Arabian Journal for Science and Engineering
https://doi.org/10.1007/s13369-020-04546-y

RESEARCH ARTICLE-CIVIL ENGINEERING

Extraction of Vehicle Turning Trajectories at Signalized Intersections Using Convolutional Neural Networks

Osama Abdeljaber[1] · Adel Younis[2] · Wael Alhajyaseen[3]

Received: 22 April 2019 / Accepted: 16 April 2020

Abstract
This paper aims at developing a convolutional neural network (CNN)-based tool that can automatically detect the left-turning vehicles (right-hand traffic rule) at signalized intersections and extract their trajectories from a recorded video. The proposed tool uses a region-based CNN trained over a limited number of video frames to detect moving vehicles. Kalman filters are then used to track the detected vehicles and extract their trajectories. The proposed tool achieved an acceptable accuracy level when verified against the manually extracted trajectories, with an average error of 16.5 cm. Furthermore, the trajectories extracted using the proposed vehicle tracking method were used to demonstrate the applicability of the minimum-jerk principle to reproduce variations in the vehicles' paths. The effort presented in this paper can be regarded as a way forward toward maximizing the potential use of deep learning in traffic safety applications.

Keywords Traffic safety · Signalized intersections · Turning vehicle trajectories · Convolutional neural networks · Minimum-jerk principle

1 Introduction

Road traffic safety is increasingly an issue of global concern. Recently, it has been estimated that 1.4 million people die and 73.25 million get disabled annually as a result of road traffic injuries worldwide [1]. Globally, the annual cost estimation for deaths, injuries, and disabilities due to road crashes is approximately 518 billion dollars, which makes up around 1.5% of the gross national product for middle-income countries [1]. Intersections are recognized as the most complex locations within a highway system, in which conflicts are easily generated, and thus traffic crashes are more likely to occur [2]. Despite them constituting a small part of the highway systems, intersection-related crashes share over 50% of all crashes in urban areas and 30% of those in rural regions [2]. Therefore, intersections are deemed crash-prone locations due to the large number of conflict points between traffic streams moving in different direction. Turning traffic has a major role in the safety performance of intersections due to the nature of their maneuvers which are usually characterized with significant variations in paths and speeds depending on drivers' targeted exit lane, their instinctive judgment, intersection geometry, and other factors [3]. As left-turning vehicles (right-hand traffic rule) pass the stop line to the intersection zone, their driving routes are often changed randomly leading generally to serious conflicts and unsmooth driving, which in turn impacts on the traffic safety [4]. Therefore, analyzing the trajectories of turning vehicles is required so as to improve safety performance at signalized intersections.

Two approaches are classically implemented to evaluate the safety performance at intersections, namely, post- and preimplementation assessments. The former involves collecting the data after implementing the countermeasure, while the latter enables the engineers to predict the safety performance at the planning stage and thus is more feasible [5]. Simulation tools are deemed promising since they provide more flexibility and opportunity to achieve reliable preimplementation safety assessments and thus overcome the

✉ Osama Abdeljaber
osama.abdeljaber@lnu.se

[1] Department of Building Technology, Faculty of Technology, Linnaeus University, P.O. Box 35195, Växjö, Sweden

[2] Department of Civil and Architectural Engineering, College of Engineering, Qatar University, P.O. Box 2713, Doha, Qatar

[3] Qatar Transportation and Traffic Safety Center, College of Engineering, Qatar University, P.O. Box 2713, Doha, Qatar

Published online: 04 May 2020

limitations associated with postimplementation assessments. However, the current simulation software tools are still oversimplified, and therefore, the consequent safety assessments at intersections are not sufficiently reliable [6, 7]. Recently, driving simulators and virtual reality systems are emerging as new tools to study road user behavior in combination with microscopic simulation tools [8–10]. These advanced tools are rapidly replacing the traditional traffic safety assessment techniques. However, realistic representation of vehicle trajectories (including path, speed, and acceleration profiles) is essential in such applications for a reliable safety assessment. Furthermore, the availability of realistic and accurate models for the trajectories of turning traffic is critically needed for the motion planning of autonomous vehicles.

The majority of vehicle path models available in the literature are developed based on a number of trajectories that are manually extracted from recorded videos. The process of manual trajectory extraction, however, can be tiresome and time-consuming since it requires tracking every single vehicle in a frame-by-frame manner. This becomes particularly problematic when a large number of trajectories are required for building an accurate vehicle trajectory model. Alternatively, automatic trajectory extraction techniques have been proposed [11–13]. Yet, most previously developed automatic trajectory extraction approaches require background subtraction to detect moving vehicles. This process is significantly vulnerable to factors such as light and shadow conditions, occlusion with obstacles and other vehicles, and camera's position and viewing angle [14]. In view of that, the effort presented in this paper aims at developing an effective tool for automatic trajectory extraction of turning vehicles, and the proposed tool relies on convolutional neural networks (CNNs) to perform the vehicles' detection task.

The motivation of using CNNs in this work is twofold:

1. CNNs have recently become the *de facto* standard for computer vision and pattern recognition as they achieved the state-of-the-art performances in challenging tasks such as handwriting recognition, classification of large image archives, and face segmentation. In the context of traffic engineering, successful applications of CNN have been reported including flow speed prediction [15], traffic density measurement [16, 17], pavement distress detection [18], road crack detection [19], and detection of traffic signs [20–23] or pedestrians [24, 25].
2. CNNs operate directly on the raw videos without requiring prior image preprocessing or background subtraction.

In this paper, the proposed vehicle tracking tool is used for an automatic extraction of left-turning vehicles trajectories at a signalized intersection located in Doha City, State of Qatar. The extracted trajectories are then compared to their manually extracted counterparts in order to demonstrate the accuracy of the CNN-based approach. After that, a minimum-jerk-based method is used to model the variations in vehicles' trajectories (paths and speed profiles). Monte Carlo simulations are then conducted to generate a large number of simulated trajectories using the proposed minimum-jerk model. Finally, in order to verify this model, the distribution of the simulated paths is compared to that of the actual extracted trajectories.

2 Related Work

2.1 Modeling of Vehicle Turning Trajectory

Several studies have been conducted in the past few decades to grasp, as possible, the turning behavior of vehicles at signalized intersections. In general, significant characteristics concerning the intersection layout and the turning vehicle were highlighted [26–29]. As an example, Alhajyaseen et al. [30] underlined that the path of the turning vehicle is strongly related to the intersection's angle, the vehicle's type, and speed. However, it is agreed that the turning maneuver of vehicles is a further complex phenomenon whose variability extends to be related to highly dynamic factors [31, 32]. For instance, the turning behavior was observed to depend on inter- and intra-subject factors concerning drivers such as the perception of traffic environment, information processing, and the ability to react correctly and to cooperate with others [33, 34]. Other factors such as the waiting time of the turning vehicles [35], the relative speed of the vehicles in conflict [36], and gaps [37] were observed to impact on the decision behavior of turning vehicles.

Since understanding the complex mechanism of turning vehicles' paths is crucial to achieve an effective traffic control and safety assessment at intersections, a reliable simulation model is required so that the variations of the turning vehicles are reproduced with high resolution. Classically, the vehicle's turning path was simulated via one-dimensional models [38–40]. In such simulations, a set of lane-based models are defined, in which the longitudinal and lateral movements are separately represented by a car-following model and a lane-changing model, respectively. Despite their simplicity and applicability to be involved in decision-making approaches, the one-dimensional models are unable to accurately reproduce the variations of turning trajectories [41].

As an alternative to the traditional one-dimensional trajectory models, the two-dimensional model has emerged as a viable simulation technique for vehicle turning paths as it breaks the lane-based assumption. Accordingly, the longitudinal and lateral movements are simultaneously simulated, and therefore, the characteristics of the turning paths are

reliably reflected [42]. However, these approaches produce the path of the turning vehicles only without the speed and acceleration profiles [30], which are usually estimated using other independent models. In this context, a microscopic simulation model that generates vehicles' turning trajectories was developed by Tan et al. [43] using different models for path (Euler spiral-based approximation method) and speed profiles. More recently, Wei et al. [44] established a left-turning vehicle's path model by means of extracting trajectories from recorded videos and analyzing their distributions, velocities, and flow-changing characteristics. On the basis of this effort, the same authors [44] proposed the idea of setting left-turning guidelines at signalized intersections, which was verified as an effective tool to reduce traffic conflicts and improve traffic efficiency. Also, Ma et al. [45] proposed a three-phased (i.e., plan-decision-action) model to estimate the vehicle's path at mixed-flow intersections. However, combining different models for turning path and speed profile does not ensure the spatial and temporal consistency between the location and the speed of turning vehicles. In an attempt to overcome such limitation, Dias et al. [7, 46] applied the concept of minimum jerk to fit the trajectories of manually tracked free-flow turning vehicles at signalized intersections in Japan. The proposed approach simultaneously estimates the temporal and the spatial profiles of the vehicle turning maneuvers with acceptable accuracy. Yet, the same authors [7, 46] did not discuss the limitations of their proposed approach and the procedure to generate the maneuvers of turning vehicles in microsimulation.

2.2 Automatic Trajectory Extraction

As an alternative to manual trajectory extraction, researchers have proposed several methods for semiautomatic and automatic tracking of the turning vehicles. For instance, Shirazi and Morris [47] proposed a semiautomatic technique to extract vehicles' trajectories from traffic footages. This method requires first to identify the locations of the vehicles in each video frame manually. After that, vehicle tracking is performed using a detection-track mapping matrix which utilizes nearest global matching. Yet, despite them producing accurate results, semiautomatic techniques are considered laborious since they initially require some steps to be performed manually [48–50].

