Soft Comput (2018) 22:1433–1444
https://doi.org/10.1007/s00500-017-2831-0

FOCUS

Vehicle trajectory clustering based on 3D information via a coarse-to-fine strategy

Huansheng Song[1] · **Xuan Wang**[1] · **Cui Hua**[1] · **Weixing Wang**[1] · **Qi Guan**[1] · **Zhaoyang Zhang**[1]

Published online: 12 September 2017

Abstract Vehicle behavior analysis is often based on the motion trajectory analysis, which lays the foundation for many applications such as velocity detection, vehicle classification, and vehicle counting. In this paper, a trajectory clustering framework is proposed for vehicle trajectory analysis. Firstly, feature points are extracted by ORB algorithm which uses binary strings as an efficient feature point descriptor. Secondly, a matching method based on Hamming distance is used to obtain the tracking trajectory points. Finally, a novel clustering method, which contains three phrases, i.e., coarse clustering, fine clustering, and agglomerative clustering, is proposed to classify vehicle trajectory points based on the 3D information in real traffic video. By applying this clustering method in actual traffic scenes, much more stable clustering results can be obtained compared with other methods. Experimental results demonstrate that the accuracy of the proposed method can reach 95%. Furthermore, vehicle type can be estimated to realize vehicle classification.

Keywords Trajectory analysis · 3D information · ORB · Trajectory clustering · Vehicle classification

1 Introduction

Over the past several years, vehicle behavior analysis and incident prediction from video analytics have emerged as an active and challenging research area (Song et al. 2014; Sivaraman and Trivedi 2013; Wu et al. 2012). We have presented a real-time vehicle behavior analysis system in Song et al. (2014), and the trajectories of feature points have been applied to analyze vehicle behaviors such as sudden speeding up and slowing down, stopping, retrogradation, and lane changing. The system has been widely used by Chinese highway management departments, and its algorithms are robust enough for vehicle behavior analysis under complex weather conditions. However, trajectory clustering still remains an open problem in Song et al. (2014). Hence, in this paper, we focus on the problem of trajectory clustering in real traffic scenes.

In traffic surveillance applications, vehicle motion is often represented by trajectories with similar spatial and dynamic patterns. Vehicle trajectory clustering has been an extremely active research area in the intelligent transportation systems (ITSs) in the advances of camera sensing and computational technologies (Li et al. 2006; Fu et al. 2005; Atev et al. 2010). It plays a considerable role in data analysis since it reveals underlying vehicle motion information. Researchers have made many attempts on this problem for different application scenes in recent years.

Many approaches cluster trajectory data based on *distance measures* between trajectories. Mostly, trajectory data are extracted by a feature detector and a tracker. For example, Beymer et al. (1997) presented a real-time computer vision system based on the Harris corner tracking algorithm (Harris and Stephens 1988). These corners were clustered into vehicles based on proximity and similar 2D motion. Similarly, in Kanhere and Birchfield (2008), the tracked points were back-projected into 3D space and then clustered by the *Euclidean distance*.

Length differences induced by the kinematic properties of moving objects make similarity assessment between two

Communicated by M. Anisetti.

✉ Xuan Wang
jessica036@126.com

[1] School of Information Engineering, Chang'an University, Xi'an, China

or more trajectories exceedingly difficult, so *similarity measure* is a key and important step for any two trajectories with different lengths and has a greater impact on the result. For this reason, some researchers devote themselves to solving the problem of similarity measure between trajectories. Atev et al. (2010) proposed a trajectory-similarity measurement based on the *Hausdorff distance* (Huttenlocher et al. 1993), and a modification strategy is given to improve its robustness by exploiting the fact that trajectories are ordered collections of points. Abraham and Lal (2012) dealt with the length difference by applying a spatial–temporal similarity measure with the given points of interest (POI) and time of interest (TOI), in which the spatial similarity is treated as a combination of structural and sequence similarities and evaluated by using the techniques of dynamic programming. Piotto et al. (2009) proposed an algorithm for the syntactic description and matching of object trajectories in videos. Similarities between trajectories were determined based on inexact or approximate matching with trajectory segmentation and syntactic description beforehand.

Furthermore, Jung et al. (2008) used a 4D histograms to cluster the trajectory data. In its training stage, captured trajectories were grouped into coherent clusters to build 4D motion histograms with the position and instantaneous velocity of every tracked object. In the test stage, each new trajectory was compared with the 4D histograms of all the clusters. Hu et al. (2013) proposed a clustering algorithm based on the time-sensitive Dirichlet process mixture model (tDPMM) and applied it to each trajectory cluster for learning the trajectory pattern which represents the time-series characteristics of the trajectories in the cluster.

Although current trajectory clustering and modeling methods have solved a variety of specific problems in trajectory analysis, they still have the following limitations:

- Some methods need to estimate the number of trajectory clusters. However, a manual choice of the cluster numbers sometimes is subjective, and the inappropriate setting may result in inaccurate trajectory clustering.
- They fail to cluster trajectories incrementally. When new trajectories are obtained, the model has to be retrained which causes a high computational complexity. It is not able to satisfy the real-time request for the detection of traffic events.
- The traditional methods of feature point clustering mostly are based on 2D images. However, due to the impact of camera angle and lighting problems, vehicle occlusion makes a great interference on the analysis of feature point clustering.

In this paper, we propose a trajectory clustering framework in which the number of clusters can be set automatically and is progressively incremental. Moreover, a novel method is proposed, which back-projects 2D image points into a 3D space and uses 3D information for clustering. Thus, the problem of vehicle occlusion can be solved by this method. In order to improve the accuracy, we employ a coarse-to-fine strategy during trajectory clustering. Compared with other methods, the main advantage of our method lies in its high accuracy.

The rest of this paper is organized as follows. An overview of the clustering framework is presented in Sect. 2. Section 3 describes the method of feature extraction. The vehicle trajectory tracking is presented in Sect. 4. Section 5 displays the vehicle trajectory clustering method which includes three phases: trajectory coarse clustering, trajectory fine clustering, and agglomerative clustering. Experimental results are reported in Sect. 6, and finally, Sect. 7 concludes this paper.

