Machine Vision and Applications (2019) 30:1097–1109
https://doi.org/10.1007/s00138-019-01040-w

ORIGINAL PAPER

Trajectory prediction of vehicles turning at intersections using deep neural networks

Mohammad Shokrolah Shirazi[1] · **Brendan Tran Morris**[2]

Received: 19 August 2017 / Revised: 29 May 2018 / Accepted: 20 June 2019 / Published online: 28 June 2019

Abstract
In this paper, an early prediction of vehicle trajectories and turning movements are investigated using traffic cameras. A vision-based tracking system is developed to monitor intersection videos and collect vehicle trajectories with their labels known as turning movements. Firstly, two intersection videos are monitored for 2 h, and collected trajectories with their labels are used to train deep neural networks and obtain the turning models for the prediction task. Deep neural networks are further investigated on a third intersection with different video settings. The future 2 s evaluation of trajectories shows the success of long short-term memory networks to early predict the turning movements with more than 92% accuracy.

Keywords Vehicle trajectories · Traffic cameras · Vision-based tracking system · Trajectory prediction · Long short-term memory networks

1 Introduction

Traffic safety is a major transportation concern in the world. According to a report in [1], 30% of all high severity and 20% of all fatal accidents occur at intersections and junctions. Traffic behavior collection and analysis at intersections such as traffic speed data, motion trajectory and turning movement counts of road users (i.e., vehicle, bicycle and pedestrian) are quite important to reduce accidents, solve traffic jams, and improve safety.

The future perspective of intersection safety relies on safety of road users (e.g., vehicles), infrastructures and their cooperation [2]. For instance, turning recognition and prediction is matter of high interest from the vehicle side (e.g., intelligent vehicles) since they need to evaluate different possible maneuver to take the safe path during their travel time. This can be utilized by the infrastructure side to improve intersection safety due to real-time performance and communication with vehicles [3]. As an outcome, the early prediction of turning movements is an essential objective of intelligent transportation systems (ITS) since it helps to study their behaviors and avoid accidents.

With this motivation in mind, we developed a vision-based framework to detect and track vehicles to collect various traffic behaviors in video streams from stationary cameras, which are usually installed for visually monitoring traffic activities. The installed cameras are inactive most of the time, and they are only used during some incidents (e.g., accident, congestion) to reduce the manual observation cost. An automated vision-based framework could help to reduce this cost and provide useful information for variety of ITS applications such an intelligent route guidance and event recognition to generate accident warning. Traffic congestion and accidents can be reported in real time to vehicles in order to improve traffic mobility since vehicle can consider alternative routes to their destination [2,4].

Turning behavior can be predicted using variety of methods such as Bayesian methods [5] and machine learning techniques. Deep learning is part of a machine learning methods based on learning data representations and has recently shown a great performance in variety of applications including traffic flow prediction [6]. In this work, the collected turning trajectories [7] are used to train the deep neural networks (DNN) and long short-term memory

✉ Mohammad Shokrolah Shirazi
m.shokrolahshirazi@csuohio.edu

Brendan Tran Morris
brendan.morris@unlv.edu

[1] Electrical Engineering and Computer Science, Cleveland State University, 2121 Euclid Avenue, Cleveland, OH 44115, USA

[2] Electrical and Computer Engineering, University of Nevada, 4505 S Maryland Parkway Box 454026, Las Vegas, NV 89154-4026, USA

(LSTM) networks. The obtained models are further utilized to predict the future movements of vehicle tracks and classify the turning predictions. Moreover, the prediction networks are evaluated for three main objectives:

1. The early prediction of future movements is difficult since there is not enough observed trajectory to find the likelihood with obtained models. For example, we are interested to see the future position of a vehicle few frames ahead of its immediate appearance in the field of view (FOV).
2. The long-term prediction of future trajectory is a challenging problem due to lack of data observation in future and other possibilities and errors which grows by time.
3. The early and long-term prediction of turning movements provides a great solution for decision support systems at intersections. The turning movement predictions can be communicated to road users (e.g., vehicles, pedestrians) to prevent accidents which is a future perspective of ITS.

In order to fulfill the mentioned objectives, we used first 30% of each trajectory length for early prediction, and we addressed future motion trajectory of 2 s ahead. The remainder of the paper will explain our method with more details, and it is organized as follows: Sect. 2 describes related works, and Sect. 3 presents the vision-based tracking system. Section 4 explains different training methods to obtain the path models. Section 5 shows experimental evaluation, and finally Sect. 6 concludes the paper.

2 Related work

The prediction and measurement of vehicles speed and acceleration [4,8] are the main subjects of infrastructure-based monitoring, and recognizing turning behaviors and detecting specific maneuver are studied by large body of works [9–11] from the intelligent vehicle's perspective. The turning prediction consists two steps of learning trajectory patterns and finds a match for an observed vehicle trajectory with learned patterns.

Kafer et al. [9] predicts probability values of turning left, right and going straight using Quaternion-based Rotationally invariant Longest Common Substring (QRLCS) as metric for comparison an observed trajectory with turning models. In [10] ego vehicle predicts observed vehicle behavior like stopping or turning by case-based reasoning. Finally, Hulnhagen et al. [12] uses probabilistic finite state machine to model complex driving maneuver based on simple foundational states like breaking, lane to the left and so forth. A Bayes filter approach infers driving maneuver by computing the probability of each basic element in the context of the maneuver model.

Maneuvers can also be learned using regression models. For example, Tran and Firl [11] use likelihood measurement to recognize the maneuver by finding the best match from Gaussian regression models. They predict future trajectory by using particle filters instead of extended Kalman filter which is more appropriate for intersections. Besides accuracy, the early prediction for a long time period is the main goals that have been studied by some studies [9,13]. For instance, the work in [9] provides the prediction for 1–2 s, but 2–4 s is an ideal [2]. Jain et al. [13] work on early prediction of anticipating maneuvers 3.5 s before they occur with accuracy more than 80%.

Deep neural networks have recently shown a great performance in different applications of computer vision such as object classification [14] and video representation [15]. Specially, LSTMs are utilized to learn the temporal structure of the time-series data for human action recognition [16,17] in robotics. For instance, [16] proposes a new class of LSTM network for skeleton-based action recognition, which is capable of focusing on the informative joints using a global context memory cell. Moreover, the conventional LSTMs performance can be further improved by differential gating scheme [18] which emphasizes on the change in information gain caused by the salient motions between the successive frames.

In ITS field, few works have recently studied deep learning networks (DNN) and LSTMs to provide the trajectory prediction with high accuracy [19,20]. For instance, kim et al. [19] proposed a system that predicts the future trajectory of the vehicles over the occupancy grid map and the study in [20] focuses on classification of surrounding vehicles' maneuvers at intersection. Note that our work is different from their method since their dataset comes from variety of sensors (e.g., lidar, image) for autonomous vehicles while we only work on image coordinates (e.g., trajectories) collected through stationary traffic cameras. Moreover, none of the mentioned methods have addressed the early prediction for long time period as we did in this work.