Automatic extraction techniques, on the other hand, are deemed more promising since they provide swift results with minimum manpower involved. In this context, Hsieh et al. [11] proposed an automatic vehicle tracking method which implements a background subtraction technique for vehicle detection along with a Kalman filter for tracking. Similarly, Shirazi and Morris [51] used a Gaussian mixture model to detect vehicles at signalized intersections together with a Kalman filter for trajectory extraction. Apeltauer et al. [12] developed another automated method for trajectory extraction in which vehicles are detected using a two-stage classifier trained based on multi-block local binary pattern (MB-LBP) features. This method also requires applying background subtraction in order to generate the foreground mask. Also, Khan et al. [13] developed a comprehensive framework for automatic trajectory extraction of the vehicles from traffic footages acquired by unmanned aerial vehicles (UAVs). This framework involved video preprocessing and stabilization, vehicle detection and tracking, and ultimately, management of the extracted trajectories. Similar to [11, 12], Khan et al. [13] carried out the vehicles' detection using background subtraction algorithm.

3 CNNs and R-CNNs

3.1 Convolutional Neural Networks (CNNs)

CNNs are deep, biologically inspired feed-forward artificial neural networks (ANNs) which have been developed based on a core model for mammalian visual cortex. One of the most attractive features of CNNs is their ability to classify images regardless of their scale and orientation [52]. A typical CNN designed to deal with 28×28 pixel RGB images is depicted in Fig. 1. The structure consists of alternating *convolutional and sub-sampling layers* followed by a number of

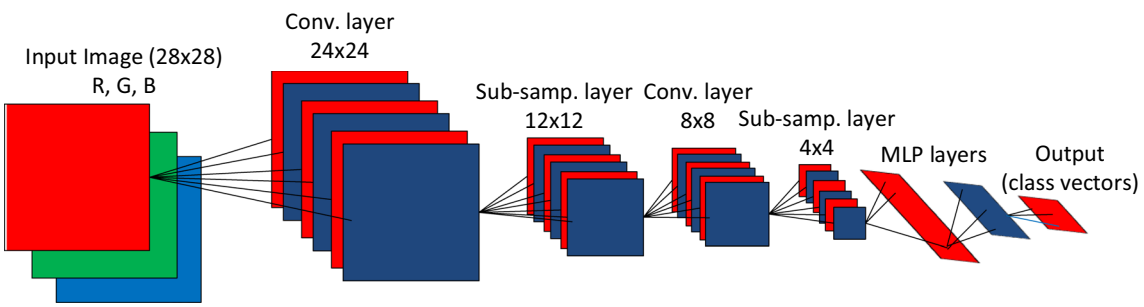


Fig. 1 A typical CNN [52]

multilayer perceptrons (MLP) layers (i.e., fully connected layers). Each convolutional layer contains a number of filters (neurons) having a specific *kernel size* ($k_x = k_y = 5$ in this illustration). These filters are responsible for extracting particular features from the input image called the *feature maps*. The convolutional filters are basically matrices of size (k_x, k_y) containing certain values referred to as the *weights*. At each neuron, 2D convolution is performed between the input image and the filter's weights. The output of this operation is processed by an *activation function* and then passed to the next subsampling layer, which decimates the feature maps extracted by the previous convolutional layer by a predefined *sub-sampling factor*. As shown in Fig. 1, after being processed by a sufficient number of convolutional and subsampling layers, the input image is reduced to a 1D array. This array is then processed by the MLP layers resulting in an output vector that represents the classification results.

The process of computing the weights of the convolutional filters and MLP layers is defined as CNN training. Before carrying out the training process, it is necessary to define the CNN's structure in advance. This includes the number of convolutional and MLP layers as well as the kernel size and subsampling factor at each level and the number of neurons in each layer of the CNN. Such hyperparameters are usually picked by trial and error since there is no systematic approach for determining the optimal CNN structure within an acceptable computational time [16]. Afterward, the weights in both convolutional and MLP layers are initialized randomly. A large set of images is then used to train the CNN according to a *back-propagation (BP) algorithm*. The objective of this training process is to adjust the CNN weights in an iterative manner until the summation of squared error between the target values and the CNN output is minimized. The BP operation is not explained in this paper for brevity. The interested reader is referred to [53] for more details about training CNNs.

Instead of training a new CNN starting from randomly generated weights, researchers often apply a technique called *transfer learning* in which a pretrained CNN is fine-tuned to learn a new task. Networks such as AlexNet [54] and GoogLeNet [55] are commonly used as a starting point in deep learning applications. Previous studies have shown that this approach is faster and more efficient than training CNNs from scratch [56].

3.2 Regions with CNNs (R-CNNs)

It must be noted here that CNNs are only designed to classify the input image into a number of predefined classes without being able to detect and localize specific objects within the image. Therefore, CNNs alone cannot satisfy the main requirement of this study, which is to detect and track vehicles automatically. To bridge the gap between image classification and object segmentation, Girshick et al. [57] have proposed a method called regions with CNNs (R-CNN). As shown in Fig. 2, this method consists of three components: (1) a region proposal algorithm that generates a large number of candidate detections, (2) a large CNN that extracts features from each proposal, and (3) linear support vector machines (SVMs) to process the extracted features and classify each candidate regions.

3.3 Data Collection and CNN Training

The south approach of Lekhwair signalized intersection located in Doha City, State of Qatar, was videotaped for a duration of two and a half hours. The video was recorded at a frame rate of 30 fps and a resolution of 3840×2160 pixel. The same video was used in the current study for both R-CNN training and trajectory extraction operations.

The images required for carrying out the training of the R-CNN were acquired by randomly selecting 26 frames of the video. To reduce the computational time required for the training, the images were cropped to the region around the middle of the intersection and the resolution was reduced to 1920×1080 pixel. For each image, vehicles were manually labeled by bounding boxes, and consequently, a dataset for the coordinates of 536 vehicles in total was obtained. Based on the images and the bounding boxes dataset, the R-CNN training process was carried out using "trainRCNNObjectDetector" function available in MATLAB Computer Vision System Toolbox. AlexNet was used as a starting point for this deep learning task. Details about the architecture of this network can be found in [54].

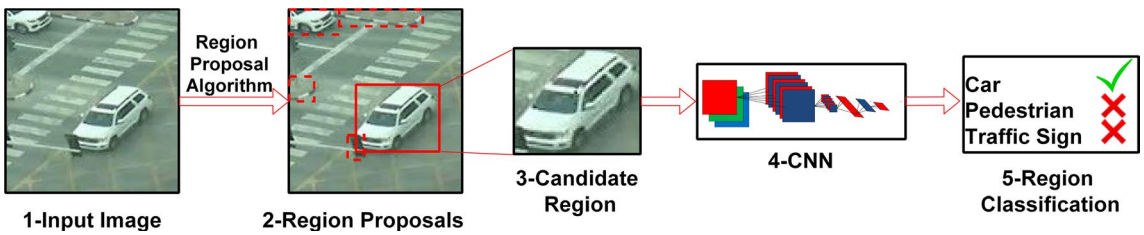


Fig. 2 Object detection and localization using R-CNN

4 Trajectory Extraction Using R-CNN

A MATLAB tool was developed to utilize the R-CNN trained in Sect. 3.3 for tracking of vehicles. This tool processes the video frames at a user-defined sampling rate and uses the R-CNN to detect the vehicles. The output of the R-CNN is a set of bounding boxes enclosing each vehicle in the processed frame. The location of a vehicle was defined here as the centroid of its bounding box. Once a vehicle is detected, the tool constructs a Kalman filter to track the location of this vehicle in the next frames until it leaves the intersection area. Using Kalman filters for tracking the vehicles is necessary to reduce trajectory noise and enable the tool to associate multiple vehicles to their correct tracks. The tool operates in two modes (Fig. 3): (1) tracking of multiple vehicles and (2) tracking of a single vehicle. The first mode allows the user to simultaneously track all moving vehicle in the video (Fig. 3a), while the second mode involves tracking a single vehicle picked by the user (Fig. 3b). The advantage of the second mode is the fact that it requires significantly lower computational time and effort compared to the first mode since the R-CNN only processes the region surrounding the vehicle of interest. The vehicle tracking process explained in the current section is illustrated in Fig. 4.

