual-based method needs a more complex model to deal with the confusing information. Besides, since the features between road and vehicles are totally different while the target is quite similar to other vehicles, sharing the features between the object detection and re-ID tasks may not perform well. To meet the need of accuracy and low computation cost from the blooming traffic industry, a feasible idea is to introduce other information and realise tracking by detection. For a tracking-by-detection paradigm, the MVT problem is broken down into three parts: 1) localize the targets in the current frame; 2) predict the position based on the past information; 3) combine the predictions and detections together. The detection and re-ID tasks then can utilize the information independently.

Therefore, the object tracking model proposed in this paper is a hybrid model structure in which one is a trajectory prediction branch based on the LSTM network and self-attention network, while the other one is an association model base on the Hungarian Algorithm. The detection findings and the trajectory predictions model are associated by the Hungarian algorithm. To train and evaluate the proposed model, we collected a new multi-vehicle tracking dataset on freeways from a bird's-eye view. Compared with the state-of-the-art MVT models, our method significantly reduces the number of ID switches. Furthermore, BVTracker updates with an average frame per second (FPS) rate of 157.7, which is much higher than other trackers.

The main contributions of this work are as follows:

- We propose a new model for MVT problem that efficiently tracks vehicles from the bird's eye view of a stationary UAV. Our model minimizes ID switches and increases algorithm computation speed and accuracy.
- We utilize both behavioral information and environmental constraints to optimize tracking performance. By incor-

(Corresponding authors: Donghai Zhai)

Mingxu Li is with the School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu China (e-mail: Li.Mingxu@my.swjtu.edu.cn).

Donghai Zhai is with School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, Sichuan, 610031, China (e-mail: dhzhai@swjtu.edu.cn).

Da Yang is with School of Transportation and Logistics, Southwest Jiaotong University, Chengdu, Sichuan, 610031, China (e-mail: yangd8@swjtu.edu.cn)

Lihua Xu is with School of Physical Science and Technology, Southwest Jiaotong University, Chengdu, Sichuan, 610031, China (e-mail: xulihua@home.swjtu.edu.cn)

porating these factors into our model, we enhance the accuracy and efficiency of our MVT system.
- We have collected a comprehensive dataset for MVT problem that covers various shooting altitudes and illumination conditions. This dataset is formatted in MOT16, facilitating compatibility with a range of tracking algorithms.

The paper is structured as follows: Section II reviews the related work on tracking methods. Section III presents our tracking model. In Section IV, we describe a new dataset and compare the outputs with different models. Finally, Section V makes conclusions.

II. Literature Review

With the emergence of excellent object detection algorithms [8]–[11], tracking-by-detection methods have been a feasible solution to MVT problems. Alex Bewley et al. proposed a simple online and real-time tracking (SORT) model [12]. In this model, the predictive coordinates were obtained by Kalman filtering, which was based on the target's historical motion state. Based on the SORT framework, Simple online and real-time tracking with a deep association metric (DEEP-SORT) [4] was proposed. This algorithm introduces visual information into the SORT framework, and uses visual features as measurement indicators for data association. Such work also included [2], [3], [6], [13]. For this series of methods, first, the position of the tracked vehicle is extracted by the detector based on Convolutional Neural Networks (CNN) and the prediction is calculated by the predictor. Then the detection results and prediction results are fed into the association method. Finally, tracks are formed by associating detections and predictions across adjacent frames.

Unlike traditional surveillance equipment that captures road sections discontinuously [14]–[19], UAV surveillance is able to cover continuous roads and provides more comprehensive data. Therefore, analysis of traffic conditions from a bird's-eye view has attracted the attention of many researchers recently. Shi et al. [20] proposed a novel vehicle trajectory extraction method that can extract high-granularity vehicle trajectories from aerial videos. Besides, they put forward a vehicle trajectory dataset named High-Granularity Highway Simulation (HIGH-SIM). Compared with the Next Generation Simulation dataset (NGSIM) [21], the HIGH-SIM dataset is more reasonable in speed and acceleration distributions. Barmpounakis et al. [22] created an urban traffic dataset to study congestion. In this dataset, traffic streams in a congested area of 1.3 km^2 are recorded by 10 drones, which helps researchers to develop and test their own models. Fukun et al. [23] proposed a context-based tracking method for remote sensing target tracking of UAV aerial videos to improve the tracking performance. The Response-Adaptive Context-Aware Correlation Filter (RA-CACF) model is also introduced into the network architecture to improve tracking performance. Ke et al. [24] presented an optical flow-based framework as a detector and an ensemble classifier as a tracker to match identified vehicles in adjacent frames. The model fills a gap in traffic flow analysis and has produced encouraging results, but the tracking result is very sensitive to the detection result. Once a vehicle is lost in a few frames, wrong trajectories may be generated. Though UAVs are widely used in traffic analysis, as shown in [25]–[28], due to the challenges of tracking and detection from UAV view, vision-based tracking algorithms tend to underperform. Therefore, we propose a lightweight and accurate multi-vehicle tracking model to meet the growing need of the traffic industry.

III. Methodology

In this section, we introduce BVTracker including the trajectory prediction network, the association algorithm and the collected dataset.

A. Model Structure

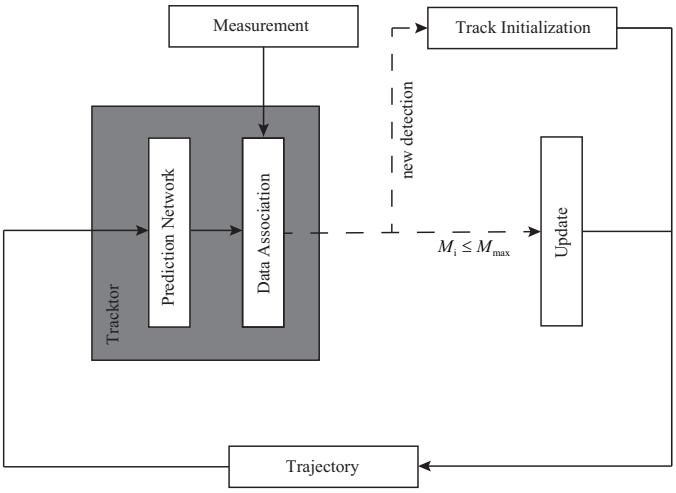


Fig. 1. Overview of BVTracker. The input image is first fed to a detection model based on CNN to extract the position of each vehicle. Then we put the measurement into the tracking model and associate the measurement with existing trajectories. Finally, a new trajectory will be generated or the existing trajectories will be updated.