In order to collect the training dataset, a vision-based system is developed to track vehicles and provide trajectories. The tracking system is equipped with zone and trajectory comparison modules to recognize different turnings and provide a label for each turning. The DNN and LSTM networks are trained by the collected trajectories of vehicles entering West, North, East, and South zones. The small portion of the early tracklets is pushed into trained networks to provide the early predictions for 2 s ahead. Moreover, the predicted trajectories are compared against the typical paths to provide the turning prediction such as turn left, right and going straight.

3 Trajectory collection system

A visual tracking system is required to collect object trajectories. Although utilizing multiple features for tracking has been proved as an effective approach, dynamically select the appropriate features is a key problem to deal with different types of variations such as illumination, occlusion and pose [21,22]. As an outcome, a robust feature template learning framework is required to deal with the tracking drifts [23] and shape interactions with shadows and occlusions [24]. For instance, Lan et al. [25] propose multiple sparse representation framework which jointly exploits the shared and feature-specific properties of different features.

In this work, a visual tracking system data collection system is developed based on road user motion cues to provide vehicle trajectories as a data collecting system. The traffic cameras are used to monitored intersections and the vision-based system consists detection, tracking and turning recognition modules [7].

3.1 Detection

The detection module receives sequences of image frames and determines the moving areas as vehicles. The Gaussian mixture model (GMM) [26] is utilized as an adaptive method to build a background model. Vehicles are recognized based on their similarity match with any K Gaussian background models. Moving objects (e.g., vehicles) are detected as pixels that do not fit any of the K background Gaussian models. The foreground image is further processed using morphological operations for clean up, and each moving region is characterized through connected component analysis.

3.2 Tracking

The bipartite graph is used for detected vehicles to find their match with available tracks utilizing distance metric by the greedy approach. In order to find a match and update a track, a detection must fit both a dynamic model and an appearance constraint. Vehicle dynamics are modeling using a constant velocity Kalman filter where the state matrix consists of the bounding box and velocity. The predicted location of a track and the centroid of a detection must be within a small error in order for the detection to match the trajectory dynamic model. Detection must match that of a track for association. For instance, when a detection does not match the existing tracks in the track list, a new track is created. If an existing track does not find a detection for 5 frames, it is marked for deletion.

The appearance model is used to resolve matching ambiguities that may arise during dense traffic situations and occlusions and ensures appearance consistency along a trajectory. The small frame window allows track numbering to remain consistent even after short disappearances such as during occlusion. More details about the tracking system can be found in [7].

3.2.1 Turning recognition

Turning recognition is performed at two different levels. At first level, the turning movements are recognized during tracking using zone comparison module [7,27]. Four cardinal directions {North, South, East, West} along with center of the intersection specify predefined regions, called zones, over the intersection image. The zones are contextually defined based on intersection legs and center. Figure 1 shows a typical intersection with predefined zone areas that are used by the zone comparison module to recognize turning movements of each trajectory.

The zones are used to define a regular sequence (RS) set, which is the set of acceptable zone traversals. For example, {2, 5, 1} indicates a northbound right. During tracking, the passed regions by a vehicle are recorded, which shows the transitions between zones, to build the track zone sequence. When track life time is ended, its associated zone sequence is compared against the members of the RS to provide the count.

The second level of turning recognition is performed after tracking for classifying the predicted trajectories as turn left, turn right or going straight. The typical paths are first recorded for each turning, and they are used for comparison against the predicted trajectories to provide a turning prediction. Although typical paths are usually learned and modeled through the clustering methods (e.g., K mean algorithm, Expectation Maximization Algorithm [28]), we have simply used each noisy free trajectory for each turning since we only focus on outcomes of modeling typical paths for turning evaluation accuracy.

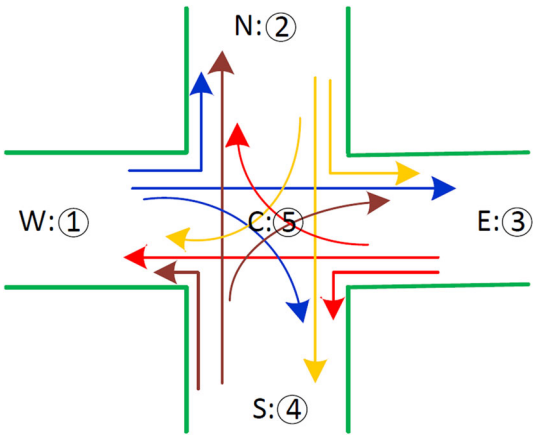


Fig. 1 Typical paths of a typical intersection. Vehicle paths are colored based on direction of travel

The predicted trajectories are recognized as movement paths using temporal alignment techniques for similarity measures of trajectories with known typical paths. Longest common subsequence (LCSS) distance is a popular technique used to compare unequal length trajectories [29]. The LCSS distance is utilized here since it is robust against noise and its high performance has been shown for intersection monitoring and turning movement counts [7,27]. The LCSS distance can be computed as

$$D_{\text{LCSS}}(F_i^{T_i}, F_j^{T_j}) = \frac{\text{LCSS}(F_i^{T_i}, F_j^{T_j})}{\min(T_i, T_j)} \tag{1}$$

where T_i is the length of trajectory F_i. The LCSS is defined in (2), and it provides the total number of matching points between two trajectories. $F^t = \{f_1, \ldots, f_t\}$ represents the trajectory centroid up to time t. ϵ and δ are two major parameters which are empirically chosen based on an intersection setting. ϵ is used to compare and find matching points within a small Euclidean distance and δ is a temporal constraint to ensure that lengths are comparable and meaningful.

$$\text{LCSS}(F_i, F_j) = \begin{cases} 0 & T_i = 0 | T_j = 0 \\ 1 + \text{LCSS}(F_i^{T_i-1}, F_j^{T_j-1}) & d_E(f_{T_i}, f_{T_j}) < \epsilon \text{ and } |T_i - T_j| < \delta \\ \max(\text{LCSS}(F_i^{T_i-1}, F_j^{T_j}), \text{LCSS}(F_i^{T}, F_j^{T_j-1})) & \text{otherwise} \end{cases} \tag{2}$$

The path recognition is performed by comparing the observed trajectory with all the stored typical paths of vehicles and pedestrians. The path with largest D_{LCSS} value is considered the best match, and its label is applied to the object trajectory.

4 Trajectory prediction methods

4.1 Vehicle kinematic

The constant acceleration model is used to predict the future location of the vehicles. This model is used in many tracking systems along with Kalman filter to predict the next trajectory position. However, unlike Kalman filter, observed positions of n future points are not utilized to correct the predictions.