4.1 Transformation from Image to Real-World Coordinates

The trajectories generated by the aforementioned approach describe the locations of moving vehicles in image coordinates (pixels) with respect to time. In order to map the trajectories to the real-world coordinates, it is necessary to define the homography matrix corresponding to this transformation. To do so, it is required to have four known points in both real-world and image coordinates. Then, the homography matrix $\mathbf{H}$ can be computed as follows:

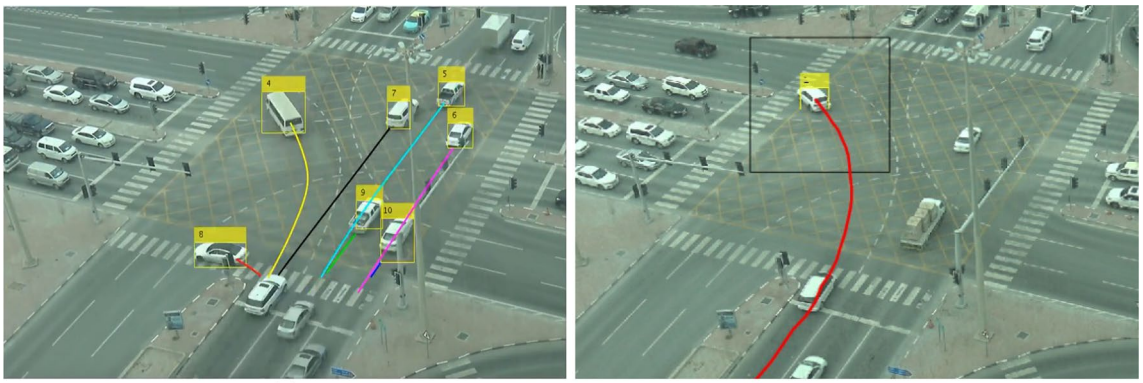
$$\mathbf{H} = \begin{bmatrix} -x_1 & -y_1 & -1 & 0 & 0 & 0 & x_1x_{w,1} & y_1x_{w,1} & x_{w,1} \\ 0 & 0 & 0 & -x_1 & -y_1 & -1 & x_1y_{w,1} & y_1y_{w,1} & y_{w,1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -x_4 & -y_4 & -1 & 0 & 0 & 0 & x_4x_{w,4} & y_4x_{w,4} & x_{w,4} \\ 0 & 0 & 0 & -x_4 & -y_4 & -1 & x_4y_{w,4} & y_4y_{w,4} & y_{w,4} \end{bmatrix} \quad (1)$$

where $(x_{w,1}, y_{w,1}), \ldots, (x_{w,4}, y_{w,4})$ are the real-world coordinates of any four noncolinear points and $(x_1, y_1), \ldots, (x_4, y_4)$ are the image coordinates (in pixels) of the same four points. The matrix $\mathbf{H}$ can be then calculated as the first eigenvector (reshaped into a 3×3 matrix) of $\mathbf{H^T H}$. Next, the homography matrix can be used to map any point from image coordinates (x_i, y_i) to world coordinates (x_w, y_w) as follows:

$$\begin{bmatrix} c * x_w & c * y_w & c \end{bmatrix}^{\mathrm{T}} = \mathbf{H} \begin{bmatrix} x_i & y_i & 1 \end{bmatrix}^{\mathrm{T}} \quad (2)$$

4.2 Verification of the Proposed CNN Tool

In order to verify its accuracy, the proposed tool was used to extract the trajectories of 18 free-flowing left-turning vehicles. The same vehicles were also tracked manually in order to identify the ground truth of the trajectories. The manual trajectory extraction was performed simply by tracking each vehicle at a 0.5-s rate, while drawing a bounding box around the vehicle in each frame. The location of a manually tracked vehicle position at a certain time was taken as the centroid of the bounding box. The manual trajectories were then transformed into real-world coordinates as explained in Sect. 4.1. A comparison between the automatically and



(a) (b)

Fig. 3 Automatic tracking of vehicles using the proposed tool. **a** Tracking using the first mode (i.e., tracking of multiple vehicles). **b** Tracking using the second mode (i.e., tracking of a single vehicle). Note the black box surrounding the tracked car which represents the R-CNN's region of interest

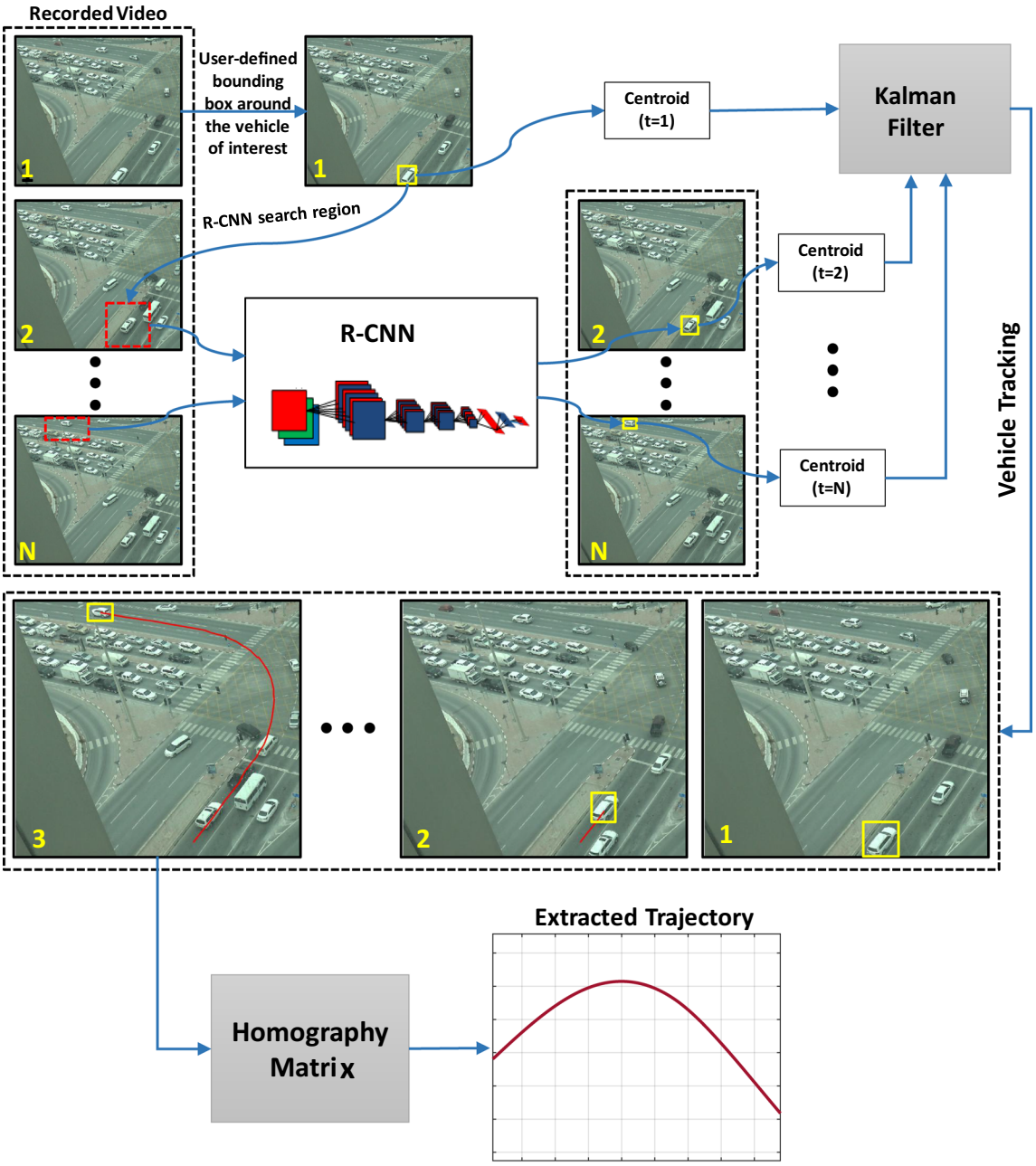


Fig. 4 Scheme for the proposed vehicle tracking approach (Mode 2: tracking of a single vehicle)

manually extracted trajectories is shown in Fig. 5. The error between the trajectories extracted by the proposed tool and their manually extracted counterparts was calculated at each point of the trajectories. The error E_p was defined here as the distance between an automatically extracted point and the corresponding manually extracted one according to the following equation:

$$E_p = \sqrt{\left(x_a - x_m\right)^2 + \left(y_a - y_m\right)^2} \tag{3}$$

where x_a and y_a are the coordinates of the automatically extracted point and x_m and y_m the associated manually extracted point. The error distribution of the points corresponding to the 18 trajectories is shown in Fig. 6. The average error across all points of the trajectories is 16.5 cm with a standard deviation 11.8 cm. These results support the ability of the proposed tool to automatically track vehicles with an acceptable level of accuracy.

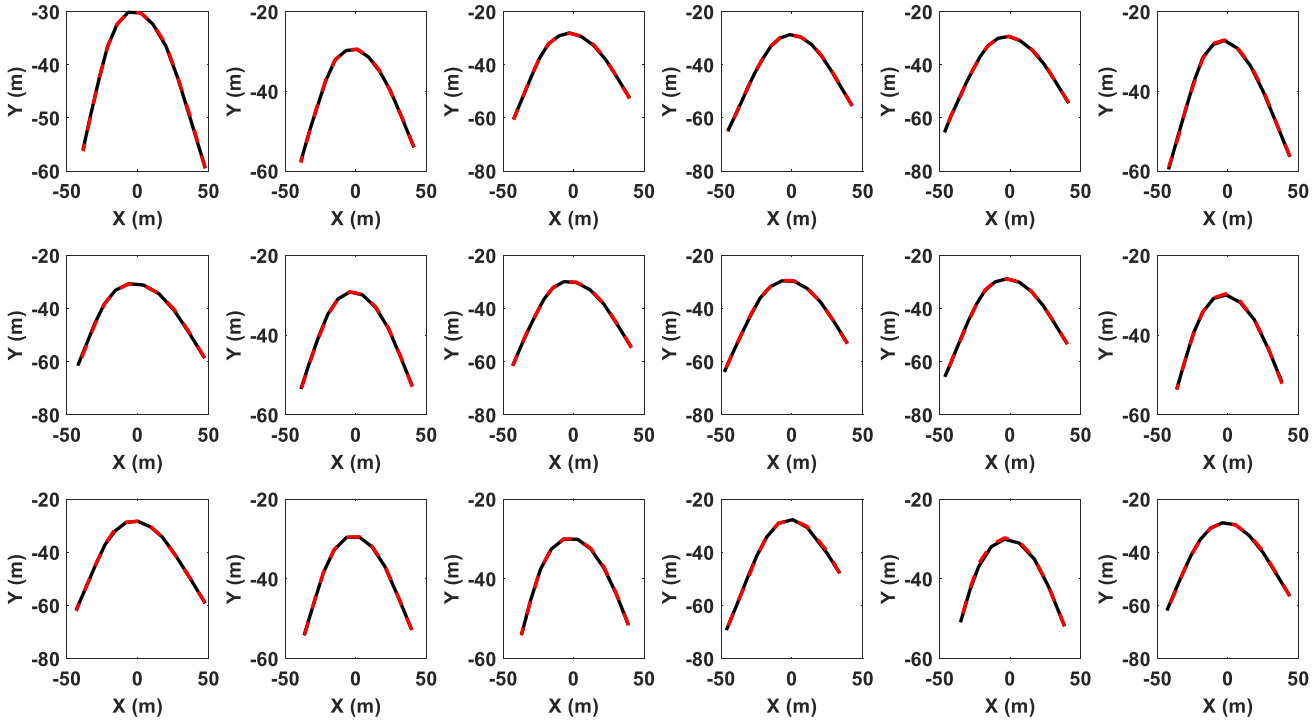


Fig. 5 Comparison between the manually and automatically extracted trajectories in real-world coordinates. The solid black line represents a manually extracted trajectory, while the dashed red line denotes an automatically extracted one