2 Framework of clustering system

Figure 1 gives the framework of the proposed clustering system. There are four main stages in the clustering framework. Firstly, image preprocessing is done to set the region of interest (ROI) and calibrate the camera. Secondly, feature points of vehicles are extracted from images. Thirdly, feature points are tracked to obtain the motion trajectories. Finally, a coarse-to-fine clustering method is used for trajectory clustering, which back-projects the 2D trajectory points into a plane with a reference height in 3D space. In order to reduce the computational complexity, all these algorithms run on ROIs.

The vehicle trajectory clustering algorithm is completed by coarse clustering, fine clustering, and agglomerative clustering. In the phase of coarse clustering, the 3D space distances of trajectory points are regarded as the similar-

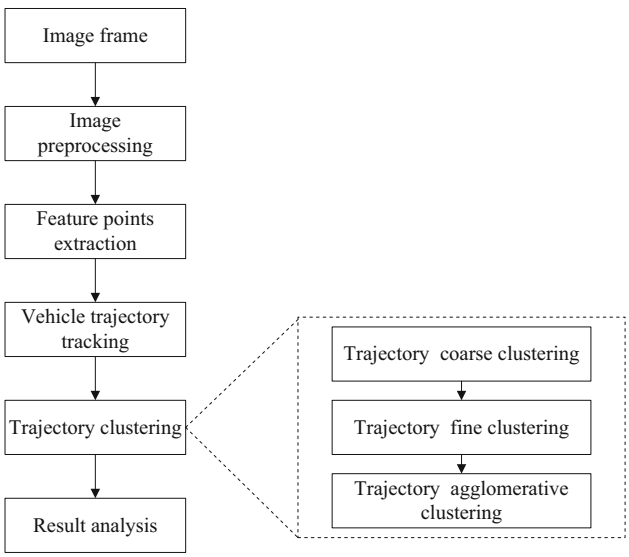

Fig. 1 Framework of the proposed clustering system

ity measure to cluster the feature points with an improved clustering method of k-means. Then, the 3D information of the tracked points is reconstructed by the correlation between back-projection velocity and height. The fine clustering is carried out by removing and reassigning the feature points within a category, which uses the threshold of vehicle models in 3D space. Finally, the 3D trajectory points are clustered among categories by the motion consistency constraint. Moreover, the result of the final clustering can be applied to the traffic flow statistics and the vehicle classification in surveillance scenes.

3 Feature point extraction

Feature point is an important local features in the vehicle detection, including motion feature information of the vehicles. It can be easily located and tracked. Therefore, this paper uses it in the research of vehicle motion trajectory clustering and traffic parameter analysis. In this stage, a detailed introduction of the ORB algorithm (Rublee et al. 2011) is given to extract the feature points. It is built on the corner detector of Features from Accelerated Segment Test (FAST) (Rosten and Drummond 2006) and feature descriptor of Binary Robust Independent Elementary Features (BRIEF) (Calonder et al. 2010). ORB is less affected by the image noise, and it is invariant to the image scale, affine, and rotation. Since it has less computing time consumption, this paper uses the ORB to gain the appropriate feature features in this stage for real-time detection.

3.1 Oriented FAST detector

In this paper, the FAST detector is employed to detect the feature points of images. One pixel will be considered as the FAST feature point if there are a set of n contiguous pixels on the circle whose intensities are larger difference than the candidate pixel $I(p)$:

$$|I(x) - I(p)| > \varepsilon \quad \forall x \in \odot p. \tag{1}$$

However, the corner response function (CRF) along with the edge is not strong in FAST algorithm, so the Harris corner response function is employed to filter the FAST feature points, and it selects the top N as candidate feature points.

Since the feature points which are detected by the original FAST have no orientations, the ORB uses the intensity centroid method (Rosin 1999) to determine the orientations of feature points. It defines some moments of a patch as:

$$M_{pq} = \sum_x \sum_y x^p y^q I(x, y), \tag{2}$$

where $I(x, y)$ is the gray intensity of feature point (x, y). With these moments, the centroid of a patch can be found:

$$C = \left(\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \right) = \left(\frac{\sum_{x,y} xI(x, y)}{\sum_{x,y} I(x, y)}, \frac{\sum_{x,y} yI(x, y)}{\sum_{x,y} I(x, y)} \right). \tag{3}$$

The orientation of the image patch can be calculated as:

$$\theta = \arctan \left(\frac{M_{01}}{M_{10}} \right) = \arctan \left(\frac{\sum_{x,y} yI(x, y)}{\sum_{x,y} xI(x, y)} \right). \tag{4}$$

3.2 rBRIEF feature descriptor

The original BRIEF can be considered as a binary string descriptor of an image patch, which is constructed by some simple binary intensity tests:

$$\tau(P; x, y) = \begin{cases} 1 & \text{if } P(x) < P(y) \\ 0 & \text{otherwise} \end{cases}, \tag{5}$$

where P is a $S \times S$ image patch of one feature point, which has been smoothed by a Gauss filter, and $P(x)$ is the gray intensity at point $x = (u, v)$. Thus, BRIEF descriptor is defined as a vector of n binary tests:

$$f_n(P) = \sum_{i=1}^{n} 2^{i-1} \tau(p; x_i, y_i). \tag{6}$$

However, the original BRIEF is considerably sensitive to the image noise, and it has no rotation invariance. Therefore, the ORB further improves it in two aspects. First, ORB uses a block of 5×5 pixel to replace the point pair so as to reduce the random noise, and these parameters are employed in this paper as well. Then, the binary string can be gained by comparing the sum of the gray value in the pixel block. Second, ORB gives the BRIEF descriptor a orientation that obtained in (4). For every feature point, it has n binary tests, and these tests compose a matrix defined as:

$$S = \begin{bmatrix} x_1 & x_2 & \dots & x_n \\ y_1 & y_2 & \dots & y_n \end{bmatrix}. \tag{7}$$