The differential equation of vehicle motion for the uniform acceleration is simple since second derivative of the vehicle's position is constant. The equation results are summarized as below,

$$\begin{aligned} u_i &= u_{i-1} + \Delta t v_{i-1} + \varepsilon_i \\ v_i &= v_{i-1} + \Delta t a_{i-1} + \zeta_i \\ a_i &= a_{i-1} + \Delta t a_{i-1} + \xi_i \end{aligned}$$

where a is the vehicle acceleration, v is speed, x is position, ε_i, ζ_i and ξ_i are Gaussian noises. First three positions of a vehicle are used to initialize the equations and continuously predict the n future trajectory locations. $\Delta t = \frac{1}{\text{fps}}$ where fps (frame per second) is the video frame rate.

4.2 Ridge regression

Regression is a simple approach of supervised learning utilized as a model for understanding the relationship between input and output numerical variables. For example, in linear regression a model is obtained to predict a quantitative response Y from the predictor variable X. For example, following shows a linear regression in which the target value is expected to be a linear combination of the input variables and $\hat{y}$ is the predicted value.

$$\hat{y}(x, y) = w_0 + w_1 x_1 + \cdots + w_p x_p$$

In linear regression linear model with coefficients $w_1, \ldots, w_p$ is fit to minimize the residual sum of squares between the observed responses in the dataset, and the responses predicted by the linear approximation.

Turning movement trajectories can be learned using regression models [9,11,13]. In this work, the ridge regression model is utilized with linear least squares and l2 regularization. A new feature matrix consisting of all polynomial combinations of the features with degree less than one and two is generated separately. For example, if an input sample is two dimensional and of the form $[a, b]$, the degree-2 polynomial features are $[1, a, b, a^2, ab, b^2]$. As an outcome, a nonlinear regression is performed with a linear model using a pipeline to add nonlinear features.

4.3 LSTM

The main drawback of using traditional neural network is its difficulty to understand a subject based on its understanding of previous subjects. Recurrent neural networks (RNN) address this issue since they have feedback allowing information to persist. Another way to think about RNNs is that they have a memory which captures recently calculated information. Although RNNs can provide an information in arbitrarily long sequences, they are limited to looking back

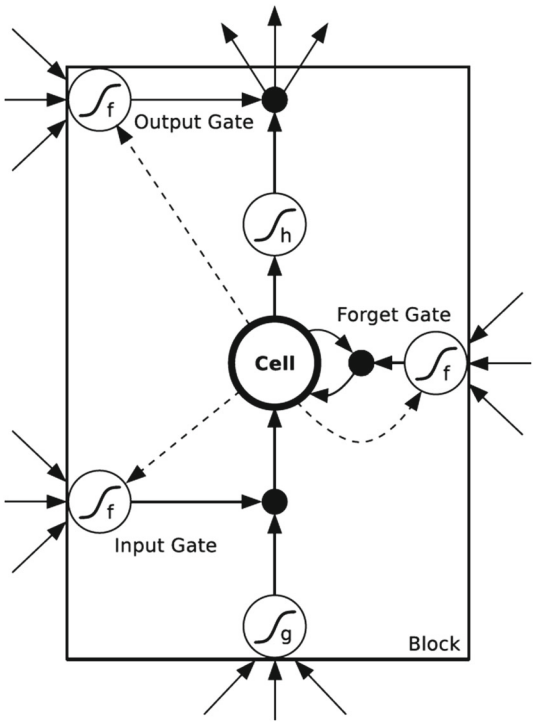


Fig. 2 LSTM memory block

only a few time steps in practice. The solution is using the LSTM networks to learn the long-term dependencies [30].

A LSTM network contains LSTM units which excels at remembering values for either long or short time periods. Figure 2 shows a typical LSTM memory block. Each memory block has forget gate, input gate and output gate in addition to a memory cell. The forget cell decides what proportion of the cell memory should be kept for next time frames. Input gate makes decision about allowing the input data and output gate decides that an output should be sent out of the block.

LSTM networks do not use any activation function within its recurrent components. Thus, the stored value is not iteratively squashed over time, and the gradient does not tend to vanish when data are trained with back-propagation through time. The LSTM networks are easily trainable urging us to study their usefulness for trajectory prediction task.

Due to large variation in the number of collected (x, y) samples and their pattern models, variety of LSTM architectures were utilized to fit the training datasets. To better learn the temporal representations, different LSTMs and dense layers were stacked on top of each other to find the best network architecture for each turning model. Figure 3 shows the LSTM architectures used to train different turning trajectories. The LSTMs and dense layers are shown with the format of (N, W), and (N) where N represents number of cells for the layer and W is the window size (see Sect. 4.3.1). The most common architecture with the least validation error is a two-layer architecture shown in Fig. 3a. However, nonlinear patterns with high variation required more complicated architectures shown in Fig. 3c.

The total number of parameters (p) varied based on number of layers and nodes for training weights. Obviously, the total number of weights is higher for deeper LSTM architecture (Fig. 3c) while the shallow LSTM network has the lowest number of parameters (Fig. 3a).

4.3.1 Windowing

In order to better use the history of the trajectories for training and prediction steps, multiple recent time frames (i.e., steps) are utilized with the LSTM network. This is called a windowing method, and the size of the window is a parameter that were carefully tuned during training and validation steps of LSTM architectures.

For example, suppose by given the trajectory position at current time (x_t, y_t) for window size one (i.e., $k = 1$), the next trajectory prediction is position (x_{t+1}, y_{t+1}). The position pairs of current time (x_t, y_t), as well as the two prior times (x_{t-1}, y_{t-1}), and (x_{t-2}, y_{t-2}) can be stacked as input feature for window size three. Varying window size is effective when there are more network layers to estimate complex model.

4.4 DNN

A deep neural network (DNN) is an artificial neural network (ANN) with multiple hidden layers between the input and output layers. DNN can model complex nonlinear relationships similar to shallow ANN. However, extra layers enable composition of features from lower layers, potentially modeling complex data with fewer units than a similarly performing

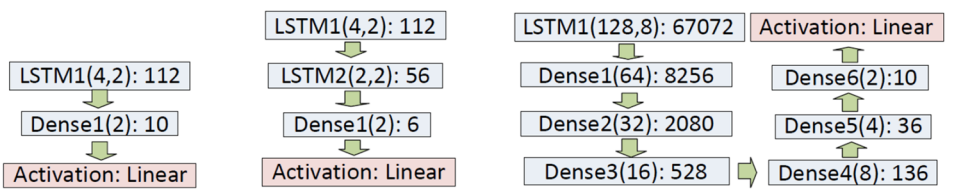


Fig. 3 LSTM architectures with total number of parameters (p)

(a) Most common LSTM architecture, $p = 122$

(b) LSTM architecture for WN trajectories (INT1), $p = 174$

(c) LSTM architecture for WS, NW trajectories (INT1), $p = 78118$

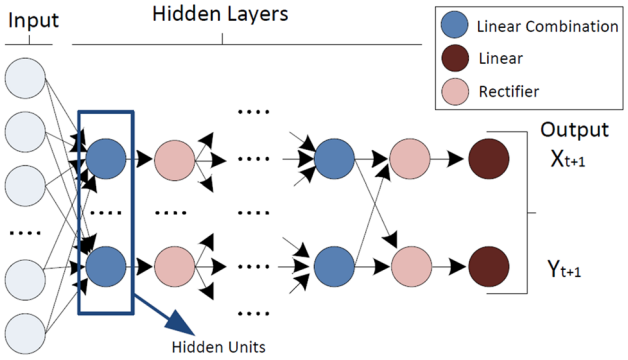


Fig. 4 Deep neural network architecture

shallow network. More data can increase the performance of DNN but requires more computation time to train.