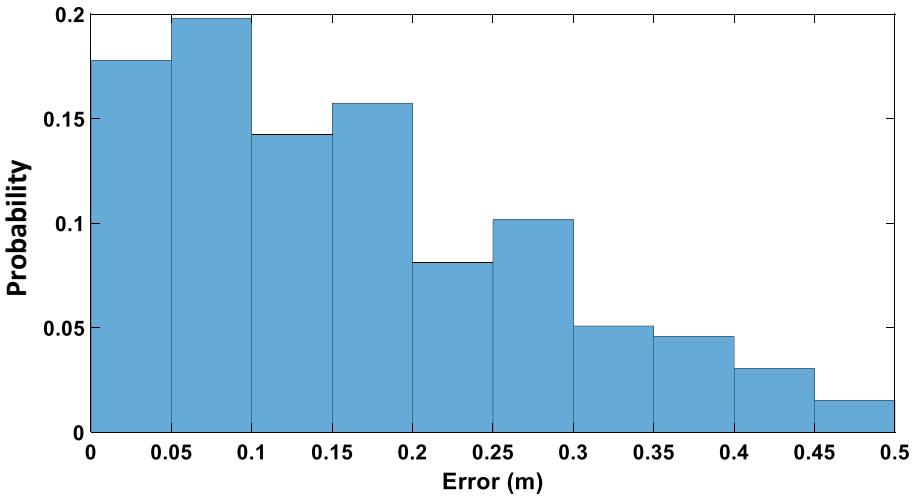


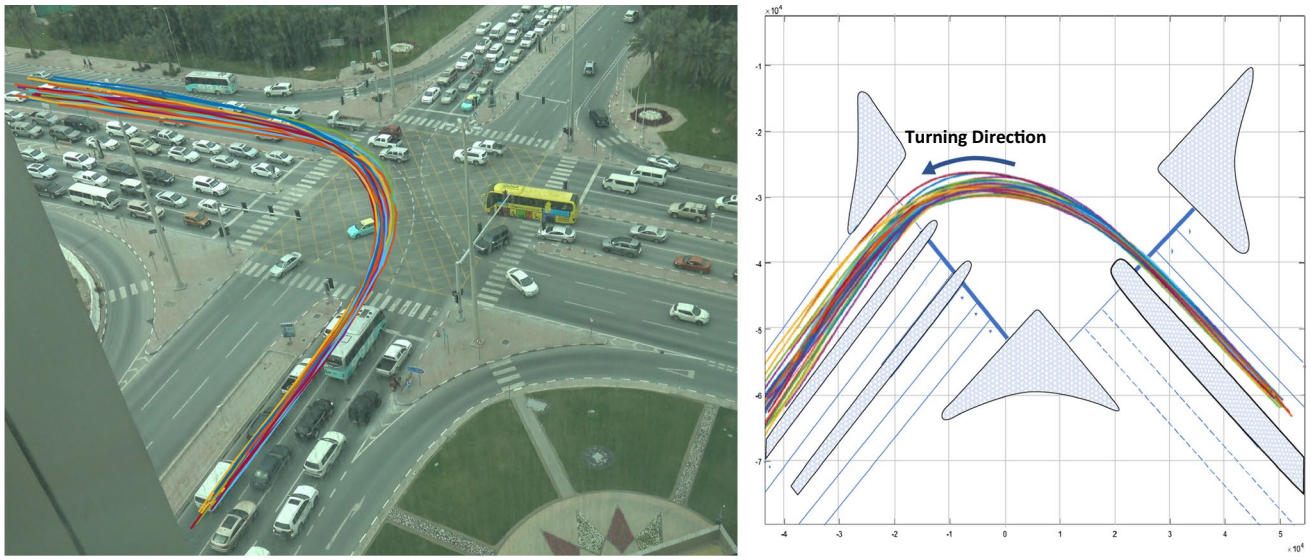
Fig. 6 The distribution of the error between the manually and automatically extracted points across the 18 trajectories

5 Trajectory Analyses

The proposed CNN-based tool was used to automatically extract the trajectories of 44 left-turning free-flowing vehicles (i.e., unimpeded by traffic/pedestrians) from the recorded video. The extracted trajectories are shown in image coordinates (in Fig. 7a) as well as in real-world coordinates (Fig. 7b).

5.1 Minimum-Jerk Method

Originally, the principle of minimum jerk was proposed in the mid-1980s by Flash and Hogan [58] to describe the planar movements of the human arm, after which the use of this method has gained more popularity and general acceptance. Successful applications of the minimum-jerk method have been reported in various contexts including human



(a) Image coordinates **(b)** Real-world coordinates

Fig. 7 The automatically extracted free-flowing left-turning trajectories at the south approach of Lekhwair signalized intersection

goal-oriented locomotion [59], robot-limb movements [60], autonomous vehicle maneuvers [61, 62], driver-following behavior [63], and more recently a preliminary application for modeling the trajectory of turning vehicles [7, 46].

In principle, the minimum-jerk model suggests that the drivers tend to optimize the smoothness of turning maneuvers by minimizing the time integration of the jerk. Thus, according to [58], the cost function to be minimized can be given as:

$$J = {}^1\!/_2 \int_0^{t_f} \left(\left(\frac{\mathrm{d}^3 x}{\mathrm{d}t^3} \right)^2 + \left(\frac{\mathrm{d}^3 y}{\mathrm{d}t^3} \right)^2 \right) \mathrm{d}t \tag{4}$$

where t_f is the time elapsed by the turning vehicle to cross the intersection.

As indicated by Flash and Hogan [58], the solution of the minimization problem given in Eq. (4) can be obtained in the time domain as a set of fifth-order polynomials for x and y as follows:

$$x(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \tag{5}$$

$$y(t) = b_0 + b_1 t + b_2 t^2 + b_3 t^3 + b_4 t^4 + b_5 t^5 \tag{6}$$

where a_i and b_i $(i = 0, 1, \ldots, 5)$ are unknown coefficients to be obtained using twelve boundary conditions corresponding to the x- and y-components of location, velocity, and acceleration at the initial and final points of the vehicle's trajectory. By applying the boundary conditions corresponding to the x-direction on Eq. (5), the following system of equations can be obtained:

$$x_0 = a_0 \tag{7a}$$

$$v_{x0} = a_1 \tag{7b}$$

$$a_{x0} = 2a_2 \tag{7c}$$

$$x_f = a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 + a_4 t_f^4 + a_5 t_f^5 \tag{7d}$$

$$v_{xf} = a_1 + 2a_2 t_f + 3a_3 t_f^2 + 4a_4 t_f^3 + 5a_5 t_f^4 \tag{7e}$$

$$a_{xf} = 2a_2 + 6a_3 t_f + 12a_4 t_f^2 + 20a_5 t_f^3 \tag{7f}$$

where x_0, v_{x0}, and a_{x0} are the position, velocity, and acceleration, respectively, in the x-direction at the starting point of the turning maneuver, and x_f, v_{xf}, and a_{xf} are those corresponding to the end point. Likewise, applying the boundary conditions corresponding to the y-direction yields a system of equations similar to that of Eq. (7) but in terms of the coefficients $b_i (i = 0, 1, \ldots, 5)$ and the parameters y_0, v_{y0}, a_{y0}, y_f, v_{yf}, and a_{yf}.

5.2 Identification of the Starting and Ending Points of the Turning Maneuver

In order to compute the coefficients a_i and $b_i (i = 0, 1, \ldots, 5)$ corresponding to each of the extracted trajectories, it is necessary to identify the x- and y-components of position, velocity, and acceleration at the points at which the vehicle starts and ends its turning maneuver, along with the time

taken during the maneuver t_f. Once these values are known, the two systems of equations (described in Sect. 5.1) can be easily solved to obtain the coefficients a_i and b_i corresponding to the trajectory's turning curve.

Therefore, it is necessary to accurately identify the two points along the trajectory at which the turning maneuver starts and ends. To do so, we utilize the spline-fitting method presented in [30]. According to this method, the trajectory of a left-turning vehicle at a signalized intersection can be represented by a spline consisting of five segments. The spline starts with a straight line followed by an Euler spiral having a curvature profile that varies almost linearly with a gradient of $1/A_1^2$. This spiral is followed by a circular segment with a curvature of $1/R_{\min}$. The end of the spline consists of another Euler spiral having a nearly linear curvature profile with a gradient of $-1/A_2^2$ followed by a straight line. As shown in Fig. 8, there are four main locations that define the beginning and the end of each Euler spiral and circular segments. These locations are basically the points of discontinuity along the curvature profile of the vehicle's path. The points of interest here are points 1 and 4 in Fig. 8, which represent the starting and ending points of the turning maneuver.