The structure of BVTracker is displayed in Fig. 1. The bounding boxes of the vehicles shot from a drone are obtained by the detector. Then, the data association method is carried out to match the detected vehicles and the predicted trajectory. For each trajectory, we count the frame number since the last prediction result is successfully matched with the detection result, which is presented as M_{i}, where i is the trajectory's id. Meanwhile, to distinguish whether the vehicle has left the screen or is shortly lost, we define a threshold $M_{\max}$. Once M_{i} exceeds a predefined value $M_{\max}$, we stop to predict its position in the next frame. A new trajectory will be generated when a new detection is found. When a successful matching happens or a new trajectory is created, the detection result will be updated to its historical trajectory. After that, the historical trajectory is fed into our prediction network. The outputs of the prediction network are associated with the detection results in the next frame and realizing the online multi-vehicle tracking.

B. Trajectory prediction model

The prediction model aims at outputting the position of the target in the next frame based on travelling trace and environmental constraints. Fig. 2 briefly demonstrates the two major components of our prediction model, and $[x_{t-h}, ..., x_t]$ is the input in a particular time window.

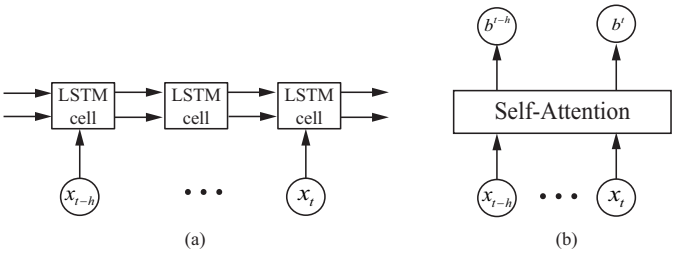


Fig. 2. The overview of the LSTM and self-attention. (a). Overview of LSTM layer structure. (b). Overview of the self-attention layer structure.

Since the LSTM network has made many successes in sequence prediction problems, we adopt an LSTM [29] network to analyze trajectory information in the video sequence. To achieve better robustness, the LSTM cell is composed of a more sophisticated structure compared with the vanilla Recurrent Neural Network (RNN) network which is shown in Fig. 3 (a). For a new input x_t, previous state m_{t-1}, and previous output h_{t-1}, the LSTM cell combines them together by a "gate" mechanism.

1) Forget gate: this gate decides how much information the internal state needs to forget. The forget gate utilizes the new input x_t and the previous state m_{t-1}, and output a vector f_t between (0,1):

$$f_t = \text{sigmoid}(W_f \bullet [h_{t-1}, x_t] + b_f) \tag{1}$$

where W_f and b_f is the parameter of linear transformation and need to be learned during the training process.

2) Input gate: this gate decides what needs to be stored in the internal state m_{t-1}. Firstly, the input gate calculates a vector i_t between (0,1) to decide what information needs to be added into m_{t-1}. Next, the input gate generates a candidate vector $\tilde{m}_t$. The i_t and $\tilde{m}_t$ are denote as:

$$i_t = \text{sigmoid}(W_i \bullet [h_{t-1}, x_{t-1}] + b_i) \tag{2}$$
$$\tilde{m}_t = \tanh(W_m \bullet [h_{t-1}, x_{t-1}] + b_m) \tag{3}$$

Finally, the input gate calculates the new internal state m_t:

$$m_t = f_t * m_{t-1} + i_t * \tilde{m}_t \tag{4}$$

3) Output gate: this gate calculates the output based on m_t, x_t and h_{t-1} and generates a vector o_t to decide what information in m_t needs to be output. o_t is shown as :

$$o_t = \text{sigmoid}(W_o \bullet [h_{t-1}, x_t] + b_o) \tag{5}$$

Next, the LSTM cell scales the m_t and generates the output h_t:

$$h_t = o_t * \tanh(m_t) \tag{6}$$

Due to the special structure, the LSTM network can elegantly handle time series prediction.

Besides, compared with the LSTM network, the self-attention model is better at figuring out the interaction among targets. Thus, BVTracker uses the self-attention network to deal with what effect the surrounding vehicles will have on the motion of the target. As shown in Fig. 3 (b), we demonstrate an example of the self-attention layer. For a input vector $[x_{t-h}, ..., x_t]$, we take linear transformations and generate three different vector q^{t-h}, k^{t-h}, v^{t-h}. Then, vector q^{t-h} multiply $k^{t-h}, ..., k^t$ separately and get $[a_{t-h,t-h}, ..., a_{t-h,t}]$. The vector $[a_{t-h,t-h}, ..., a_{t-h,t}]$ is processed with softmax function to the normalized vector, and the result denotes as $[\hat{a}_{t-h,t-h}, \cdots, \hat{a}_{t-h,t}]$. Finally, the output denotes as :

$$b^{t-h} = \sum_{i=t-h}^{t} \hat{a}_{t-h,i} \times v^{t-h} \tag{7}$$

For each vehicle, we use a matrix X to record their information which is denoted as:

$$X = [v^{(t-h)}, ..., v^{(t-1)}, v^{(t)}] \tag{8}$$

where v is the target feature in each frame which can be represented as:

$$\begin{aligned} v^{(t)} = [&x_{targ}^{(t)}, y_{targ}^{(t)}, \Delta x_f^{(t)}, \Delta y_f^{(t)}, \Delta x_b^{(t)}, \\ &\Delta y_b^{(t)}, \Delta x_l^{(t)}, \Delta y_l^{(t)}, \Delta x_r^{(t)}, \Delta y_r^{(t)}, \Delta x_{fl}^{(t)}, \Delta y_{fl}^{(t)}, \\ &\Delta x_{fr}^{(t)}, \Delta y_{fr}^{(t)}, \Delta x_{bl}^{(t)}, \Delta y_{bl}^{(t)}, \Delta x_{br}^{(t)}, \Delta y_{br}^{(t)}] \end{aligned} \tag{9}$$

where (x_{targ}, y_{targ}) denote the coordinates of the target and Δx and Δy respectively denote the longitudinal and lateral distances between the target and specific vehicle.