The DNN has recently shown promising results for traffic flow prediction with big data [31] and this urged us to evaluate them for trajectory prediction and vehicle turning classification. The addition of more hidden layers and rectifiers (e.g., $ReLUs$) helps to come up with complex functions to provide a model with better accuracy that can capture features from different levels of the hierarchy. Figure 4 shows the typical structure of the developed DNN architecture.

Figure 5 shows DNN architectures used to train different turning trajectories. The Dense layers are shown with the format of $(N) : lp$, where N represents number of cells and lp is number of parameters for each layer. Moreover, kernel initializer and activation function were set to 'normal' and 'relu,' respectively. The two- and three-layer architectures are sufficient to find the turning models with least validation error. However, introducing dropout layers is essential to avoid overfitting and generalize the learned model for test datasets with least validation errors.

5 Experimental results

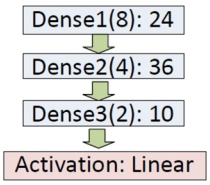
The vision-based tracking system was implemented in C++ using OpenCV 2.3, and it was run on quad core Intel i7 processor with 6 GB RAM. The initial steps of experiments included scene preparation and arranging the collected trajectories into training and test and datasets.

For trajectory training and prediction experiments, the Theano framework with Keras wrappers was used in a Python environment. Training and testing of trajectories and turnings were performed on a dedicated, high end PC with a quad core intel i7-7700 3.6 GHz processor, 16 GB RAM and an Nvidia Geforce GTX 1070 GPU. The GTX 1070 has 8 GB of total RAM with 2048 CUDA cores, allowing for fast parallelization during training and testing of deep learning architectures.

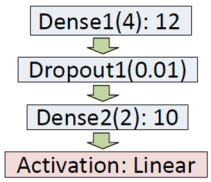
5.1 Scene preparation

Scene preparation is a first step required by the vision-based tracking system. The zone areas are defined based on some (x, y) points and line equations for zone comparison module to provide turning movement labels during the tracking. Figure 6 shows the predefined zone areas determined by

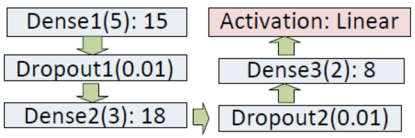
Fig. 5 DNN architectures with total number of parameters (p)



(a) Most common DNN architecture, $p = 70$

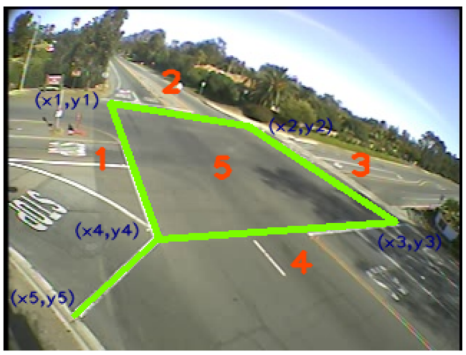


(b) DNN architecture for WS trajectories (INT1), $p = 22$

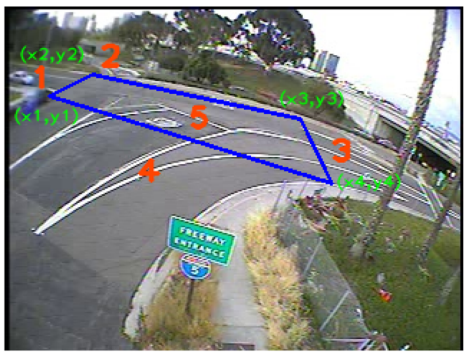


(c) DNN architecture for WS trajectories (INT2), $p = 41$

Fig. 6 Zone areas



(a) INT 1



(b) INT 2

Fig. 7 Typical paths

(a) INT 1 **(b)** INT 2

Table 1 Number of training tracks, test tracks and experiments for each turning

Intersection	Quantity	WN	WE	WS	NW	NE	NS	EW	EN	ES	SW	SN	SE
INT 1	Training tracks	15	46	110	24	32	272	68	46	61	85	240	40
	Testing tracks	7	23	54	12	15	134	34	23	30	42	118	19
	Experiments	10	38	65	9	15	160	56	22	55	88	262	30
INT 2	Training tracks	–	473	8	–	12	12	–	–	–	–	–	108
	Testing tracks	–	232	4	–	5	5	–	–	–	–	–	53
	Experiments	–	238	9	–	8	13	–	–	–	–	–	57
INT 3	Training tracks	11	6	–	11	47	309	–	45	36	9	300	–
	Testing tracks	5	3	–	6	23	152	–	22	18	4	148	–
	Experiments	10	4	–	2	23	18	–	20	17	3	38	–

couple of points and line equations. The predefined zone areas are used by the tracking system to provide a real-time count during the runtime.

The second essential step is the definition of typical paths. The typical paths are defined for each turning movement based on sequence of (x, y) points, and they can help to provide a label for the turning prediction phase. The LCSS method is used to compare the generated predicted points against each typical path and return the one that has the highest similarity score based on number of matched (x, y) points.

Figure 7 shows the defined typical paths for two intersections used in our evaluation. The trajectories of each ways are colored based on their starting zone area. The paths are color-coded based on their approach zone. All paths starting in a particular zone have the same color. The typical path starting from North, West, South, and East zones has red, green, brown, and yellow colors, respectively.

5.2 Dataset preparation

The developed tracking system collects vehicle trajectories of two intersection videos with rate of 15 frames per seconds (FPS) and 2-h time period [7]. The third intersection has low frame rate of 5 FPS.

Turning movement count results are shown in Table 1 that were collected using the developed vision-based tracking system. The total turning movements trajectories are used to prepare our dataset by 67–33% split for training and test datasets. For instance, high vehicle flow is manifested for going straight paths (SN) of first intersection (INT 1) as 358 vehicles that are arranged by 240–118 for training and test trajectories.

Table 1 also shows the total number of experiments for each turning movement. In order to perform an early prediction of turning movements, prediction module receives the third trajectory point (e.g., (x, y)) of test dataset. First two trajectory points were used by kinematic method to estimate velocity and acceleration for future predictions.