A MATLAB code was written to fit the aforementioned spline to each of the extracted trajectories in order to identify the four points of curvature discontinuity along the vehicles' turning paths. The code applies the nonlinear programming solver "fmincon" available in MATLAB Optimization Toolbox to compute the optimal location of the four key points (described in Fig. 8) so that the error between the tracked path and the fitted spline is minimized. Four constraints were imposed to enforce continuity of the fitted spline at the four points. Also, another four constraints were applied to make sure that there are no sudden jumps in the curvature profile at the key points. The fitting of the two Euler spirals was conducted according to the approach proposed in [64]. Figure 9 displays four examples of splines fitted to their corresponding automatically extracted paths.

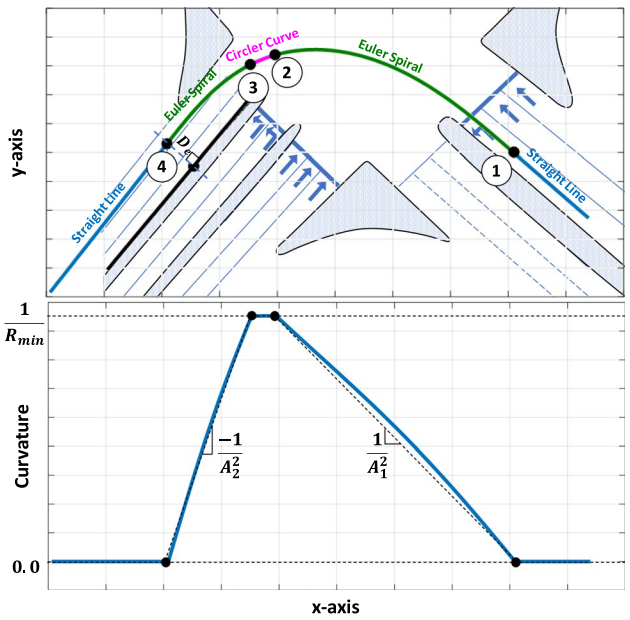


Fig. 8 Components of the spline used for trajectory curve fitting

5.3 Statistical Analysis

After identifying the two points of interest along each of the extracted trajectories (as explained in Sect. 5.2), the parameters x_0, $v_{x,0}$, $a_{x,0}$, x_f, $v_{x,f}$, $a_{x,f}$, y_0, $v_{y,0}$, $a_{y,0}$, y_f, $v_{y,f}$, $a_{y,f}$, along with t_f, were computed for each trajectory. Figure 10 displays the probability distributions for these 13 parameters. As shown in the figure, a normal distribution was fitted for each parameter. One-sample Kolmogorov–Smirnov test (95% confidence level) indicated that each of the 13 parameters comes from a normal distribution with the mean and standard deviation shown in Table 1.

5.4 Comparison Between Simulated and Observed Trajectories

Monte Carlo simulation with 500 trials was conducted using the developed models. In each trial, random values of x_0, $v_{x,0}$, $a_{x,0}$, x_f, $v_{x,f}$, $a_{x,f}$, y_0, $v_{y,0}$, $a_{y,0}$, y_f, $v_{y,f}$, $a_{y,f}$, and t_f were

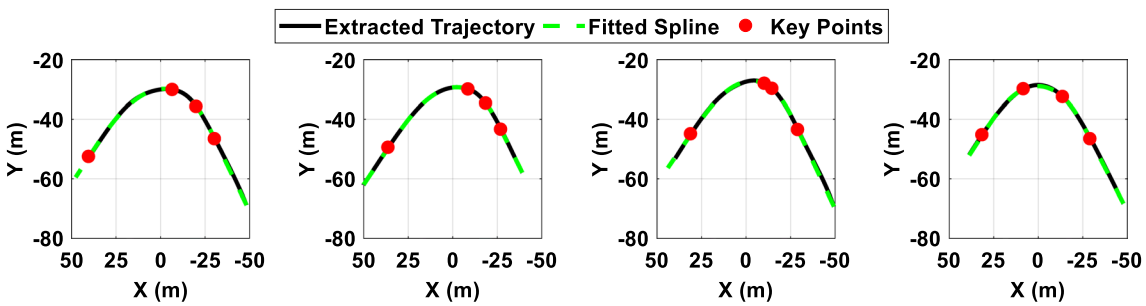


Fig. 9 Examples of the curve fitting process showing the fitted splines and their curvature profiles along with the speed profiles

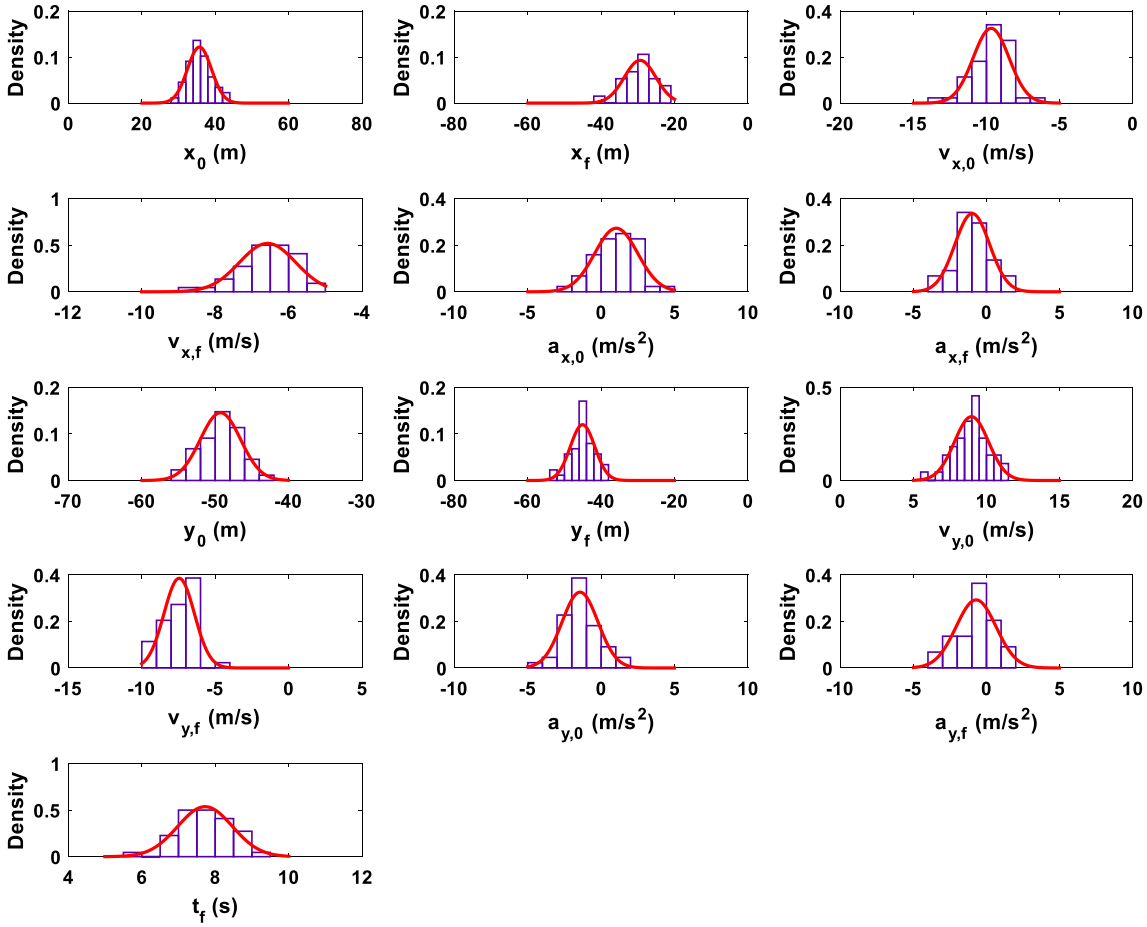


Fig. 10 Probability distribution of the 13 extracted parameters across the extracted trajectories