The binary test set S can be transformed into S_θ using the orientation θ of the image patch. The corresponding rotation matrix R_θ can be calculated as:

$$S_\theta = R_\theta S, \tag{8}$$

where R_θ is a matrix as:

$$R_\theta = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}. \tag{9}$$

Fig. 2 ORB descriptor applied in different scenes

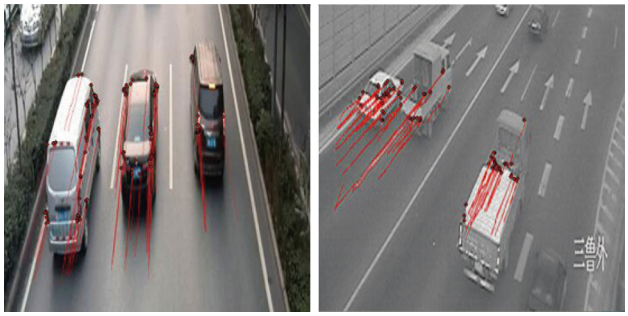

Fig. 3 Results of trajectory tracking in different scenes

Table 1 Running time analysis

Descriptor	SIFT	SURF	MSER	ORB	BRISK	A-KAZE
Time (ms)	385	103	225	19	91	277

So the steered BRIEF descriptor can be represented as:

$$f_n(P, \theta) = f_n(P) \mid (x_i, y_i) \in S_\theta. \tag{10}$$

Since the correlation of the random point pairs sometimes can be considerably large and the variance of steered BRIEF is lower, a greedy search algorithm is adopted to find the weakly correlated random point pair, which is called rBRIEF. Then, the ORB descriptor can be obtained by the above process. Figure 2 shows the results of ORB algorithm which is applied in different traffic scenes with differently viewing angles, and Table 1 presents the measured running time compared with other feature descriptors, such as SIFT (Lowe 2004), SURF (Bay et al. 2006), MSER (Forssén and Lowe 2007), BRISK (Leutenegger et al. 2011), and A-KAZE (Alcantarilla et al. 2012).

4 Vehicle trajectory tracking

In the stage of vehicle trajectory tracking, the matching method based on *Hamming distance* (Meng and You 2013) is used to solve the matching problems of feature points in consecutive frames:

$$D(F_1, F_2) = \sum_{i=0}^{n} x_i \oplus y_i, \tag{11}$$

where $F_1 = \{x_0, x_1, \ldots, x_n\}$, $F_2 = \{y_0, y_1, \ldots, y_n\}$ are ORB descriptors that are binary strings of two feature point patches in the adjacent frames, and $D(F_1, F_2)$ is the *similarity distance* of two descriptors represented by *Hamming distance*. Since the descriptor is a binary string, *Hamming distance* can be indicated as the sum of the XOR results between each two bits of the descriptors. If $D(F_1, F_2)$ is smaller than the setting threshold, F_1 and F_2 are considered as matched.

During tracking, the matching method is used to find the best matching points between the current frame and the previous frame. Finally, a 2D trajectory can be represented as a point data set as follows:

$$T_{2D}(i, n) = \{(u_{i1}, v_{i1}), (u_{i2}, v_{i2}), \ldots, (u_{in}, v_{in})\}, \tag{12}$$

where i is the number of a tracked trajectory which is in a trajectory data set, n is the number of nodes on the tracked trajectory, and (u_{in}, v_{in}) is the pixel coordinate of different nodes. The motion trajectories of feature points are shown in Fig. 3.

5 Trajectory clustering using 3D information

In this stage, a coarse-to-fine clustering method, which is based on the 3D information reconstruction of the feature point, is proposed to cluster the trajectory points. It contains three phases, including coarse clustering, fine clustering, and agglomerative clustering. In the phase of coarse clustering, all the 2D feature points are back-projected to a 3D space with the height of 0 m. An improved method of k-means is used to obtain the categories of coarse clustering. Trajectory fine clustering is carried out by the method of Assume-And-Then-Verify within the class, which uses the result of coarse clustering. Agglomerative clustering is used to merge the same categories and improve the clustering accuracy.

5.1 Trajectory coarse clustering

In this phase, the 2D trajectory points are back-projected onto 3D space and the 3D information of vehicle model is used to cluster the trajectory points. First of all, the relationship between the world coordinate system and the image coordinate system is established (Zheng and Peng 2014):

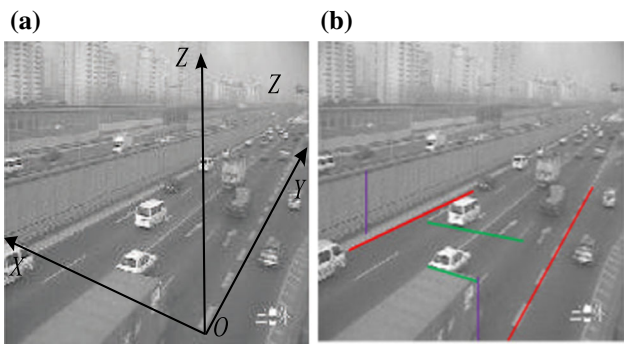

Fig. 4 Camera calibration

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X_W \\ Y_W \\ Z_W \\ 1 \end{bmatrix}, \tag{13}$$

where K is the camera internal parameters, and R and t compose the external parameter matrix of the camera. The internal and external parameters can be calculated accurately by the recovery method of vanishing points (Zheng and Peng 2014). As shown in Fig. 4, three vanishing points are formed in the image using three sets of parallel lines in the directions of X, Y, and Z, respectively. In addition, a known height (the height of the camera) is required to calculate the scale factor. Then, the camera calibration matrix can be calculated.

Apparently, the image coordinate can be calculated by the world coordinate, but the world coordinate cannot be gotten from image coordinate directly. In order to obtain the 3D information of a trajectory, the height information is necessary. However, the real height information cannot be gotten from image plane directly. Therefore, this paper adopts a novel method to handle this problem. It back-projects the 2D trajectory points to a plane called back-projected plane with $Z_W = 0$ in 3D space as shown in Fig. 5. Then, we cluster and mark these trajectory points for trajectory coarse clustering by the *Euclidean distance* in 3D space with an improved clustering method of k-means:

Step 1: Set the threshold $d_{\max}$. It is regarded as the maximum distance for different feature points in the same vehicle. The feature point, which reached the ROI first and satisfied the requirement of trajectory, is selected as the initial cluster center, and $k = 1$. The other feature points are recorded in order.

Step 2: Calculate the distance d between the cluster center P_i and the feature point P_j. If $d < d_{\max}$, P_i and P_j belong to the same category. Meanwhile, update the cluster center P_i using the average of the feature point P_i and P_j. Otherwise, P_j is regarded as a new cluster center, and $k = k + 1$.