The prediction is performed for next 2 s which is $t = 2 \times FPS$ future points (e.g., 28 for INT 1,2) since intersection videos were recorded by the rate of 14 frame per second. The prediction process is continue for the same trajectory if $3+t \times i$ is less than 30% of the length of the trajectory in order to meet the early prediction requirement. i is a counter variable to control the loop initialized with zero. Since turning movements of West–East (WE) for INT 2 and South–North (SN) for INT 1 have the highest flow, they lead to highest number of vehicle tracks. As an outcome, more experiments were performed for them.

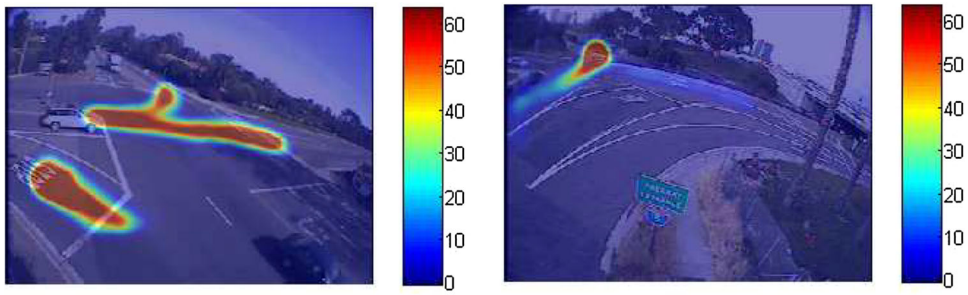


Fig. 8 Heatmaps of training dataset

(a) Entered vehicles from west zone (INT 1) **(b)** Entered vehicles from north zone (INT 2)

Table 2 A typical evaluation of different widow sizes for ES and WE movements

Intersection	TM	Type	2	3	4	5	6	7	8	9	10
INT 1	ES	AE	13.64	29.53	11.46	18.64	19.42	13.21	16.08	18.6	15.66
		TPA	90%	0	0	0	0	2%	0	0	2%
INT 2	WE	AE	21.23	30.83	34.62	37.97	16.62	16.21	23.88	17.35	29.51
		TPA	100%	100%	100%	100%	100%	100%	100%	100%	5%

Figure 8 shows the generated heatmaps from the training datasets for EW and NS movements of INT 1 and INT 2, respectively. As shown in Fig. 8a, turning right vehicles entering different areas than those intend to take straight or left turn. The higher width area of turning rights indicates more variance among these trajectories in comparison to those to other turnings.

Figure 8b shows high frequency of trajectories at north zone representing entering vehicles from the North. However, these trajectories are split into south and west ways which shown with the low density (i.e., blue color) in the heatmap image. This just presents the 24 trajectories shown in Table 1 for NS and NE paths.

The collected turning trajectories were pushed into different LSTM and DNN architectures to find the best models. The mentioned architectures (e.g., see Sect. 4.3) with different windows sizes were evaluated to find the best fit for each turning movement. For example, Table 2 shows the turning prediction accuracy (TPA) and average error (AE) for different window sizes of ES and WE movements for INT 1 and INT 2. The windows size of 2 is appropriate for ES since it has lower AE and higher TPA. WE movement has same TPA for different window sizes of INT 2 and window size of 7 is selected base on lower AE value.

The maximum 500 and 1000 epochs were used to find the models with least mean squared error for each turning trajectories in DNN and LSTM networks, respectively. The main optimizer was Adam and rectifiers were used as activation functions for DNN. Since we addressed a regression problem for (x, y) coordinates, the activation function of last layer is linear for DNN and LSTM. The batch size and verbose of 10 and 2 were selected, respectively.

5.3 Results

The predictions are performed for 28 future points (i.e., frames) to construct the 2 s ahead. The average accuracy (AE) is calculated in Eq. (3).

$$\mathrm{AE} = \frac{\sum_{i=1}^{N} \sqrt{\left(x_i - \hat{x}_i\right)^2 + \left(y_i - \hat{y}_i\right)^2}}{N} \tag{3}$$

x_i, y_i are actual image coordinates of a turning trajectory and $\hat{x}_i$, $\hat{y}_i$ are the predicted coordinates. N is set to 28 in our experiments which corresponds to 2 s ahead. Turning prediction accuracy (TPA) is obtained by Eq. (4), where M is the total number of experiments for each turning movement and TP is a correct turning prediction as right, go straight or left.

$$\mathrm{TPA} = \sum_{i=1}^{M} \frac{\mathrm{TP}}{M} \tag{4}$$

Figure 9 shows the prediction module for LSTM and DNN networks. For instance, the module receives an early trajectory point like (x_0, y_0), (x_1, y_1) for window size $k = 2$, the obtained trained models from the training process and typical paths that were explained in Sect. 5.1.

Each next predicted point is feeding back to the module to provide the next prediction points (e.g., $(\hat{x}_k, \hat{y}_k)$). The predicted trajectory up to t frames (e.g., $t = 28$ for INT 1, 2) is pushed into the LCSS to find the match by comparison with each three typical paths. Since the tracker provides the entering zone of each trajectory, the comparison with only three possible paths are sufficient to find a predicted turning.

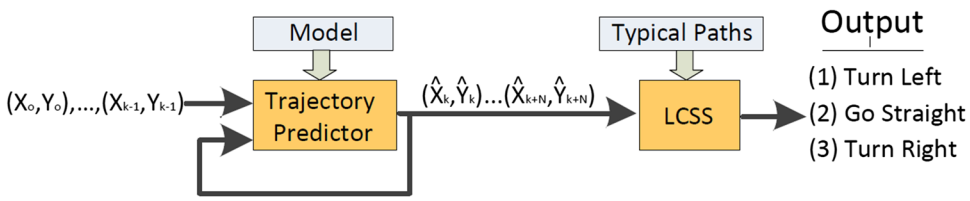


Fig. 9 Turning prediction module

Table 3 Turning prediction accuracy and average error results (INT 1)

ME	Type	WN	WE	WS	NW	NE	NS	EW	EN	ES	SW	SN	SE	Avg
KINE	AE	13.21	6.92	30.45	21.05	15.52	15.01	9.35	20.31	7.7	13.32	8.71	11.43	14.41
	TPA	100%	89%	82%	100%	100%	98%	85%	82%	58%	90%	92%	40%	84%
REG	AE	30.07	55.36	15.33	7.18	38.42	25.11	59.89	39.14	7.99	31.79	17.61	21.73	29.15
	TPA	100%	29%	77%	100%	62%	100%	100%	86%	80%	100%	97%	43%	81%
LSTM	AE	20.59	15.26	18.67	19.51	16.95	8.95	7.63	11.6	13.64	23.2	20.09	22.30	16.53
	TPA	100%	87%	91%	100%	100%	99%	93%	100%	90%	94%	92%	73%	93%
DNN	AE	15.4	55.42	13.97	16.99	19.87	17.14	22.76	16.18	30.87	20.3	15.65	16.68	21.76
	TPA	100%	68%	83%	100%	100%	99%	98%	82%	98%	99%	100%	83%	92%