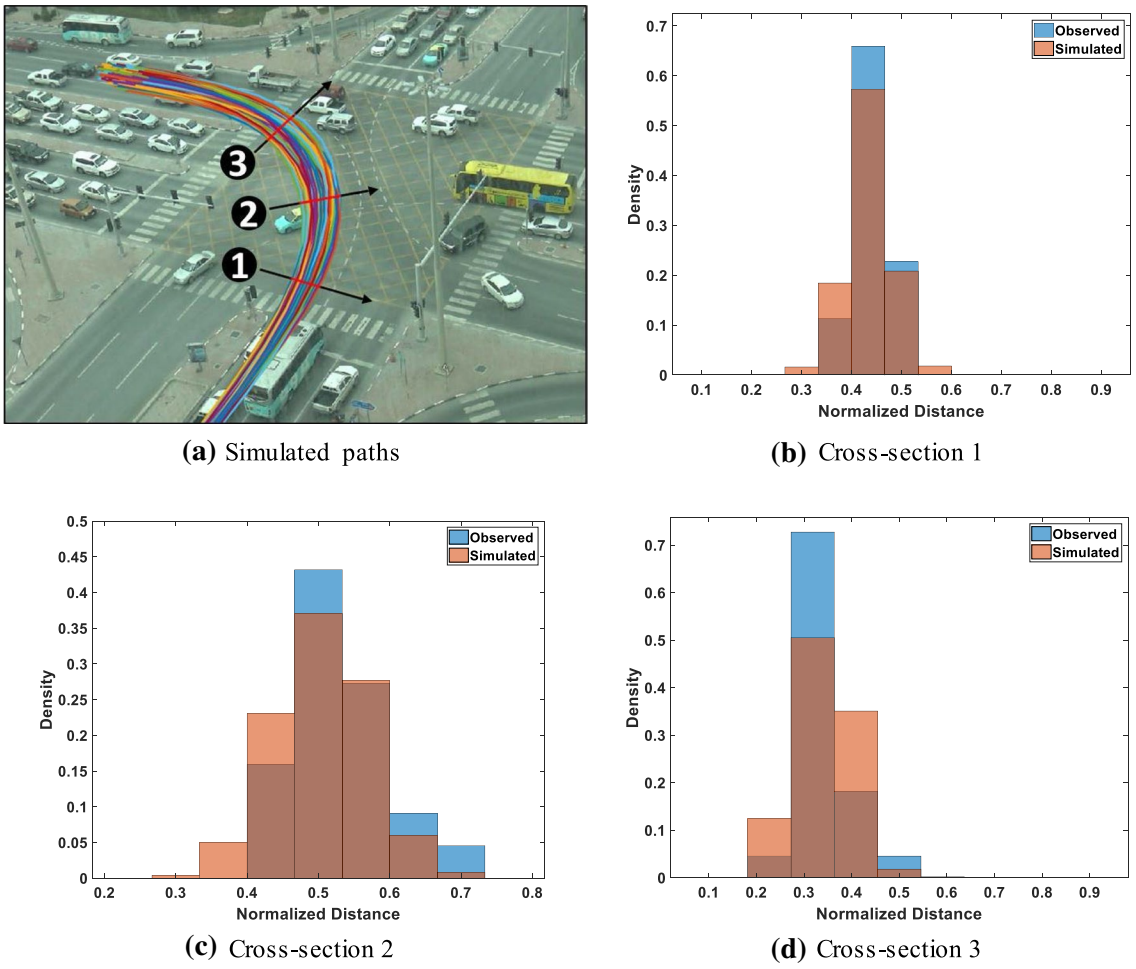
Table 1 Mean, standard deviation, and coefficient of variation of the parameters' distribution

	x_0 (m)	x_f (m)	$v_{x,0}$ (m/s)	$v_{x,f}$ (m/s)	$a_{x,0}$ (m/s^2)	$a_{x,0}$ (m/s^2)	x_0 (m)	x_f (m)	$v_{x,0}$ (m/s)	$v_{x,f}$ (m/s)	$a_{x,0}$ (m/s^2)	$a_{x,0}$ (m/s^2)	t_f (s)
μ	35.7	−29.4	−9.68	−6.57	1.03	−0.971	−49.3	−45.06	8.98	−7.45	−1.44	−0.708	7.72
σ	3.27	4.29	1.22	0.77	1.46	1.17	2.74	3.33	1.16	1.04	1.22	1.36	0.74

generated according to the normal distributions described in Fig. 10 and Table 1. The resulting parameters were then used to compute the coefficients which determine the shape of the trajectory (a_i and b_i) by solving the two systems of equations described in Sect. 5.1. The coefficients were then used as per Eqs. (5) and (6) to obtain the simulated paths shown in Fig. 11a.

The distributions of the simulated paths, and those of the observed/actual trajectories, were analyzed and compared along three selected cross sections (drawn in Fig. 11a). Figure 11b–d depicts a comparison between the observed and the simulated distributions. Kolmogorov–Smirnov test (performed at 95% confidence level) revealed that the simulated distributions at the three cross sections are not significantly different from the actual/observed counterparts. Furthermore, the comparison shown in Fig. 12 indicates a reasonable agreement between the observed and simulated speed and acceleration profiles, which supports the reliability of the proposed model in generating accurate and realistic vehicle turning maneuvers.

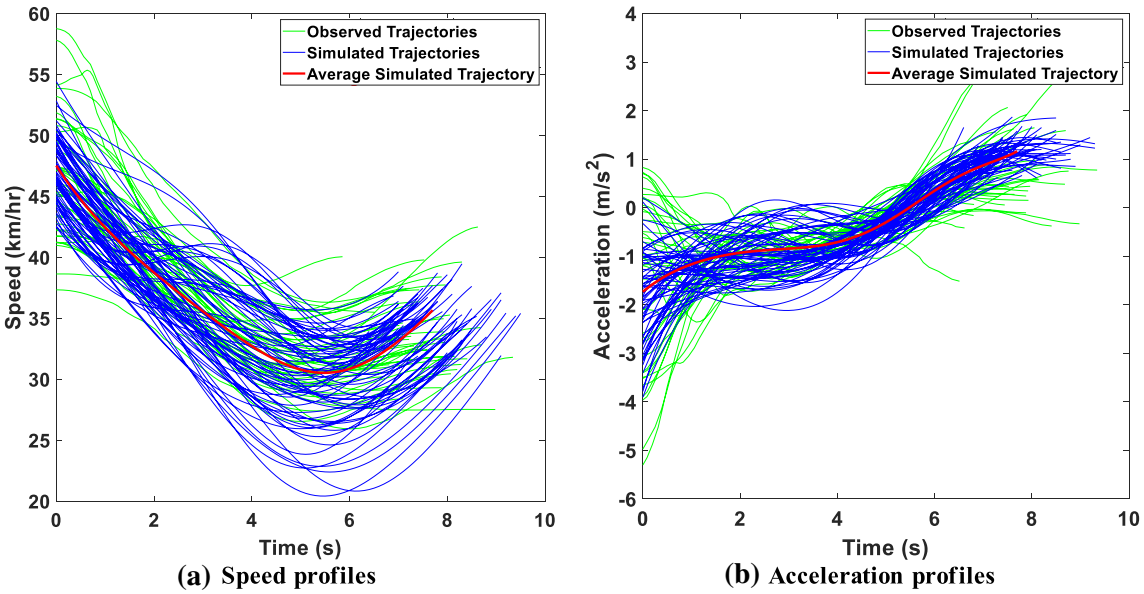
Finally, Fig. 13 provides a concise summary of the overall procedure followed to develop and validate the proposed minimum-jerk-based trajectory model, starting from trajectory extraction and ending with the use of Monte Carlo simulations to generate trajectories of the turning vehicles.



(a) Simulated paths (b) Cross-section 1

(c) Cross-section 2 (d) Cross-section 3

Fig. 11 Comparison between observed and simulated distributions at different cross sections



(a) Speed profiles (b) Acceleration profiles

Fig. 12 Comparison between observed and simulated speed and acceleration profiles