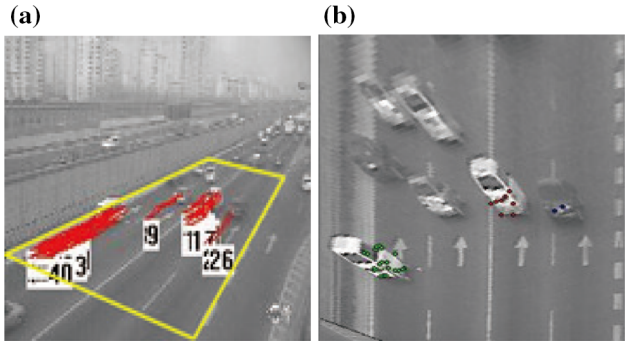

Fig. 5 Results of trajectory coarse clustering. **a** Region of back-projected plane. **b** Clustering results in the back-projected image

Step 3: Iterate the Step 2 until all the clustering centers are stable within a certain range. Then, the number of categories can be obtained.

5.2 Trajectory fine clustering

5.2.1 *Correlation between back-projection velocity and height*

The feature points are non-uniform motion in the image plane, but the points back-projected on the zero plane (horizontal plane) can be considered as uniform motion. Therefore, the velocity of all trajectory points in the same category can be calculated, which can be used for further clustering.

Assuming that the camera and the motion area are so far from each other, the camera and vehicle can be considered as in the same plane. As shown in Fig. 6, the vehicle object moves from L_1 to L_2 in time t. Meanwhile, the vehicle feature points P and Q move from P_1 to P_2 and from Q_1 to Q_2, respectively, as shown in Fig. 6. According to (13), the 2D points are back-projected to the road plane with $Z_W = 0$ in 3D space. The back-projected points A_1, A_2, B_1, B_2 can be obtained, which are corresponding to P_1, P_2, Q_1, Q_2, respectively. According to similar triangle calculations, the similarity relation can be obtained:

$$\begin{aligned} \frac{P_1P_2}{A_1A_2} &= \frac{CP_1}{CA_1} & \frac{Q_1Q_2}{B_1B_2} &= \frac{CQ_1}{CB_1} \\ \frac{V_Pt}{V_At} &= \frac{CK}{CN} & \frac{V_Qt}{V_Bt} &= \frac{CT}{CN} \\ \frac{V_P}{V_A} &= \frac{h-h_p}{h} & \frac{V_Q}{V_B} &= \frac{h-h_q}{h} \end{aligned}, \tag{14}$$

where V_P and V_Q are the actual velocities of points P and Q, respectively. V_A and V_B are the back-projection velocities. Since the vehicle can be considered as a rigid object, all the points in the same rigid body have the same actual velocity V. Thus

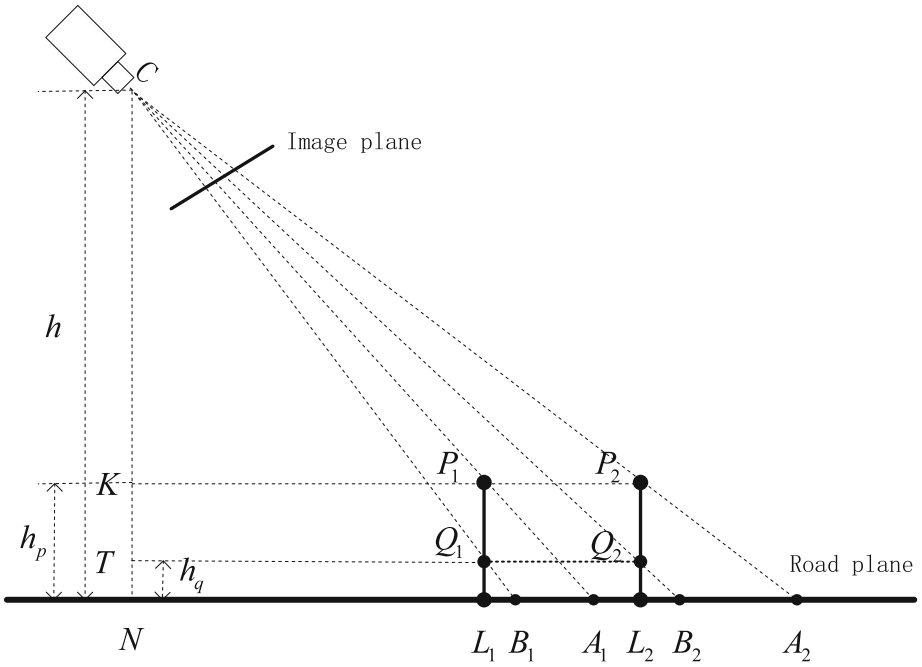

Fig. 6 Correlation between back-projection velocity and height

$$V = V_P = V_Q. \tag{15}$$

Combined with (14) and (15), it can be obtained that

$$\frac{V_A}{V_B} = \frac{h - h_q}{h - h_p}. \tag{16}$$

Furthermore, if the feature point is low enough, its back-projection velocity can be considered as the actual velocity of the vehicle.

5.2.2 Reconstruction of 3D information

After the treatment of trajectory coarse clustering, the feature points, which are in the same category, are considered to be in the same vehicle. Then, the 3D information of these feature points can be reconstructed by the following steps:

Step 1: Calculate back-projection velocity. For feature point P, its back-projection point coordinates $(X_i, Y_i, 0)$ can be calculated by (13) during the tracking. If the frame rate of the video is known, the back-projection velocity V_A can be got by (17). In order to reduce the error caused by feature point tracking, the least square method (LSM) is employed to solve this problem. The back-projection velocity V_A can be calculated by

$$\begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_N \end{bmatrix} = \begin{bmatrix} V_A & S_0 \end{bmatrix} \begin{bmatrix} t_1 & t_2 & \cdots & t_N \\ 1 & 1 & 1 & 1 \end{bmatrix}, \tag{17}$$

where S is the displacement corresponding to interval t, and N is the number of consecutive frames.

Step 2: Estimate the actual velocity of vehicle. In Sect. 5.2.1, it is known that if the feature point is close to the road, the back-projection velocity can be taken as the actual velocity of the vehicle. Therefore, the least back-projection velocity of a certain category is considered as the estimation of the actual velocity V, which is defined as:

$$V = \min\{V_i\} \ (i \in R), \tag{18}$$

where V_i is the back-projection velocity of the feature point, and R is the number of feature points in the same category.