ME method, *KINE* kinematic, *REG* regression, *Avg* average

Table 4 Turning prediction accuracy and average error results (INT 2)

Method	Results type	WE	WS	NE	NS	SE	Avg
Kinematic	Average error	46.46	8.4	29.95	6.66	52.38	28.77
	Turning prediction accuracy	100%	33%	25%	100%	100%	71%
REG	Average error	64.67	5.62	29.68	6.61	37	28.71
	Turning prediction accuracy	100%	44%	12%	100%	100%	71%
LSTM	Average error	21.41	22.58	12.93	7.93	28.1	18.29
	Turning prediction accuracy	100%	100%	100%	100%	100%	100%
DNN	Average error	26.3	30.68	33.79	19.41	37.36	29.5
	Turning prediction accuracy	100%	100%	100%	100%	100%	100%

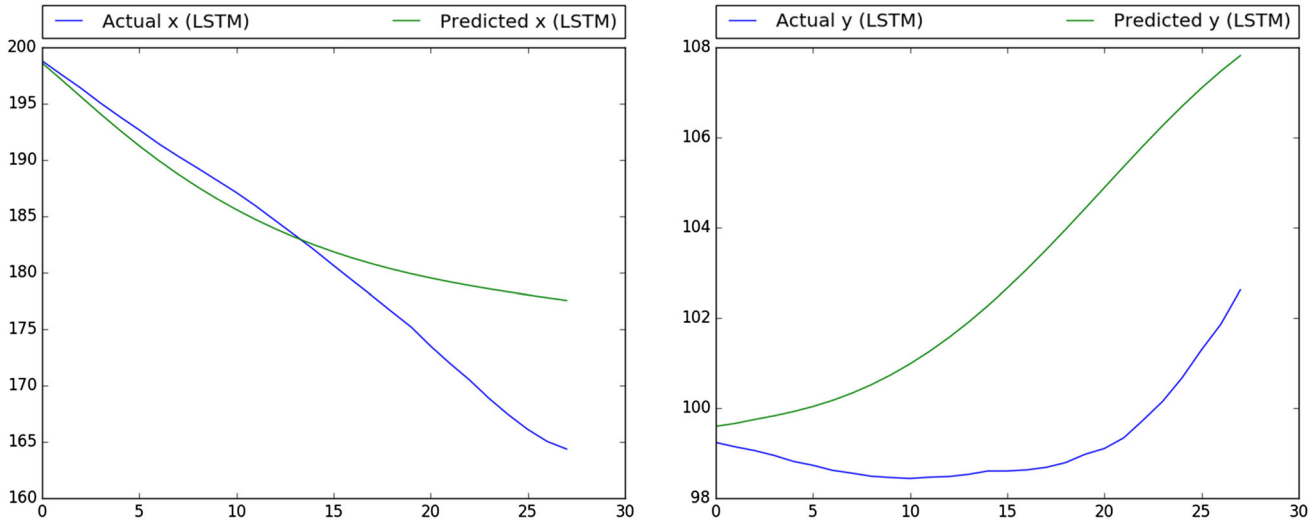
The turning prediction accuracy (TPA) and average error (AE) results of future 2 s are shown in Tables 3 and 4. The simple kinematic approach is a linear method showing a good performance for straight paths of INT 1 such as West–East (WE) and South–North (SN). This is shown by low AE and high TPA values in comparison with other turning movements (i.e., red colors in Table 3). The ridge regression method shows a good performance for few movements such as NW movement (i.e., brown colors). DNN and LSTM networks are successful in estimating linear and nonlinear movements including NS and EW. However, DNN shows higher AE in comparison with LSTM (i.e., green colors in Table 3).

Although some turning movements such as ES show lower average error for Kinematic approach, they do not have higher turning prediction accuracy in comparison with LSTM, and DNN. The LSTM network is good to predict the curvy patterns for some portion of future seconds ahead though it has distance/bias from the actual trajectory. For instance, Fig. 10 shows that error is not always linearly increased for x values and predicted y values keep the actual pattern with some bias. The average of all turning movements indicates that LSTM network has higher performance than DNN due to lower AE (i.e., magenta colors in Table 3).

Similar to Table 3, Table 4 indicates better results in average for DNN, and LSTM networks (i.e., see magenta colors in Table 4). While vehicle kinematic and ridge regression methods show the similar accuracy, LSTM outperforms DNN in AE, but they show the similar accuracy in turning predictions. The main difference has been revealed for WS and NE due to great improvement in TPA value (i.e., red colors). There is also a significant reduction in average error in WE and SE shown with blue font in Table 4.

5.3.1 Intersection 3 with new settings

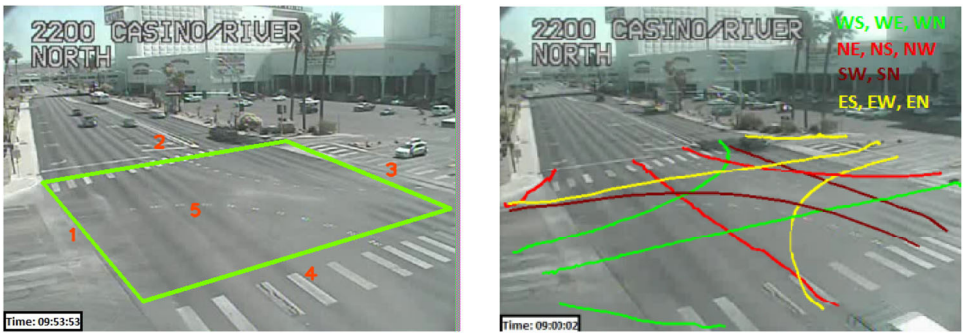
In order to verify the applicability of the proposed methods for trajectory prediction, a new intersection (INT 3) was monitored using the proposed tracking system for 2 h with new configuration. The captured video recorded with low frame



(a) Prediction of x coordinates (AER: 3.93) **(b)** Prediction of y coordinates (AER: 3.51)

Fig. 10 Trajectory prediction of a vehicle following ES path

Fig. 11 Zone areas and typical paths



(a) Zones (INT 3) **(b)** Paths (INT 3)

rate of 5 FPS leading to higher displacement showing vehicle movements. Moreover, the camera has lower distance to intersection which intensifies the displacement movements of vehicles. The new intersection setting leads to collect the turning movement trajectories with shorter length in comparison with INT 1 and INT 2.

Figure 11 shows the predefine zone areas and typical paths for the INT 3. Table 1 shows number of collected turning movement trajectories used to train trajectory prediction models. The table shows the lower number of trajectory with shorter lengths for WE, WN, and SW typical paths.