As exhibited in Fig. 4, the relative distances of the eight surrounding vehicles are chosen as constraints to replicate the information that humans rely on in driving. To represent the coordinates for the target in the next frame, the output of the prediction network is defined as follows,

$$\left(\hat{x}_{targ}^{(t+1)}, \hat{y}_{targ}^{(t+1)}\right) \tag{10}$$

Where $\hat{x}_{targ}^{(t+1)}$ and $\hat{y}_{targ}^{(t+1)}$ are the prediction of the target vehicle vertically and horizontally. The network structure is exhibited in Fig. 5. We divide the prediction model into two parts: prediction based on target motion and bias based on environmental factors.

As represented in Fig. 5, the input sequence shown in Eq. 8 is split and feed into the two branches. One is a matrix that indicates the historical trajectory of the target vehicle, and the other represents the state of the current frame. The reference LSTM network uses the first layer of the LSTM layer with 256 neurons, followed by two dense layers with 256 and 128 cells respectively, and the final dense layer contains the same size with the output. In BVTracker, the role of the LSTM layer is to capture the motion feature of the target during the time window. The other component of the reference architecture is a hierarchical self-attention network. The self-attention block is shown as Fig. 3 (b). Firstly, we reshape a single input

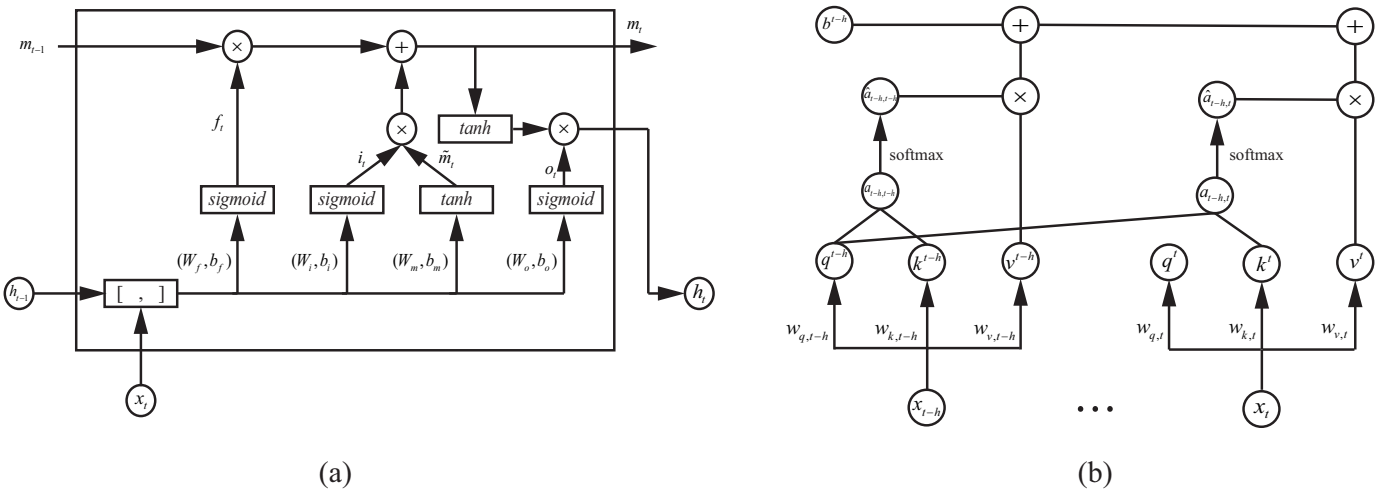


Fig. 3. (a). The detail of LSTM cell. *sigmoid* and *tanh* represent a sigmoid and tanh activation function respectively and [,] indicates concatenate $\boldsymbol{x_t}$ and $\boldsymbol{h_t}$ together. (b). An illustration of the work process of the self-attention layer.

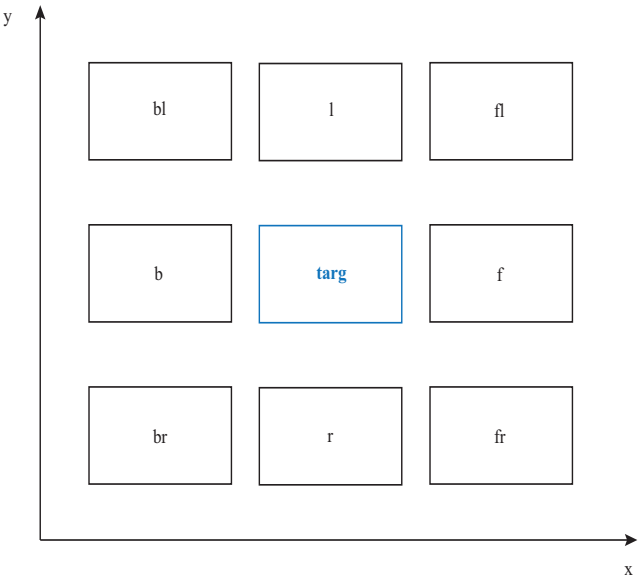


Fig. 4. The relative position between vehicles. The left and right vehicles of the target are denoted as $\boldsymbol{l}$ $(\boldsymbol{y_l} > \boldsymbol{y_{targ}})$, $\boldsymbol{r}$ $(\boldsymbol{y_r} < \boldsymbol{y_{targ}})$. The vehicles preceding $\boldsymbol{l}$, $\boldsymbol{targ}$, and $\boldsymbol{r}$ are denoted as $\boldsymbol{fl}$, $\boldsymbol{f}$, and $\boldsymbol{fr}$. Similarly, the vehicles at the back of the target are denoted as $\boldsymbol{bl}$, $\boldsymbol{b}$, and $\boldsymbol{br}$.

vector into a sequence format like $[seq\_len, feature\_len]$. The sequence includes information on the current situation and the environmental constraint.