Step 3: Reconstruct the 3D information of feature point. According to (13), the 3D coordinate (X, Y, Z) of the feature point P can be calculated with $Z = h_p$. The actual height h_p of the feature point P can be calculated by

$$h_p = h - h\frac{V}{V_A}. \tag{19}$$

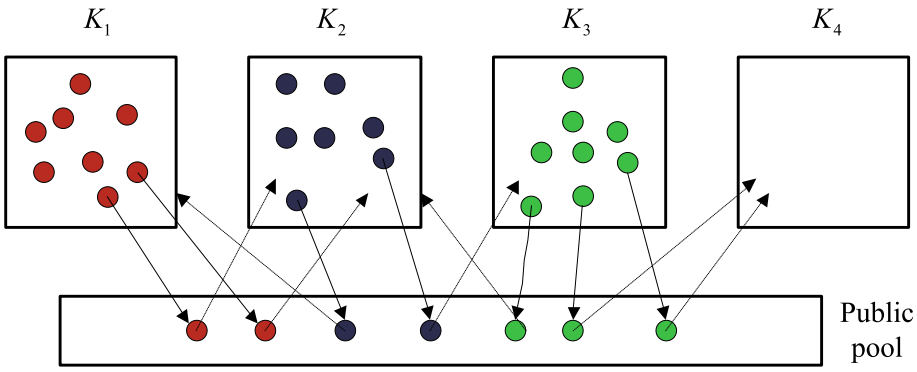

Fig. 7 Fine clustering within the category

5.2.3 Trajectory fine clustering within the category

In this phase, the method of Assume-And-Then-Verify is adopted to analyze the trajectory coarse clustering results. It will determine whether the feature points in the same category belong to the same vehicle. The fine clustering process can be found in Fig. 7. First of all, K_1, K_2, and K_3 are the results of trajectory coarse clustering. Then, some points are eliminated from the three categories by the setting threshold of 3D vehicle model. Meanwhile, they are removed to the public pool. Finally, the feature points in the public pool are redistributed. They may be reassigned to the other existing categories or become a new category K_4. The specific steps are as follows:

Step 1: Assuming that all the feature points in a certain category belong to the same vehicle and the 3D information of all the feature points has been reconstructed.

Step 2: Calculate the *Euclidean distance* between the feature point P_i and the lowest point $P_{\min}$:

$$\begin{aligned} \triangle X &= |X_{P_i} - X_{P_{\min}}| \\ \triangle Y &= |Y_{P_i} - Y_{P_{\min}}| \\ \triangle Z &= |Z_{P_i} - Z_{P_{\min}}|. \end{aligned} \tag{20}$$

Step 3: Match $D(\triangle X, \triangle Y, \triangle Z)$ with the preset 3D vehicle model $T(L, W, H)$. If it satisfies any of the three models, it means that the feature point belongs to a certain category. Otherwise, it is removed to public pool.

$$P_i \in \begin{cases} \text{Small} & D(\triangle X, \triangle Y, \triangle Z) \le T_{\min}(L, W, H) \\ \text{Midsize} & T_{\min}(L, W, H) \le D(\triangle X, \triangle Y, \triangle Z) \\ & \le T_{\text{mid}}(L, W, H) \\ \text{Oversize} & D(\triangle X, \triangle Y, \triangle Z) \ge T_{\text{mid}}(L, W, H). \end{cases} \tag{21}$$

Step 4: Redistribute the feature points in the public pool. As shown in Fig. 7, if the feature point was removed from K_1, it needs to be verified in other categories (K_2 and K_3). Assuming it belongs to K_2, it will be verified by Steps 2 and 3 in turn. If the feature point does not belong to K_2 or K_3, it is regarded as a new category K_4 and the number of categories is increased by 1.

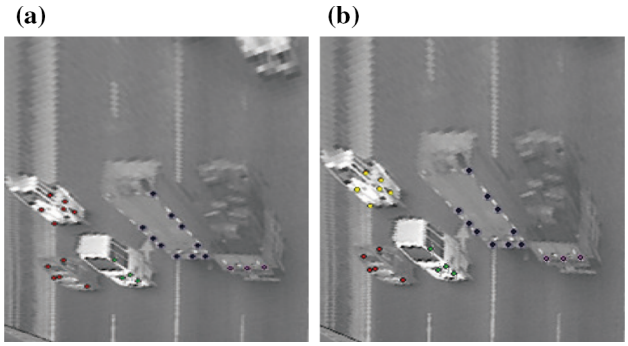

Fig. 8 Clustering result in the back-projected image. **a** Result of trajectory coarse clustering. **b** Result of trajectory fine clustering

The feature points have been clustered in Sect. 5.1, but the results are inaccurate as shown in Fig. 5. Then, trajectory fine clustering within the category is done based on the results of coarse clustering, as shown in Fig. 8. It can be seen that the feature points of two vehicles are considered as the same group after trajectory coarse clustering as shown in Fig. 8a. Then, fine clustering is applied to do treatment within this category, as shown in Table 2. There are 11 feature points in this category, and their back-projection velocities are calculated in Table 2. In order to make the cluster center near the vehicle center, we select the medium of back-projection velocities as the reference velocity, and the corresponding feature point is regarded as reference point, as shown by the P_{14} in Table 2. Some feature points are removed from the current category by the 3D vehicle model, and then the feature points in the public pool are redistributed. Finally, the new clustering categories are obtained, as shown in Fig. 8b.