Similar to other two intersections, the collected trajectories of turning movements are plugged into REG, DNN and LSTM networks to obtain the turning models. The similar architectures (see Sects. 4.3, 4.4) with experiment settings are utilized, but the prediction is performed for $t = 10$ future points for 2 s due to low frame rate of the intersection video (i.e., 5 FPS).

Table 5 shows the TPA and AE values of evaluated turning movements of INT 3. Although the AE values of kinematic method are lower than DNN (e.g., see magenta colors), LSTM outperforms all other methods. Similarly, ridge regression shows the worst performance based on AE value.

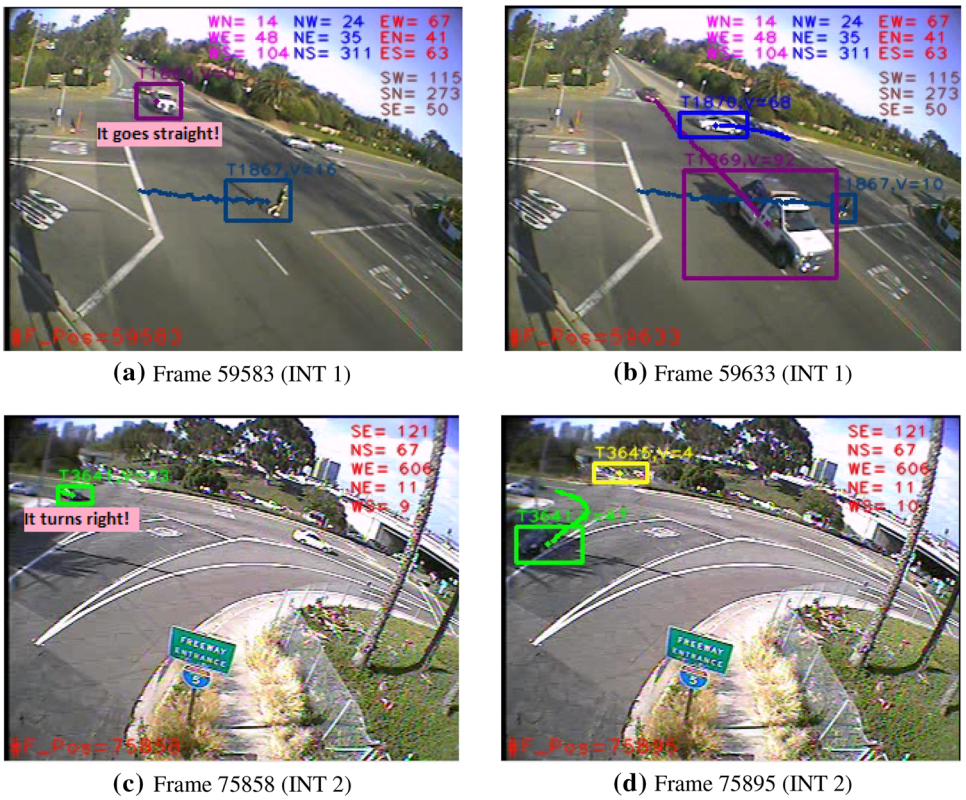
Figure 12 depicts some typical turning predictions of vehicles for two different intersections. Figure 12a shows the correct prediction for an appeared vehicle in north zone (i.e., magenta track) which has a pedestrian ahead running in front of it. The early prediction is also shown in Fig. 12c for the vehicle in west zone (i.e., green track) which seems to go straight, but it actually turns right. The accuracy of predictions can be evaluated using Fig. 12b, and 12d for same tracked vehicles.

6 Concluding remarks

This works take an advantage of deep neural networks for long-term trajectory prediction of vehicles from traffic cameras perspective. The developed vision-based system collected turning movement trajectories to train the DNN,

Table 5 Turning prediction accuracy and average error results (INT 3)

ME	Type	WN	WE	NW	NE	NS	EN	ES	SW	SN	Avg
KINE	AE	22.05	20.5	20.99	50.22	78.64	5.6	30.36	13.57	63.45	33.93
	TPA	100%	100%	100%	91%	94%	100%	35%	100%	61%	87%
REG	AE	57.88	183.47	18.12	60.78	55.78	126.2	46.05	61.2	40.11	72.17
	TPA	60%	100%	100%	83%	100%	100%	71%	100%	100%	90%
LSTM	AE	28.9	26.54	14.02	31.67	47.75	13.95	27.25	37.27	22.21	27.72
	TPA	100%	100%	100%	96%	100%	100%	53%	100%	79%	92%
DNN	AE	62.89	37.93	20.97	45.85	50.18	34.65	42.66	94.1	43.03	48.02
	TPA	90%	100%	100%	100%	100%	100%	35%	100%	58%	87%



(a) Frame 59583 (INT 1) **(b)** Frame 59633 (INT 1)

(c) Frame 75858 (INT 2) **(d)** Frame 75895 (INT 2)

Fig. 12 Early prediction of vehicles turnings

LSTM networks and find an appropriate turning models for early prediction of future 2 s. The predicted trajectory points show good accuracy when compared with ground truth that can be used to provide turning prediction as well. The proposed system can be used in variety of scenarios as follows:

1. The future perspective of intersection safety relies on data collection/analysis from different sensors of infrastructures, road users, and the real-time communication and sharing of useful data among them. For instance, the future trajectory of a vehicle and a pedestrian can show whether there is a high probability of an accident to warn the pedestrian ahead.
2. Besides real-time accident detection and warning road users, such an approach can help to study the hazardous level of intersections by finding spatial/temporal conflicts between vehicles and pedestrians. For example, time to collision (TTC) data can be estimated for each intersection during tracking [32] and TTC heatmap will help road users to chose a safe route toward their destinations. This will lead to higher traffic mobility and intersection safety.

References

1. LeóCano, J.A., Kovaceva, J., Lindman, M., Brännström, M.: Automatic incident detection and classification at intersections. In: 21st International Conference on Enhanced Safety of Vehicles, ESV. Paper 09–0234 (2009)