C. Data Association

Based on the detection and trajectory prediction results, the problem of multi-vehicle tracking is transformed into a frame matching problem which is solved optimally using the Hungarian algorithm [4], [12]. The objective function of the Hungarian algorithm is as follows,

$$\min z = \sum_{q} \sum_{p} C_{qp} x_{qp} \tag{11}$$

where x_{qp} denotes whether the q_{th} detection frame is assigned to match the p_{th} prediction and equals 1 or 0, and C_{qp} denotes the cost between q_{th} detection and the p_{th} prediction.

In BVTracker, we use the Intersection over Union (IoU) [30] to measure the cost between prediction which is denoted as,

$$C_{ij} = -\frac{Area(prediction) \cap Area(detection)}{Area(prediction) \cup Area(detection)} \tag{12}$$

Additionally, we found that IoU distance is able to handle the occlusion problem implicitly. Once an occlusion happens, only the occluder is captured by the detector. Since the target length-width ratio is stable from a bird's-eye view, and the target tracking box always keeps a similar scale, the occluder bounding box will not be matched with the target vehicle.

IV. Experiments

To evaluate the effectiveness of BVTracker, we compare the proposed model with several state-of-the-art algorithms. Both designed experiments and the dataset are described in this section. All experiments were conducted on Ubuntu 18 platform. The computer has an Intel(R) Core(TM) i7-10700 CPU and NVIDIA GeForce RTX 3060 GPU with a memory of 32 GB.

A. Dataset

Although new challenges are brought to tracking tasks by UAVs, limited datasets have been published for vehicle tracking. Due to the movement of the camera, the published datasets don't provide the trajectory information. To validate BVTracker, we collect a new dataset for the MVT problem from a bird's-eye view. Based on the different situations on the freeway, the dataset is captured at a fixed position over a freeway and includes scenes varying from day, evening, 50 m and 100 m. Specifically, this dataset covers different scenes like the interference from street lights and shadows, which poses a challenge for MVT tasks. The video is taken at 30 fps with a resolution of 1920*1080. About 47000 frames are extracted from the videos and finely annotated

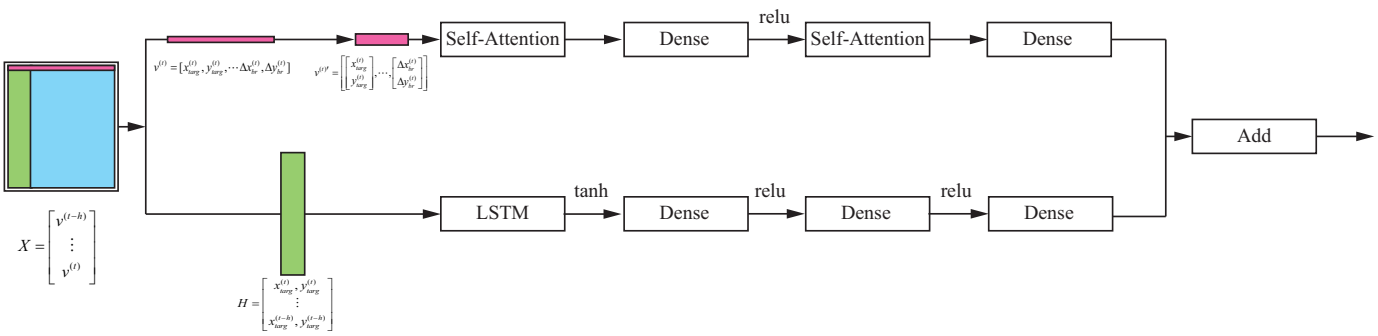


Fig. 5. Overview of our prediction model. The matrix Eq. 8 is split into two slices. One matrix is presented as $\boldsymbol{H}$ and denotes the motion of the target vehicle. The other slice is named $\boldsymbol{v^{(t)}}$ and denotes the current state. The two slices are fed into different branches of our prediction model. Finally, the two branches are added together and generate the final output.

Fig. 6. Sample of annotated frame in our dataset.

with bounding boxes by the vatic tool [31]. Finally, about 2.3 million bounding boxes are labelled, as well as 2740 different vehicles are marked. We select about 30% of the dataset collected on both day and night as the evaluation data, and the rest of the dataset as the training data. The whole dataset is saved in the MOT16 [32] format. Fig. 6 shows some sample frames in our dataset.

B. Implementation Details

We use the standard tracking metrics [33] to evaluate models:

- Multi-object tracking accuracy (MOTA): a measure that combines overall tracking accuracy including identity switches, false positives as well as false negatives which is defined by

$$MOTA = 1 - \frac{\sum_t (fn_t + fp_t + mme_t)}{\sum_t gt} \tag{13}$$

where fn_t is the number of false negative, t is the frame number, fp_t is the number of false positive, mme_t is the number of ID switches and gt is the number of ground-truth.

- Multi-object tracking precision (MOTP): a measure that combines overall tracking precision including bounding boxes overlap between ground-truth and reported positions, which is defined by

$$MOTP = \frac{\sum_{k,t} d_t^k}{\sum_t c_t} \tag{14}$$

where k is the identity of each match, c_t is the number of successful matches between detections and predictions at frame t, d_t^k is the distance between each detection-prediction pair.