Table 2 3D information of feature points

P_i	$V_i(m/s)$	$Z/\triangle Z(m)$	$X(m)$	$Y(m)$	$\triangle X(m)$	$\triangle Y(m)$	Keep
3	19.0017	0.00047	23.3565	54.5949	0.8627	12.9864	No
5	19.3521	−0.16867	24.4520	51.7258	1.9582	10.1173	No
6	18.9761	0.01286	23.4899	49.0271	0.9961	7.4186	No
7	19.3099	−0.14831	23.7127	50.9656	1.2189	9.3571	No
8	17.7905	0.58514	21.7788	46.2992	−0.715	4.6907	Yes
9	14.7117	2.07132	20.2717	38.3467	−2.2221	−3.2618	Yes
14	19.0027	0	22.4938	41.6085	0	0	Yes
13	24.9512	−2.8714	27.6698	54.141	5.1760	12.5325	No
15	16.6256	1.14745	21.2763	36.1522	−1.2175	−5.4563	No
16	19.9981	−0.48051	24.8816	37.8451	2.3878	−3.7634	Yes
17	20.2295	−0.59220	24.7772	38.3853	2.2834	−3.2232	Yes

5.3 Agglomerative clustering

In this phase, the method of Assume-And-Then-Verify is still used to judge whether two categories need to be merged. Assuming that the two categories belong to the same category, the feature point, which has the lowest back-projection velocity in the two categories, is considered as the reference point. Its category is regarded as the reference category. The back-projection velocity of the reference point is used as the real velocity to reconstruct the 3D information of feature points in the other category. Then, the 3D trajectories can be obtained by the process of tracking. Last but not least, motion consistency constraint is used to determine whether the other feature points and the reference point are in the same category. We verified the authenticity of the assumption by recording the number of the feature points which meet the condition of motion consistency constraint. If the number is larger than the setting threshold (the threshold needs to account for 60% of the total number), the two categories are considered to be in the same vehicle. Then, they need to be merged into one category. Otherwise, the two belong to different vehicles. The final clustering results are obtained by comparing all the categories in turn.

The determinant steps of motion consistency constraint are as follows:

Step 1: Estimate the reference trajectory of category K_i. Select the feature point which has the lowest back-projection velocity of K_i as the reference point, and set its height with $Z = 0$. The 3D trajectory points of the reference point can be obtained from the 2D trajectory points by (13):

$$L_i = \{(X_{i1}, Y_{i1}, Z_{i1}), (X_{i2}, Y_{i2}, Z_{i2}), \ldots, X_{im}, Y_{im}, Z_{im})\}. \tag{22}$$

Step 2: Estimate the 3D trajectory of feature points in category K_j. Assuming that the two categories belong to the same vehicle, the 3D information of the feature points in K_j can be reconstructed with the height of the reference point by (16). Then, the 3D trajectory points in K_j can be obtained by the process of tracking:

$$L_j = \{(X_{j1}, Y_{j1}, Z_{j1}), (X_{j2}, Y_{j2}, Z_{j2}), \ldots, (X_{jn}, Y_{jn}, Z_{jn})\}. \tag{23}$$

Step 3: Calculate the distance between the two trajectory points by

$$\mathrm{Dis}_t = \sqrt{(X_{it} - X_{jt})^2 + (Y_{it} - Y_{jt})^2 + (Z_{it} - Z_{jt})^2}, \tag{24}$$

where t means that the two trajectory points are at the same moment.

Step 4: Calculate the sum of absolute differences between the two trajectory points:

$$C = \frac{i}{l-1} \sum_{t=2}^{l} |\mathrm{Dis}_t - \mathrm{Dis}_{t-1}|. \tag{25}$$

Step 5: Compare C with the threshold $C_{\min}$. If $C \leq C_{\min}$, it means that the two trajectories satisfy the motion consistency constraint. Otherwise, the two trajectories belong to different vehicles.

In theory, if two trajectories belong to the same rigid object, they should meet the motion consistency constraint. It means that they are in the same motion state. Thus, the distance between the two trajectory points is a constant value, which means $C = 0$. However, there are some errors during

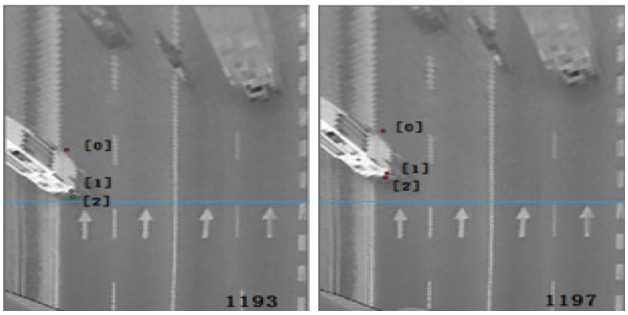

Fig. 9 Result of agglomerative clustering between categories in the back-projected image. **a** Result of trajectory coarse clustering. **b** Result of merging

the camera calibration and the real velocity has not been got accurately. Therefore, we set the threshold $C_{\min} = 0.2$ by a series of tests. Figure 9 shows the merging results between the categories. There are three trajectories in frame 1193. The coordinates of 3D trajectory points are reconstructed by the reference point P_0 as shown in Table 3. The motion consistency constraint C is calculated in Table 4. It can be seen that the feature points of the vehicle are divided into two categories after coarse clustering as shown in Fig. 9a, but they are merged into the same category in frame 1197 after the analysis of the motion consistency constraint as shown in Fig. 9b.

6 Experimental results

In order to demonstrate the effectiveness and robustness of the proposed clustering framework, it has been tested in different real video sequences, including highway and urban. These scenarios are tested on a Windows 7 platform with a Pentium 4 3.2 GHz central processing unit and 4 GB random access memory. The size of each image is 720 × 288, and the sampling frequency is 25 FPS. The proposed cluster framework is implemented with Visual C++ on a raw video format. The experimental results are shown in Fig. 10.

Table 4 Motion consistency discriminant

Time	P_0 and P_1		P_1 and P_2	
	Dis	C_1	Dis	C_2
t_1	3.2420		4.0207	
t_2	3.2726		3.9818	
t_3	3.3070		4.0089	
t_4	3.3459	0.0152	4.0153	0.03621
t_5	3.3455		3.9867	
t_6	3.4152		4.1157	
t_7	3.3668		4.0312	

There are three parts in the experimental results. The statistical results of traffic flow are addressed in Part A. Part B shows the results of vehicle classification and accuracy in different scenes. Finally, a comparison with other systems is shown in Part C.

6.1 Statistical results of traffic flow

In the test, the number of vehicles is counted according to the moment when a trajectory appears and disappears. In other words, the final clustering results are obtained by the coarse-to-fine cluster method: When feature points of a certain category appear, they are clustered by coarse clustering, and when the trajectory points disappear, fine clustering and agglomerative clustering are carried out. Meanwhile, this category is deleted and the counter is added by one, as shown in Fig. 11. The proposed method has been tested in different traffic scenes, and the results are shown in Table 5.