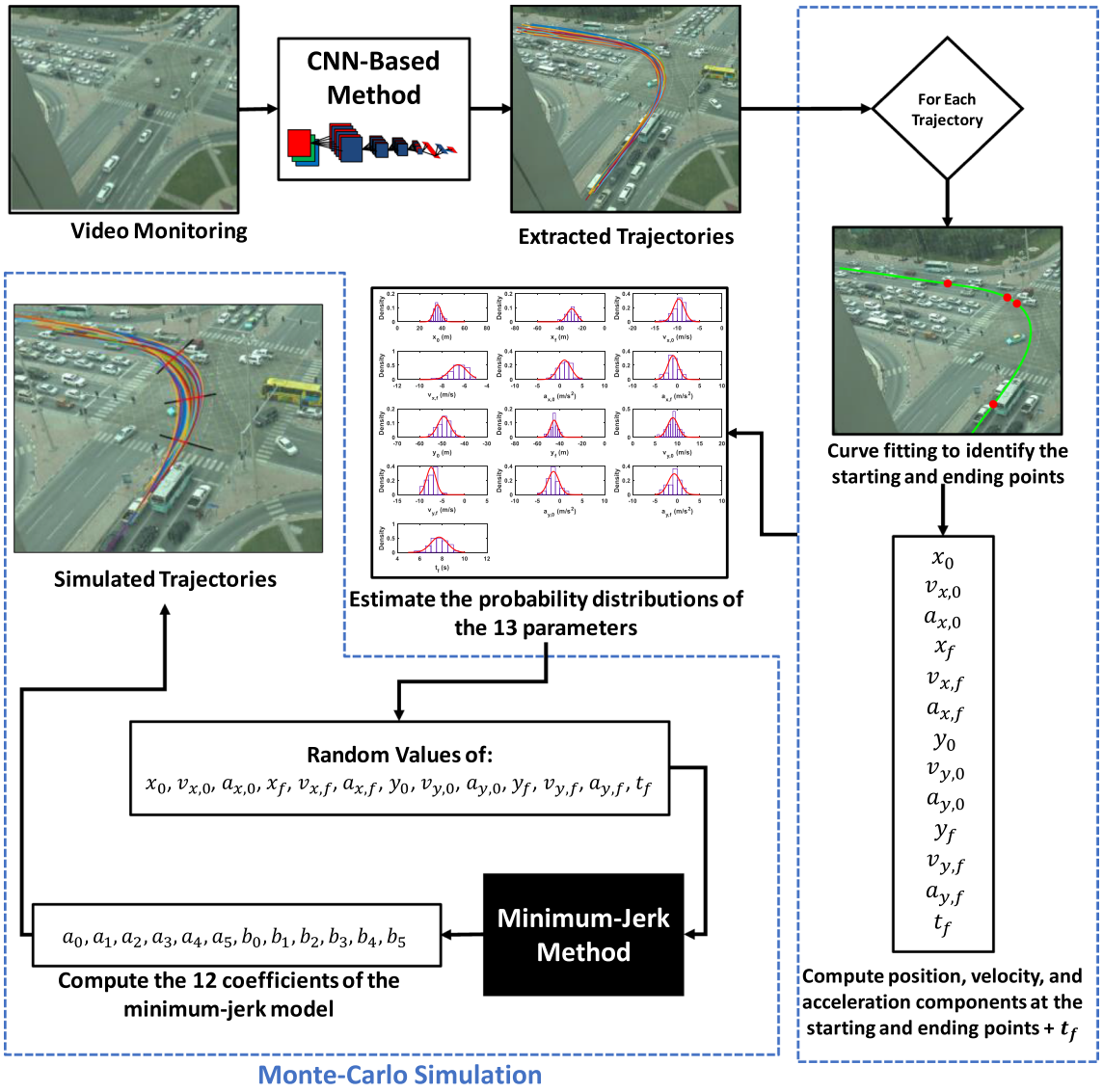


Fig. 13 Flowchart for the overall procedure followed in the trajectory analyses

6 Conclusions and Future Recommendations

In this paper, a CNN-based tool was developed for the automatic extraction of vehicle trajectories. In order to test the proposed tool, video data were collected at a signalized intersection located in Doha City, State of Qatar. Several trajectories were extracted both manually and automatically. The average error between the manually and automatically extracted trajectory paths was 16.5 cm, which demonstrates the accuracy of the proposed method. A minimum-jerk-based approach was used to statistically model the variations in left-turning vehicle trajectories including paths and speed profiles. The minimum-jerk approach was found to be effective and reliable in producing realistic turning maneuvers. Monte Carlo simulation was conducted to verify the statistical model by comparing the simulated and actual trajectories.

Finally, the effort presented in this paper can be regarded as a step forward toward maximizing the potential use of deep learning in traffic safety applications. However, in order to further improve the applicability of the proposed methods, the following recommendations can be considered in future studies:

- The R-CNN used in this work was trained using images taken from a single intersection with specific geometric characteristics and surrounding conditions. Therefore, the proposed R-CNN can only be used to accurately track vehicles at this particular intersection. Training the net-

work using a larger set of images that are collected from multiple intersections is required to generate a more versatile R-CNN.

- The computational efficiency of the proposed tool can be improved by optimizing the structure of the R-CNN. Also, the updated versions of the standard R-CNN used in this work (i.e., fast R-CNN [65] or faster R-CNN [66]) can be implemented to minimize the required computational time.
- The trajectory models developed in this study are based on a limited number of trajectories ($N = 44$) extracted from a single intersection. A larger number of trajectories obtained from several intersections are essential to achieve a deeper insight into the behavior of drivers at signalized intersections. Furthermore, the proposed trajectory model assumes that the start and end points of the turning trajectory are known; accordingly, it is required to develop probabilistic models to estimate the distribution of these points (i.e., start and end of turning path) as functions of the vehicle entry speed and intersection geometry.

Acknowledgements Open access funding provided by Linnaeus University. This publication was made possible by the NPRP award [NPRP 9-360-2-150] from Qatar National Research Fund (a member of The Qatar Foundation). The statements made herein are solely the responsibility of the authors. Special thanks are due to Dr. Deepti Muley for the support in collecting the video records used in the current paper.

References

1. Gao, J.; Chen, X.; Woodward, A.; Liu, X.; Wu, H.; Lu, Y.; Li, L.; Liu, Q.: The association between meteorological factors and road traffic injuries: a case analysis from Shantou city, China. Sci. Rep. **6**, 37300 (2016). https://doi.org/10.1038/srep37300
2. Zhang, G.; Qi, Y.; Chen, J.: Exploring factors impacting paths of left-turning vehicles from minor road approach at unsignalized intersections. Math. Probl. Eng. **2016**, 1305890 (2016). https://doi.org/10.1155/2016/1305890
3. Ma, W.; Yang, X.: Coordination design of left movements of signalized intersections group. J. Tongji Univ. **36**, 1507–1511 (2008)
4. Liu, P.; Xu, C.; Wang, W.; Wan, J.: Identifying factors affecting drivers' selection of unconventional outside left-turn lanes at signallised intersections. IET Intell. Transp. Syst. **7**, 396–403 (2013). https://doi.org/10.1049/iet-its.2011.0229
5. Sarvi, M.; Young, W.; Sobhani, A.; Lenn, M.G.: Simulation of safety: a review of the state of the art in road safety. Simul. Model. **66**, 89–103 (2014). https://doi.org/10.1016/j.aap.2014.01.008
6. Alhajyaseen, W.K.M.; Asano, M.; Nakamura, H.: Estimation of left-turning vehicle maneuvers for the assessment of pedestrian safety at intersections. IATSS Res. **36**, 66–74 (2012). https://doi.org/10.1016/j.iatssr.2012.03.002
7. Dias, C.; Iryo-asano, M.; Oguchi, T.: Concurrent prediction of location, velocity and acceleration profiles for left turning vehicles at signalized intersections. In: 土木計画学研究発表会･講演集, pp. 3054–3062 (2016)
8. Yu, Y.; El Kamel, A.; Gong, G.; Li, F.: Multi-agent based modeling and simulation of microscopic traffic in virtual reality system. Simul. Model. Pract. Theory **45**, 62–79 (2014). https://doi.org/10.1016/j.simpat.2014.04.001
9. Meuleners, L.; Fraser, M.: A validation study of driving errors using a driving simulator. Transp. Res. Part F **29**, 14–21 (2015). https://doi.org/10.1016/j.trf.2014.11.009
10. Helmer, T.; Wang, L.; Kompass, K.;, Kates, R.: Safety performance assessment of assisted and automated driving by virtual experiments: stochastic microscopic traffic simulation as knowledge synthesis. In: IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC, pp. 2019–2023 (2015)
11. Hsieh, J.W.; Yu, S.H.; Chen, Y.S.; Hu, W.F.: Automatic traffic surveillance system for vehicle tracking and classification. IEEE Intell. Transp. Syst. Mag. **7**, 175–187 (2006). https://doi.org/10.1109/TITS.2006.874722
12. Apeltauer, J.; Babinec, A.; Herman, D.; Apeltauer, T.: Automatic vehicle trajectory extraction for traffic analysis from aerial video data. In: International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science—ISPRS Arch. 40, 9–15 (2015). https://doi.org/10.5194/isprsarchives-XL-3-W2-9-2015
13. Khan, M.A.; Ectors, W.; Bellemans, T.; Janssens, D.; Wets, G.: Unmanned aerial vehicle-based traffic analysis: methodological framework for automated multivehicle trajectory extraction. Transp. Res. Rec. J. Transp. Res. Board. **2626**, 25–33 (2017). https://doi.org/10.3141/2626-04
14. Fleuret, F.; Berclaz, J.; Lengagne, R.; Fua, P.: Multicamera people tracking with a probabilistic occupancy map. IEEE Trans. Pattern Anal. Mach. Intell. **30**, 267–282 (2008). https://doi.org/10.1109/TPAMI.2007.1174
15. Ma, X.; Dai, Z.; He, Z.; Ma, J.; Wang, Y.; Wang, Y.: Learning traffic as images: a deep convolutional neural network for large-scale transportation network speed prediction. Sensors (Switzerland) **17**, 818 (2017). https://doi.org/10.3390/s17040818
16. Chung, J.; Sohn, K.: Image-based learning to measure traffic density using a deep convolutional neural network. IEEE Trans. Intell. Transp. Syst. (2017). https://doi.org/10.1109/TITS.2017.2732029
17. Tang, T.; Zhou, S.; Deng, Z.; Zou, H.; Lei, L.: Vehicle detection in aerial images based on region convolutional neural networks and hard negative example mining. Sensors (Switzerland) (2017). https://doi.org/10.3390/s17020336
18. Gopalakrishnan, K.; Khaitan, S.K.; Choudhary, A.; Agrawal, A.: Deep convolutional neural networks with transfer learning for computer vision-based data-driven pavement distress detection. Constr. Build. Mater. **157**, 322–330 (2017). https://doi.org/10.1016/j.conbuildmat.2017.09.110
19. Zhang, L.; Yang, F.; Daniel Zhang, Y.; Zhu, Y.J.: Road crack detection using deep convolutional neural network. In: 2016 IEEE International Conference on Image Processing, pp. 3708–3712 (2016). https://doi.org/10.1109/ICIP.2016.7533052
20. Qian, R.; Zhang, B.; Yue, Y.; Wang, Z.; Coenen, F.: Robust Chinese traffic sign detection and recognition with deep convolutional