- Mostly tracked objects (MT): a ratio of ground-truth trajectories which have the same label with their hypothesis at least 80% in their life span.
- Mostly lost objects (ML): a ratio of ground-truth trajectories which have the same label with their hypothesis at most 20% in their life span.
- Identity switches (IDs): the number of identity switches.
- IDF1 Score (IDF1): a ratio of correctly identified targets over the ground truth and the average number of calculated targets. IDF1 is given by:

$$IDF1 = \frac{2 \times IDTP}{2 \times IDTP + IDFP + IDFN} \tag{15}$$

where $IDTP$ is the number of true positive identities and $IDFP$, $IDFN$ are the number of false positive, the number of false negative respectively.

In our evaluation, to avoid the influence of different units, the pixel distance extracted from videos is converted into the Système International d'Unités (SI) units. In addition, YOLO v4 proposed by Bochkovskiy et al. [8] is used as the detector and trained by the collected dataset. The outputs of the detector with confidence larger than 0.6 are selected. The comparison algorithms are trained in the same experimental setting as ours.

In Subsection III-A, $M_{\max}$, the maximum number that is allowed to skip for unmatched tracks is set to 2. To more realistically model the decision-making during driving, we train the network using inputs with a time window of 30 frames, representing the observed data within the past 1 s. We utilize the Adam optimizer [34] with a starting learning rate of 10^{-4} to update network parameters. The proposed model is implemented by PyTorch. Furthermore, the batch size is set to 128 and Root Mean Square Error (RMSE) is used as the loss function.

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N}\|P_{pre} - P_{gt}\|^2} \tag{16}$$

where P_{pre} is the predicted value (x_{pre}, y_{pre}), P_{gt} is the ground truth (x_{gt}, y_{gt}) and N is the number of predictions in each batch.

C. Ablation Studies

To demonstrate the impact of different factors of the model, we have experimented with the following variations of the model:

1) BVTracker.
2) Only uses target car information.
3) Only LSTM network.
4) Adding a third self-attention layer with 256 neurons.
5) Only one self-attention block.
6) Using Kalman Filtering to replace deep learning model.

The results are summarized in Table I. By comparing whether we use information from environmental targets, it can be found that the experimental results are consistent with our expectations. In experiments, compared to using only the target vehicle or only the LSTM branch, the proposed model reduced ID switches by 94% (1835) and 99% (12856), respectively. BVTracker had better performance on the main indicators but had only slight differences on other indicators, as its model with or without self-attention layers was similar to the proposed model. On the other hand, the Kalman filter-based method had similar ID switch results to the proposed method, but due to its inability to handle environmental constraint information, it had a significant difference in MOTA compared to the proposed method.

For scenarios such as lane changing and acceleration, the model considering environmental factors can track the target more effectively. Fig. 7 demonstrates a comparison between whether the model uses environmental constraints. As shown in the upper sequence, since the model uses only previously observed data, the predicted response delay when the vehicle state changed, which causes an ID switch. Fig. 8 shows the performance of the two models mentioned above when a vehicle accelerates. As seen in Fig. 8, the incorrect tracking of two vehicles happens at the same time in the upper sequence. The missing environmental constraints make it hard for the model to distinguish them. While BVTracker keeps the target's ID stable.

D. Comparing with SOTA methods

To verify the effectiveness of the proposed method, we compare BVTracker with several state-of-the-art algorithms.

- DEEPSORT [4]: DEEPSORT is based on the two-step MOT method, which takes visual information into consideration and a cascade matching mechanism to improve the accuracy of inter-frame association.
- FairMOT [35]: FairMOT combines the detection task and the tracking task together. This model implements tracking based on Re-ID features.
- MPNTracker [36]: This framework uses the message-passing mechanism in graph convolutional networks directly. And tracking by using features based on Re-ID as well as position data. MPNTracker has been ranked at the top of various MOT Challenge lists [37].
- UniTrack [36]: UniTrack is a unified framework to handle tracking problems. This model contains a task-agnostic appearance model and proposes a novel reconstruction-based similarity metric for association tasks.

The results are given in Table II. MPNTracker and UniTracker are powerful trackers which perform well on FN, MT and ML indicators but bring much more complexity and lead to a drop in FPS. Besides, as can be seen from the comparison between the results of UniTrack(resnet18) and UniTrack(resnet50), a deeper appearance model helps little on IDs and FP indicators but lead to a significantly slower updates rate. FairMOT and DEEPSORT get remarkable

TABLE I
COMPARISON OF THE DIFFERENT MODELS

Model	MOTA ↑	MOTP ↑	FP ↓	FN ↓	MT ↑	ML ↓	IDs ↓
BVTracker	**76.0**	**76.0**	48700	39552	1321	41	**108**
Only use target car information	75.0	75.9	48392	39559	1321	41	1943
Only LSTM network	59.5	75.5	**40099**	92500	685	202	12964
Adding a third self-attention layer with 256 neurons	75.4	75.9	48836	39427	**1322**	**40**	110
Only one self-attention block	75.4	75.9	48903	**39417**	**1322**	60	111
Using Kalman Filtering to replace deep learning model	75.3	75.9	48372	40231	1315	42	120

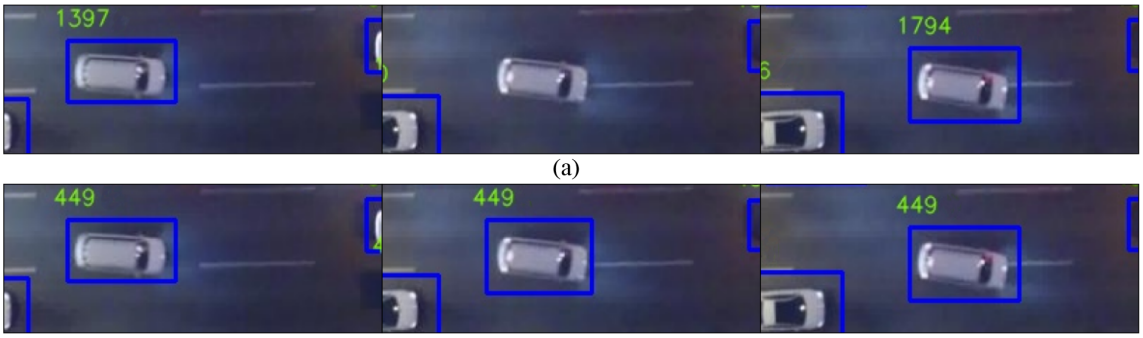


Fig. 7. Comparison of BVTracker in the lane change scenario with a model that does not use environmental vehicle information. (a) Model that without environmental information. (b) Our model.