6.2 Vehicle classification

According to the process of fine clustering, the 3D parameters of feature points can be reconstructed. When the final clustering is completed, the feature points of the same vehicle are aggregated into the same category. Then, the 3D coordinates of each feature point can be used to determine the position

Table 3 Reconstruction of 3D information

P_i	P_0		P_1		P_2	
Time	$X(m)$	$Y(m)$	$X(m)$	$Y(m)$	$X(m)$	$Y(m)$
t_1	23.8815	66.0134	23.6070	62.7830	23.5879	62.0034
t_2	23.8792	66.4132	23.6102	63.1516	23.5910	62.4418
t_3	23.8804	66.8908	23.6096	63.5949	23.5917	62.8923
t_4	23.8804	67.3044	23.5998	63.9702	23.5805	63.3003
t_5	23.8830	67.7352	23.6015	64.4015	23.5881	63.7594
t_6	23.8835	68.2533	23.6068	64.8493	23.5900	64.1481
t_7	23.8829	68.6146	23.6116	65.2587	23.5931	64.5938

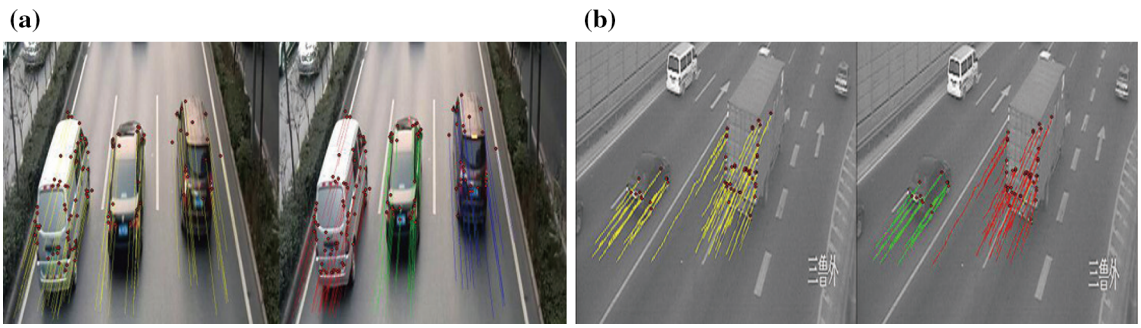

Fig. 10 Clustering results for different traffic scenes. **a** Urban road in Xi'an. **b** Highway in Shanghai

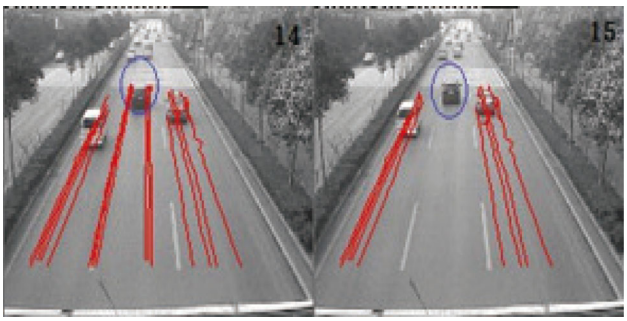

Fig. 11 Counting process

of the vehicle boundary. Thus, the vehicle classification can be obtained, as shown in Fig. 12. Table 6 shows the results of vehicle classification statistics in different traffic scenes. The experimental result shows that the detection rate of the proposed method is above 90%.

6.3 Comparison with other clustering frameworks

In order to show the improvements achieved by the proposed clustering framework, it is compared with other clustering frameworks as given in Table 7. These clustering frameworks are applied to the urban road of Shanghai. As shown in Table 7, Rabaud and Belongie (2006) clustered the moving objects by using motion consistency constraint and identified individuals under the assumption that features from a single object will have the same affine motion; Kanhere and Birchfield (2008) used a plumb line projection (PLP) to compute the 3D coordinates of the stable feature points, and these stable feature points are grouped and tracked to provide the vehicle count. However, the clustering framework of Rabaud and Belongie (2006) is adapted in 2D image plane, which cannot handle the perspective effects encountered at low camera angles. The motion consistency constraint is not suitable for segment vehicle in such traffic scene. From the results in Table 7, it can be seen that the detection rate is

Table 5 Statistical results of traffic flow

Traffic scenes	Detection/inspection	Errors	Detection rate (%)	Recall rate (%)
Outer Ring Load of Shanghai	1548/1610	102	96.14	89.81
Second South Ring Road of Xi'an	1892/1988	99	95.17	90.19
Suburb of Beijing	803/838	53	95.82	89.50

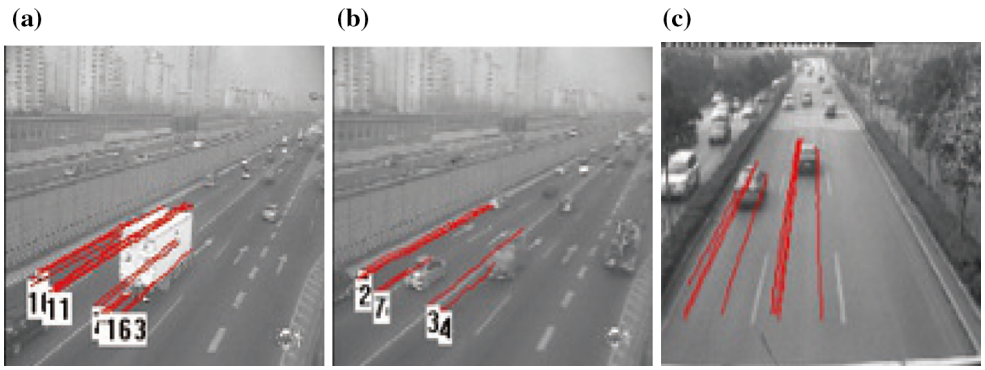

Fig. 12 Vehicle classification. **a** Oversize vehicle. **b** Midsize vehicle. **c** Small vehicle

Table 6 Results of vehicle classification

Traffic scenes	Detection/inspection	Errors	Detection rate (%)	Recall rate (%)
Outer Ring Load of Shanghai				
Oversize	572/635	63	90.07	80.16
Midsize	296/314	18	94.27	88.54
Small	617/662	45	93.20	86.40
Second South Ring Road of Xi'an				
Oversize	365/391	26	93.33	86.70
Midsize	494/534	49	90.97	83.33
Small	958/1033	75	92.74	85.48
Suburb of Beijing				
Oversize	336/364	28	92.31	84.62
Midsize	253/278	25	91.01	82.01
Small	221/244	23	90.57	81.15

Table 7 Comparison with other cluster frameworks

Cluster framework	Recall rate (%)	Detection rate (%)
Rabaud and Belongie (2006)	68.67	77.67
Kanhere and Birchfield (2008)	80.23	90.91
Our method	89.81	96.14

lower than the others. Even though the clustering framework of Kanhere and Birchfield (2008) has a good performance, the detection rate is also lower than the proposed clustering framework. We combine the idea of the former two papers and use the motion consistency constraint in 3D space to cluster vehicle trajectory points. It can be seen that the proposed clustering framework has good robustness in different traffic scenes.