2. Shirazi, M.S., Morris, B.T.: Looking at intersections: a survey of intersection monitoring, behavior and safety analysis of recent studies. IEEE Trans. Intell. Transp. Syst. **18**, 4–24 (2017)
3. Shirazi, M.S., Morris, B.: Observing behaviors at intersections: a review of recent studies and developments. In: IEEE Intelligent Vehicles Symposium (IV), pp. 1258–1263 (2015)
4. Kumar, P., Ranganath, S., Weimin, H., Sangupta, K.: Framework for real-time behavior interpretation from traffic video. IEEE Trans. Intell. Transp. Syst. **6**, 43–53 (2005)
5. Wu, J., Cui, Z., Zhao, P., Chen, J.: Traffic Vehicle Behavior Prediction Using Hidden Markov Models, pp. 383–390. Springer, Berlin (2012)
6. Ma, X., Dai, Z., He, Z., Wang, Y.: Learning traffic as images: a deep convolution neural network for large-scale transportation network speed prediction. CoRR (2017). arXiv:1701.04245
7. Shirazi, M.S., Morris, B.T.: Vision-based turning movement monitoring: count, speed and waiting time estimation. IEEE Intell. Transp. Syst. Mag. **8**, 23–34 (2016)
8. Viti, F., Hoogendoorn, S.P., van Zuylen, H.J., Wilmink, I.R., van Arem, B.: Speed and acceleration distributions at a traffic signal analyzed from microscopic real and simulated data (2008)
9. Kafer, E., Hermes, C., Wohler, C., Ritter, H., Kummert, F.: Recognition of situation classes at road intersections. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3960–3965 (2010)
10. Graf, R., Deusch, H., Seeliger, F., Fritzsche, M., Dietmayer, K.: A learning concept for behaviour prediction at intersections. In: IEEE Intelligent Vehicles Symposium (IV), Dearborn, Michigan, pp. 939–945 (2014)
11. Tran, Q., Firll, J.: Online maneuver recognition and multimodal trajectory prediction for intersection assistance using non-parametric regression. In: IEEE Intelligent Vehicles Symposium (IV), Dearborn, Michigan, pp. 918–923 (2014)
12. Hulnhagen, T., Dengler, I., Tamke, A., Dang, T., Breuel, G.: Maneuver recognition using probabilistic finite-state machines and fuzzy logic. In: Proceedings of IEEE Intelligent Vehicles Symposium, San Diego, USA, pp. 65–70 (2010)
13. Jain, A., Koppula, H.S., Raghavan, B., Soh, S., Saxena, A.: Car that knows before you do: anticipating maneuvers via learning temporal driving models. In: 2015 IEEE International Conference on Computer Vision (ICCV), pp. 3182–3190 (2015)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. CoRR (2015). arXiv:1512.03385
15. Srivastava, N., Mansimov, E., Salakhutdinov, R.: Unsupervised learning of video representations using lstms. CoRR (2015). arXiv:1502.04681
16. Liu, J., Wang, G., Duan, L., Hu, P., Kot, A.C.: Skeleton based human action recognition with global context-aware attention LSTM networks. CoRR (2017). arXiv:1707.05740
17. Liu, J., Shahroudy, A., Xu, D., Kot, A.C., Wang, G.: Skeleton-based action recognition using spatio-temporal LSTM network with trust gates. CoRR (2017). arXiv:1706.08276
18. Veeriah, V., Zhuang, N., Qi, G.: Differential recurrent neural networks for action recognition. CoRR (2015). arXiv:1504.06678
19. Kim, B., Kang, C.M., Lee, S., Chae, H., Kim, J., Chung, C.C., Choi, J.W.: Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network. CoRR (2017). arXiv:1704.07049
20. Khosroshahi, A., Ohn-Bar, E., Trivedi, M.M.: Surround vehicles trajectory analysis with recurrent neural networks. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 2267–2272 (2016)
21. Lan, X., Ma, A.J., Yuen, P.C.: Multi-cue visual tracking using robust feature-level fusion based on joint sparse representation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1194–1201 (2014)
22. Lan, X., Zhang, S., Yuen, P.C.: Robust joint discriminative feature learning for visual tracking. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence. IJCAI'16, pp. 3403–3410. AAAI Press (2016)
23. Lan, X., Yuen, P.C., Chellappa, R.: Robust mil-based feature template learning for object tracking. In: AAAI Conference on Artificial Intelligence (2017)
24. Pernici, F., Bimbo, A.D.: Object tracking by oversampling local features. IEEE Trans. Pattern Anal. Mach. Intell. **36**, 2538–2551 (2014)
25. Lan, X., Zhang, S., Yuen, P.C., Chellappa, R.: Learning common and feature-specific patterns: a novel multiple-sparse-representation-based tracker. IEEE Trans. Image Process. **27**, 2022–2037 (2018)
26. Stauffer, C., Grimson, W.: Adaptive background mixture models for real-time tracking. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 2, p. 252 (1999)
27. Shirazi, M.S., Morris, B.: Vision-based turning movement counting at intersections by cooperating zone and trajectory comparison modules. In: IEEE Intelligent Transportation Systems Conference (ITSC), Qingdao, China, pp. 3100–3105 (2014)
28. Akoz, O., Karsligil, M.: Traffic event classification at intersections based on the severity of abnormality. Mach. Vis. Appl. **25**, 613–632 (2014)
29. Morris, B., Trivedi, M.: Learning trajectory patterns by clustering: experimental studies and comparative evaluation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 312–319 (2009)
30. Bengio, Y., Simard, P., Frasconi, P.: Learning long-term dependencies with gradient descent is difficult. IEEE Trans. Neural Netw. **5**, 157–166 (1994)
31. Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F.Y.: Traffic flow prediction with big data: a deep learning approach. IEEE Trans. Intell. Transp. Syst. **16**, 865–873 (2015)
32. Shirazi, M.S., Morris, B.T.: Investigation of safety analysis methods using computer vision techniques. J. Electron. Imaging **26**, 051404 (2017)

Mohammad Shokrolah Shirazi received his PhD degree in electrical and computer engineering from University of Nevada, Las Vegas (2016). He received his BS degree in computer engineering from Ferdowsi University of Mashhad (2005) and his MS degree in computer architecture from Sharif University of Technology, Tehran, Iran (2007). He has been working as a visiting assistant professor at Cleveland State University, and he was the member of the "Real-Time Intelligent Systems Laboratory" at UNLV. Dr. Shirazi researches in computer vision, machine learning, embedded systems and their applications in intelligent transportation systems, robotics and health. His dissertation "Vision-based Intersection Monitoring: Behavior Analysis & Safety Issues" received two awards as the second-winner place for UNLV, College of Engineering, and the IEEE ITSS Best Dissertation Award in 2016.

Brendan Tran Morris received his BS degree from the University of California, Berkeley, in 2002 and his MS and PhD from the University of California, San Diego (UCSD), in 2006 and 2010, respectively. He is currently Professor of Electrical and Computer Engineering, the Founding Director of the Real-Time Intelligent Systems (RTIS)

Laboratory, and the Director of the Transportation Research Center (TRC) at the University of Nevada, Las Vegas. He and his team conduct research on computationally efficient systems that utilize computer vision techniques for activity analysis, situational awareness, and scene understanding. Dr. Morris serves as an Associate Editor for the IEEE Intelligent Transportation Systems Magazine and the IEEE Robotics and Automation Letters and is a Guest Editor for the Special Issue on ITS Empowered by AI Technology for the IEEE Transactions on Intelligent Transportation Systems. His current research focuses on understanding and predicting human behavior and activity which has been applied toward traffic and pedestrian safety and for action quality assessment.