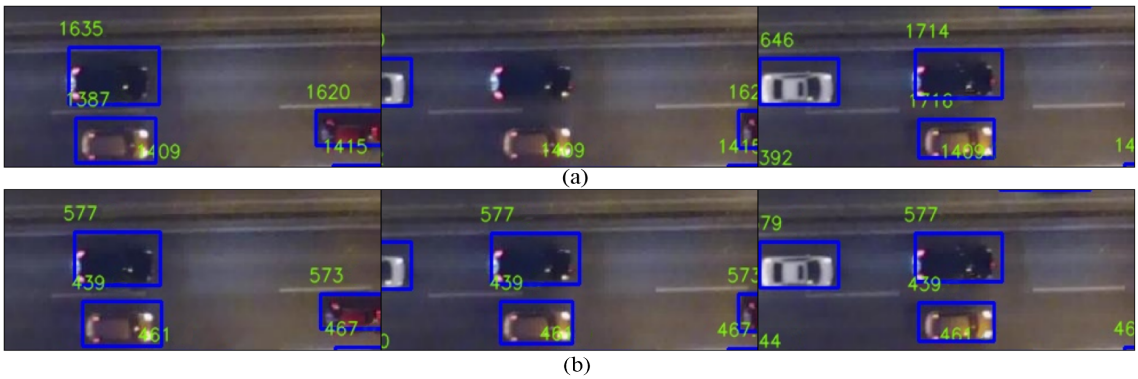


Fig. 8. Acceleration and vehicles run parallel scenarios in which BVTracker is compared with a model that does not apply information about the surrounding vehicles. (a) Model that without environmental information. (b) Our model.

TABLE II
TRACKING RESULTS ON THE PROPOSED DATASET

model	MOTA ↑	MOTP ↑	FP ↓	FN ↓	MT ↑	ML ↓	IDs ↓	IDF1 ↑	FPS ↑
BVTracker	**76.0**	**76.0**	**48700**	39552	1321 (92.3%)	41 (2.9%)	**108**	**86.4**	**157.7**
FairMOT	52.5	71.8	89079	81113	1153 (80.6%)	96 (6.7%)	501	74.8	15.4
MPNTracker	74.3	**76.0**	53546	**38681**	**1332 (93.1%)**	43 (3.0%)	321	83.5	38.7
DEEPSORT	72.9	75.5	54138	43156	1313 (91.8%)	40 (2.8%)	184	84.5	39.6
UniTrack(resnet18)	74.1	75.5	51818	40764	1321 (92.4%)	**38 (2.7%)**	689	85.9	6.4
UniTrack(resnet50)	74.1	75.5	51784	40784	1319 (92.2%)	**38 (2.7%)**	676	85.9	2.9

results on public datasets but get poor performances on the MVT problem due to their results depending more on visual information. Since BVTracker utilizes visual and behavioural information independently, the proposed model gets the best performance on the leading indicators. It further decreases the number of ID switches reduces from 501, 312, 184, 689 and 676 to 108 and runs at the highest frame rate.

Besides, as demonstrated in Fig. 9, with the decrement of the frame rate, the advantages of BVTracker become more obvious. Compared with the state-of-the-art algorithms, our method achieves much better results in ID-related indicators, while making a slight sacrifice in some metrics like MOTA when the frame rate comes to 6. This is mainly due to the fact that BVTracker takes the influence of surrounding vehicles into account. It is noteworthy that the DEEPSORT algorithm also shows a significant drop in the IDs metric when the frame rate is lowered. However, from the MOTA figure, the comprehensive index of the DEEPSORT algorithm has a more dramatic drop. This means that the algorithm is only tracking a very small number of vehicles at low frame rates, hence the drop in the IDs metric. The UniTrack(resnet18) and UniTrack(resnet18) perform almost the same on different indexes. It means that a more complex appearance model is not helpful to the robustness and accuracy in the MVT problem. Notably, our paper uses the YOLOv4 algorithm, which has been officially released and is widely used in multi-object tracking algorithms. Thus the impact from the detection phase is stable.

TABLE III
TRACKING RESULTS IN DIFFERENT CONDITIONS

Condition	model	MT ↑	ML ↓	IDs ↓	MOTA ↑	MOTP ↑	IDF1 ↑
Daytime	BVTracker	719	18	**12**	**87.8**	76.0	**94.0**
	DEEPSORT	715	21	15	81.7	74.5	91.0
	FairMOT	664	42	39	84.3	**76.1**	91.9
	MPNTracker	**722**	20	156	83.7	**76.1**	88.6
	UniTrack(resnet18)	721	**17**	440	86.0	74.6	92.4
	UniTrack(resnet50)	719	**17**	426	85.9	74.7	92.4
Evening	BVTracker	599	24	**96**	**73.2**	**76.0**	**84.6**
	DEEPSORT	598	**19**	169	70.8	75.7	83.0
	FairMOT	489	54	462	45.0	70.5	70.9
	MPNTracker	610	23	165	72.0	**76.0**	82.3
	UniTrack(resnet18)	600	21	255	71.3	75.8	84.4
	UniTrack(resnet50)	**600**	21	250	71.3	75.8	84.4
50 m	BVTracker	699	14	**9**	**92.6**	86.9	**96.0**
	DEEPSORT	697	15	10	90.1	86.2	94.8
	FairMOT	595	40	429	79.5	73.7	86.3
	MPNTracker	**703**	15	54	91.1	**87.1**	94.1
	UniTrack(resnet18)	**703**	**13**	236	91.6	86.2	95.3
	UniTrack(resnet50)	700	**13**	225	91.6	86.2	95.3
100 m	BVTracker	619	28	**99**	**68.3**	69.9	**81.8**
	DEEPSORT	616	**25**	174	64.8	69.6	79.7
	FairMOT	558	56	72	39.9	**70.7**	69.5
	MPNTracker	**629**	28	267	66.4	70.0	78.4
	UniTrack(resnet18)	618	**25**	459	65.9	69.6	81.4
	UniTrack(resnet50)	619	**25**	451	65.8	69.6	81.4