7 Conclusion

In this paper, a bottom-up clustering framework is proposed for vehicle trajectory clustering in traffic scenes. There are four main stages in the proposed clustering framework: image preprocessing, feature point extraction, vehicle trajectory tracking, and trajectory clustering. In the stage of the feature point extraction, ORB algorithm is used to select the appropriate feature points. Since it can reduce the computational time and satisfy the requirement of real-time detection, ORB descriptor performs better than the other feature descriptors in the real-time application. In the stage of trajectory clustering, the 3D parameters of the feature points are reconstructed by the proposed method, which can be applied in the coarse-to-fine clustering method. The proposed clustering framework has been employed in different traffic scenes (highway and urban) to test the performance of clustering. Experimental results show that the cluster accuracy of the proposed method is very high, reaching 96%, even if the traffic condition is quite complex. Moreover, it has a good performance in solving the problem of vehicle occlusion. The proposed clustering framework can also be applied in different areas, such as traffic parameter measurement, vehicle classification, and abnormal event alarm.

Acknowledgements This work is supported by the National Natural Science Fund of China (No. 61572083), the Natural Science Foundation of Shaanxi Province (Nos. 2015JQ6230, 2015JZ018), and the Fundamental Research Funds for the Central Universities of China (Nos. 310824163411, 310824171003).

Compliance with ethical standards

Conflict of interest All authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Abraham S, Lal PS (2012) Spatio-temporal similarity of network-constrained moving object trajectories using sequence alignment of travel locations. Transp Res Part C 23:109–123

Alcantarilla PF, Bartoli A, Davison AJ (2012) Kaze features. In: Proceedings of European conference on computer vision

Atev S, Miller G, Papanikolopoulos N (2010) Clustering of vehicle trajectories. IEEE Trans Intell Transp Syst 11(3):647–657

Bay H, Tuytelaars T, Gool LV (2006) Surf: speeded up robust features. In: Proceedings of European conference on computer vision

Beymer D, McLauchlan P, Coifman B, Malik J (1997) A real-time computer vision system for measuring traffic parameters. In: Proceedings of IEEE conference on computer vision and pattern recognition, pp 495–501

Calonder M, Lepetit V, Strecha C, Fua P (2010) Brief: binary robust independent elementary features. In: Proceedings of European conference on computer vision

Forssén PE, Lowe DG (2007) Shape descriptors for maximally stable extremal regions. In: Proceedings of IEEE conference on computer vision

Fu Z, Hu W, Tan T (2005) Similarity based vehicle trajectory clustering and anomaly detection. In: IEEE international conference on image processing, ICIP, pp 602–605

Harris C, Stephens M (1988) combined corner and edge detector. In: Proceedings of the Alvey vision conference, pp 147–151

Hu W, Li X, Tian G, Maybank S, Zhang Z (2013) An incremental DPMM-based method for trajectory clustering, modeling, and retrieval. IEEE Trans Pattern Anal Mach Intell 35(5):1051–1065

Huttenlocher D, Klanderman G, Rucklidge W (1993) Comparing images using the Hausdorff distance. IEEE Trans Pattern Anal Mach Intell 15:850–863

Jung C, Hennemann L, Musse S (2008) Event detection using trajectory clustering and 4-D histograms. IEEE Trans Circuits Syst Video Technol 18(11):1565–1575

Kanhere NK, Birchfield ST (2008) Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features. IEEE Trans Intell Transp Syst 9(1):148–160

Leutenegger S, Chli M, Siegwart R (2011) BRISK: binary robust invariant scalable keypoints. In: Proceedings of IEEE conference on computer vision

Li X, Hu W, Hu W (2006) A coarse-to-fine strategy for vehicle motion trajectory clustering. In: 18th International conference on pattern recognition (ICPR'06), vol 1, pp 591–594

Lowe DG (2004) Distinctive image features from scale-invariant keypoints. Int J Comput Vision 60(2):91–110

Meng F, You F (2013) A tracking algorithm based on ORB. In: Proceedings of international conference on mechatronic sciences, electric engineering and computer, pp 1187–1190

Piotto N, Conci N, Natale FD (2009) Syntactic matching of trajectories for ambient intelligence applications. IEEE Trans Multimedia 11(7):1266–1275

Rabaud V, Belongie S (2006) Counting crowded moving objects. In: Proceedings of IEEE conference on computer vision and pattern recognition

Rosin PL (1999) Measuring corner properties. Comput Vis Image Underst 73(2):291–307

Rosten E, Drummond T (2006) Machine learning for high speed corner detection. In: Proceedings of European conference on computer vision, pp 2091–2096

Rublee E, Rabaud V, Konolige K, Bradski G (2011) ORB: an efficient alternative to SIFT or SURF. In: Proceedings of IEEE conference on computer vision, pp 2564–2571

Sivaraman S, Trivedi M (2013) Looking at vehicles on the road: a survey of vision-based vehicle detection, tracking, and behavior analysis. IEEE Trans Intell Transport Syst 14(4):1773–1795

Song H, Lu S, Ma X, Yang Y, Liu X, Zhang P (2014) Vehicle behavior analysis using target motion trajectories. IEEE Trans Veh Technol 63(8):3580–3591

Wu J, Cui Z, Chen J, Zhang G (2012) A survey on video-based vehicle behavior analysis algorithms. J Multimedia 7(3):223–230

Zheng Y, Peng S (2014) A practical roadside camera calibration method based on least squares optimization. IEEE Trans Intell Transport Syst 15(2):831–843