neural network. In: 2015 11th International Conference on Nature Computing, pp. 791–796 (2015). https://doi.org/10.1109/ICNC.2015.7378092
21. Lau, M.M.; Lim, K.H.; Gopalai, A.A.: Malaysia traffic sign recognition with convolutional neural network. In: 2015 IEEE International Conference on Digital Signal Processing, pp. 1006–1010 (2015). https://doi.org/10.1109/ICDSP.2015.7252029
22. Jin, J.; Fu, K.; Zhang, C.: Traffic sign recognition with hinge loss trained convolutional neural networks. IEEE Trans. Intell. Transp. Syst. **15**, 1991–2000 (2014). https://doi.org/10.1109/TITS.2014.2308281
23. Sermanet, P.; Lecun, Y.: Traffic sign recognition with multi-scale convolutional networks. In: Proceedings of International Joint Conference on Neural Networks, pp. 2809–2813 (2011). https://doi.org/10.1109/IJCNN.2011.6033589
24. Szarvas, M.; Yoshizawa, A.; Yamamoto, M.; Ogata, J.: Pedestrian detection with convolutional neural networks. In: Intelligent Vehicles Symposium 2005. Proceedings. IEEE, pp. 224–229 (2005)
25. Szarvas, M.; Sakai, U.: Jun Ogata: Real-time Pedestrian detection using LIDAR and convolutional neural networks. In: 2006 IEEE Intelligent Vehicles Symposium, pp. 213–218 (2006). https://doi.org/10.1109/IVS.2006.1689630
26. Reed, M.: Intersection kinematics: a pilot study of driver turning behavior obscuration by a-pillars. Report No. UMTRI-2008-54, University of Michigan, Ann Arbor, Industry Affiliation Program for Human Factors in Transportation Safety (2008)
27. Stover, V.G.; Koepke, F.J.: Transportation and Land Development, pp. 1–239. Prentice-Hall, Englewood Cliffs (1988)
28. Stover, V.G.: Issues relating to the geometric design of intersections. In: Proceedings of 8th International Conference on ACCESS Management (2008)
29. Sando, T.; Ph, D.; Moses, R.: Influence of intersection geometrics on the operation of triple left-turn lanes. J. Transp. Eng. **135**, 253–259 (2009). https://doi.org/10.1061/(ASCE)TE.1943-5436.0000005
30. Alhajyaseen, W.K.M.; Asano, M.; Nakamura, H.; Tan, D.M.: Stochastic approach for modeling the effects of intersection geometry on turning vehicle paths. Transp. Res. Part C Emerg. Technol. **32**, 179–192 (2013). https://doi.org/10.1016/j.trc.2012.09.006
31. Kaysi, I.A.; Abbany, A.S.: Modeling aggressive driver behavior at unsignalized intersections. Accid. Anal. Prev. **39**, 671–678 (2007). https://doi.org/10.1016/j.aap.2006.10.013
32. Gu, Y.; Hashimoto, Y.; Hsu, L.T.; Iryo-Asano, M.; Kamijo, S.: Human-like motion planning model for driving in signalized intersections. IATSS Res. **41**, 129–139 (2017). https://doi.org/10.1016/j.iatssr.2016.11.002
33. Moussa, G.; Radwan, E.; Hussain, K.: Augmented reality vehicle system: left-turn maneuver study. Transp. Res. Part C Emerg. Technol. **21**, 1–16 (2012). https://doi.org/10.1016/j.trc.2011.08.005
34. Sun, R.: Cognition and Multi-agent Interaction: From Cognitive Modeling to Social Simulation. Cambridge University Press, Cambridge (2005)
35. Alexander, J.; Barham, P.; Black, I.: Factors influencing the probability of an incident at a junction: results from an interactive driving simulator. Accid. Anal. Prev. **34**, 779–792 (2002). https://doi.org/10.1016/S0001-4575(01)00078-1
36. Liu, M.; Lu, G.; Wang, Y.; Wang, Y.; Zhang, Z.: Preempt or yield? An analysis of driver's dynamic decision making at unsignalized intersections by classification tree. Saf. Sci. **65**, 36–44 (2014). https://doi.org/10.1016/j.ssci.2013.12.009
37. Pollatschek, M.A.; Polus, A.; Livneh, M.: A decision model for gap acceptance and capacity at intersections. Transp. Res. Part B Methodol. **36**, 649–663 (2002). https://doi.org/10.1016/S0191-2615(01)00024-8
38. Hamed, M.M.; Easa, S.M.; Batayneh, R.R.: Disaggregate gap-acceptance model for unsignalized T-intersections. J. Transp. Eng. **123**, 36–42 (1997). https://doi.org/10.1061/(ASCE)0733-947X(1997)123:1(36)
39. Madanat, S.; Cassidy, M.; Wang, M.: Probabilistic delay model at stop-controlled intersection. J. Transp. Eng. ASCE **120**, 21–36 (1994). https://doi.org/10.1061/(ASCE)0733-947X(1994)120:1(21)
40. Gipps, P.G.: A model for the structure of lane-changing decisions. 8 (1986)
41. Huang, W.; Fellendorf, M.; Schönauer, R.: Social force based vehicle model for two-dimensional spaces. In: Transportation Research Board 91st Annual Meeting, pp. 1–16 (2012)
42. Xu, Y.; Ma, Z.; Sun, J.: Simulation of turning vehicles' behaviors at mixed-flow intersections based on potential field theory. Transp. B Transp. Dyn. (2018). https://doi.org/10.1080/21680566.2018.1447407
43. Tan, D.; Alhajyaseen, W.; Asano, M.; Nakamura, H.: Development of microscopic traffic simulation model for safety assessment at signalized intersections. Transp. Res. Rec. J. Transp. Res. Board **2316**, 122–131 (2012). https://doi.org/10.3141/2316-14
44. Wei, F.; Guo, W.; Liu, X.; Liang, C.; Feng, T.: Left-turning vehicle trajectory modeling and guide line setting at the intersection. Discret. Dyn. Nat. Soc. **2014**, 950219 (2014). https://doi.org/10.1155/2014/950219
45. Ma, Z.; Sun, J.; Wang, Y.: A two-dimensional simulation model for modelling turning vehicles at mixed-flow intersections. Transp. Res. Part C Emerg. Technol. **75**, 103–119 (2017). https://doi.org/10.1016/j.trc.2016.12.005
46. Dias, C.; Iryo-Asano, M.; Oguchi, T.: Predicting optimal trajectory of left-turning vehicle at signalized intersection. Transp. Res. Procedia **21**, 240–250 (2017). https://doi.org/10.1016/j.trpro.2017.03.093
47. Salvo, G.; Caruso, L.; Scordo, A.: Gap acceptance analysis in an urban intersection through a video acquired by an UAV. In: Recent Advances in CivilEngineering and Mechanics, WSEAS Press, ISSN: 2227-4588, pp. 199–205 (2014)
48. Salvo, G.; Caruso, L.; Scordo, A.: Gap acceptance analysis in an urban intersection through a video acquired by an UAV. In: Recent Advances in Civil Engineering and Mechanics, pp. 199–205. WSEAS Press (2014)
49. Salvo, G.; Caruso, L.; Scordo, A.: Urban traffic analysis through an UAV. Procedia Soc. Behav. Sci. **111**, 1083–1091 (2014). https://doi.org/10.1016/j.sbspro.2014.01.143
50. Barmpounakis, E.N.; Vlahogianni, E.I.; Golias, J.C.: Extracting kinematic characteristics from unmanned aerial vehicles. In: Transportation *Research* Board 95th Annual Meeting, p. 16 (2016)
51. Shokrolah Shirazi, M.; Morris, B.T.: Trajectory prediction of vehicles turning at intersections using deep neural networks. Mach. Vis. Appl. **30**, 1097–1109 (2019). https://doi.org/10.1007/s00138-019-01040-w
52. Abdeljaber, O.; Avci, O.; Kiranyaz, S.; Gabbouj, M.; Inman, D.J.: Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks. J. Sound Vib. **388**, 154–170 (2017). https://doi.org/10.1016/j.jsv.2016.10.043
53. Kiranyaz, S.; Ince, T.; Gabbouj, M.: Real-time patient-specific ECG classification by 1-D convolutional neural networks (2016)
54. Krizhevsky, A.; Sutskever, I.; Hinton, G.: ImageNet classification with deep convolutional neural networks. Adv. Neural. Inf. Process. Syst. **25**, 1097–1105 (2012)
55. Szegedy, C., Liu, W., Jia, Y., Sermanet, P.: Going deeper with convolutions. arXiv Preprint, pp. 1–9. arXiv:1409.4842 (2014). https://doi.org/10.1109/CVPR.2015.7298594
56. Oquab, M.; Bottou, L.; Laptev, I.; Sivic, J.: Learning and transferring mid-level image representations using convolutional neural

networks. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1717–1724 (2014). https://doi.org/10.1109/CVPR.2014.222
57. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition
58. Flash, T.; Hogan, N.: The coordination of arm movements: an experimentally confirmed mathematical model. J. Neurosci. **5**, 1688–1703 (1985)
59. Pham, Q.C.; Hicheur, H.; Arechavaleta, G.; Laumond, J.P.; Berthoz, A.: The formation of trajectories during goal-oriented locomotion in humans. II. A maximum smoothness model. Eur. J. Neurosci. **26**, 2391–2403 (2007). https://doi.org/10.1111/j.1460-9568.2007.05835.x
60. Aloulou, A.; Boubaker, O.: Minimum Jerk-based control for a three dimensional bipedal robot. Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics). 7102 LNAI, pp. s251–262 (2011). https://doi.org/10.1007/978-3-642-25489-5_25
61. Bianco, C.G. Lo; Romano, M.: Bounded velocity planning for autonomous vehicles. In: 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005 (IROS 2005), pp. 685–690. IEEE (2005)
62. Shamir, T.: How should an autonomous vehicle overtake a slower moving vehicle: design and analysis of an optimal trajectory. Autom. Control. IEEE Trans. **49**, 2002–2005 (2004). https://doi.org/10.1109/TAC.2004.825632
63. Hiraoka, T.; Kunimatsu, T.; Nishihara, O.; Kumamoto, H.: Modeling of driver following behavior based on minimum-jerk theory. In: Proceedings of 12th World Congress ITS (2005)
64. Bertolazzi, E.; Frego, M.: Fast and accurate clothoid fitting. In: The 14th International Conference on Artificial Intelligence and Statistics, vol. 15, pp. 434–442 (2011). https://doi.org/10.1145/0000000.0000000
65. Girshick, R.: Fast R-CNN. In: ICCV 2015 (2015)
66. Ren, S.; He, K.; Girshick, R.; Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. **39**, 1137–1149 (2017). https://doi.org/10.1109/TPAMI.2016.2577031