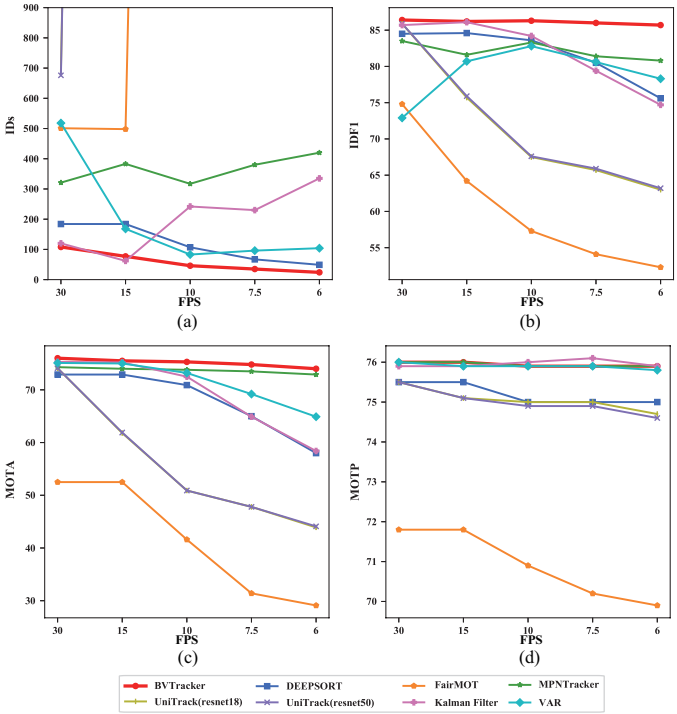


Fig. 9. Tracking results with different frame rates. (a), (b), (c) and (d) represent IDs, IDF1, MOTA, MOTP respectively.

To validate the experimental outcomes under various shooting settings, our experiments cover different scenes including days, evenings, 50 m and 100 m. The different illumination condition in the day and evening affects the visual features and videos taken in the evening introduce shadows which cause uneven illumination. The shooting height affects the scale variation of objects. When shooting at 50 m, targets with more details are captured and occupy a larger area of the frame. When a video is taken at a higher altitude, the objects are of less clarity. The results of the comparison are shown in Table III. BVTracker has clear advantages in ID-related indices. With the condition changing, our method always keeps the least number of ID switches and reaches the highest IDF1 score. With the shooting conditions becoming worse (Daytime to Evening, 50 m to 100 m), the BVTracker is least affected. Compared to the state-of-the-art algorithms, the accuracy is higher and the robustness is better. Specifically, although the other methods are slightly higher in MOTP than ours, it is known from their MOTA and IDs metric that the higher MOTA values obtained after sacrificing the accuracy and does not imply better tracking results.

The results on the entire dataset show that our method can assign correct identities with the help of a high-performance prediction model. Also, the high performance of BVTracker at a low frame rate indicates that the proposed method is more robust compared to the existing state-of-art models, and our method is more stable and applicable to a wider range of

applications.

V. Conclusions

From the start of finding why the visual-based vehicle tracking model cannot achieve the good tracking results, as they perform in the standard dataset, we found the main reason is that the visual feature from a bird's-eye view includes less visual information compared with the pedestrian view. To address both the robustness and real-time requirement, we propose a reliable MVT model by taking vehicles' behavioural information as well as environmental constraints into the MVT problem. The proposed model consists of two branches, an association model based on the Hungarian Algorithm and a trajectory prediction model. In contrast to the existing studies, the trajectory of each vehicle is predicted by considering the inherent relevance and continuity of the vehicle's positions in a certain period. In this paper, the algorithm is validated under different scenarios of light and height. Compared to existing models, BVTracker achieves the fastest frame rate and the best tracking performance. Our model barely lost the target ID, which benefits the traffic data analysis.

However, due to the lack of relevant trajectory datasets, we only validated the scenarios for the freeway. The next objective is to track vehicles in different road structures like roundabouts, and intersections.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61961038), the National Natural Science Foundation of China (Grant No. 52172333), and Key Research and Development Project of Chengdu City (Grant No. 2022YF0500376SN).

References

[1] H. Yao, Z. Duan, Z. Xie, J. Chen, X. Wu, D. Xu, and Y. Gao, "City-scale multi-camera vehicle tracking based on space-time-appearance features," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 3310–3318, June 2022.

[2] K. Fang, Y. Xiang, X. Li, and S. Savarese, "Recurrent autoregressive networks for online multi-object tracking," in *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 466–475, IEEE, 2018.

[3] N. Mahmoudi, S. M. Ahadi, and M. Rahmati, "Multi-target tracking using cnn-based features: Cnnmtt," *Multimedia Tools and Applications*, vol. 78, no. 6, pp. 7077–7096, 2019.

[4] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*, pp. 3645–3649, IEEE, 2017.

[5] Q.-y. He and L.-q. Li, "Pfet: Multi-vehicle tracking with pseudo-feature embeddings for traffic video surveillance," in *2022 25th International Conference on Information Fusion (FUSION)*, pp. 1–7, 2022.

[6] Z. Zhou, J. Xing, M. Zhang, and W. Hu, "Online multi-target tracking with tensor-based high-order graph matching," in *2018 24th International Conference on Pattern Recognition (ICPR)*, pp. 1809–1814, IEEE, 2018.

[7] Y. Li, C. Fu, Z. Huang, Y. Zhang, and J. Pan, "Intermittent contextual learning for keyfilter-aware uav object tracking using deep convolutional feature," *IEEE Transactions on Multimedia*, vol. 23, pp. 810–822, 2021.

[8] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[9] 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*, pp. 779–788, 2016.

[10] Redmon, Joseph and Farhadi, Ali, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7263–7271, 2017.

[11] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[12] 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, IEEE, 2016.

[13] F. Yu, W. Li, Q. Li, Y. Liu, X. Shi, and J. Yan, "Poi: Multiple object tracking with high performance detection and appearance feature," in *European Conference on Computer Vision*, pp. 36–42, Springer, 2016.

[14] M. B. Khalkhali, A. Vahedian, and H. S. Yazdi, "Situation assessment-augmented interactive kalman filter for multi-vehicle tracking," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–11, 2021.

[15] P. Liu, G. Wang, Z. Yu, X. Guo, and W. Lu, "Vehicle tracking based on shape information and inter-frame motion vector," *Computers & Electrical Engineering*, vol. 78, pp. 22–31, 2019.

[16] X. Xiang, W. Bao, H. Tang, J. Li, and Y. Wei, "Vehicle detection and tracking for gas station surveillance based on adaboosting and optical flow," in *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pp. 818–821, IEEE, 2016.

[17] W. Tian and M. Lauer, "Tracking objects with severe occlusion by adaptive part filter modeling - in traffic scenes and beyond," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 4, pp. 60–73, 2018.

[18] R. Sadli, M. Afkir, A. Hadid, A. Rivenq, and A. Taleb-Ahmed, "Aggregated euclidean distances for a fast and robust real-time 3d-mot," *IEEE Sensors Journal*, vol. 21, no. 19, pp. 21872–21884, 2021.

[19] L. He, G. Liu, G. Tian, J. Zhang, and Z. Ji, "Efficient multi-view multi-target tracking using a distributed camera network," *IEEE Sensors Journal*, vol. 20, no. 4, pp. 2056–2063, 2020.

[20] X. Shi, D. Zhao, H. Yao, X. Li, D. K. Hale, and A. Ghiasi, "Video-based trajectory extraction with deep learning for high-granularity highway simulation (high-sim)," *Communications in Transportation Research*, vol. 1, p. 100014, 2021.

[21] U. FHWA, "Next generation simulation (ngsim)," *URL:(visited on 05/01/2019)*, 2019.

[22] E. Barmpounakis and N. Geroliminis, "On the new era of urban traffic monitoring with massive drone data: The pneuma large-scale field experiment," *Transportation Research Part C: Emerging Technologies*, vol. 111, pp. 50–71, 2020.

[23] F. Bi, M. Lei, and Y. Wang, "Context-aware mdnet for target tracking in uav remote sensing videos," *International Journal of Remote Sensing*, vol. 41, no. 10, pp. 3784–3797, 2020.

[24] R. Ke, Z. Li, J. Tang, Z. Pan, and Y. Wang, "Real-time traffic flow parameter estimation from uav video based on ensemble classifier and optical flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 1, pp. 54–64, 2019.

[25] M. Barekatain, M. Martí, H.-F. Shih, S. Murray, K. Nakayama, Y. Matsuo, and H. Prendinger, "Okutama-action: An aerial view video dataset for concurrent human action detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 28–35, 2017.

[26] D. Du, Y. Qi, H. Yu, Y. Yang, K. Duan, G. Li, W. Zhang, Q. Huang, and Q. Tian, "The unmanned aerial vehicle benchmark: Object detection and tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 370–386, 2018.

[27] M. R. Hsieh, Y. L. Lin, and W. H. Hsu, "Drone-based object counting by spatially regularized regional proposal network," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017.

[28] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *European conference on computer vision*, pp. 445–461, Springer, 2016.

[29] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[30] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information [challenge winner iwot4s]," in *International Workshop on Traffic and Street Surveillance for Safety and Security at IEEE AVSS 2017*, 2017.

[31] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International journal of computer vision*, vol. 101, no. 1, pp. 184–204, 2013.

[32] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *arXiv preprint arXiv:1603.00831*, 2016.

[33] B. Keni and S. Rainer, "Evaluating multiple object tracking performance: The clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, no. 1, 2008.
[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[35] 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.
[36] G. Brasó and L. Leal-Taixé, "Learning a neural solver for multiple object tracking," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
[37] L. Leal-Taixé, A. Milan, I. Reid, S. Roth, and K. Schindler, "Motchallenge 2015: Towards a benchmark for multi-target tracking," *arXiv preprint arXiv:1504.01942*, 2015.

Mingxu Li received the M.S. degree in software engineering from Southwest Jiaotong University, Chengdu, China, in 2021. He is currently pursuing his Ph.D. degree in School of Computing and Artificial Intelligence from Southwest Jiaotong University. His research interests include automatic driving and computer vision.

Donghai Zhai (IEEE Member) received his Ph.D. degree in traffic information engineering and control from Southwest Jiaotong University, China, in 2003.

From 2003 to 2005 he was employed at IBM China Research Laboratory. Since 2006 he has been associated with Southwest Jiaotong University in School of Computing and Artificial Intelligence. He has been a visiting scholar at Louisiana State University, Baton Rouge in 2016. His research interests include autonomous driving, digital image processing, computer vision, and pattern recognition.

Da Yang received the B.S. and M.S. degrees in logistics engineering and the Ph.D. degree in transportation engineering from Southwest Jiaotong University, Chengdu, China, in 2007, 2009, and 2013, respectively. He studied in the University of Wisconsin-Madison for two years between August 2010 and August 2012.

He is currently an Associate Professor with the School of Transportation and Logistics, Southwest Jiaotong University. He is also a Post-Doctoral Research Associate with the Traffic Management Research Institute of the Ministry of Public Security. His research interests include automated vehicles, traffic flow, and vehicular networks.

Lihua Xu received the M.S. degree in Communication and Information System and the Ph.D. degree in Optics from Sichuan University, Chengdu, China, in 1999 and 2002 respectively.

He is currently an Associate Professor with the School of Physical Science and Technology, Southwest Jiaotong University. His research interests include automated vehicles and deep learning.