A Picture is Worth a Thousand Words: Share Your Real-Time View on the Road

Xiong Wang, Lei Ding, Qi Wang, Jin Xie, Tianyi Wang, Xiaohua Tian, *Member, IEEE*, Yunfeng Guan, and Xinbing Wang, *Senior Member, IEEE*

***Abstract*—The visual information provided by applications facilitating urban communications such as Google Street View and Waze scene report is intuitionistic and useful since it conforms to human cognitive habits. However, current street view services cannot provide real-time information, and the scene report service is passive. An on-demand and real-time visual-information-providing mechanism is still unavailable. In this paper, we develop the crowdsourcing-based Real-time View Share (RVShare) system, which provides pictures of requested locations taken by travelers just passing by. To enable the RVShare system, we propose a view-sharing job distribution mechanism, where a wavelet-based vehicle prediction scheme and a tree-searching-based tracking scheme are developed to select vehicles for view sharing. We also design a simple but effective incentive mechanism to encourage more travelers to participate into the view-sharing activity, where the rewarding process takes the quality of uploaded pictures into account. Moreover, we develop a processing flow for uploaded pictures to improve the accuracy of the picture quality evaluation and verify the picture content. Comprehensive experiment results from road tests are conducted to evaluate the performance of the RVShare system.**

***Index Terms*—Crowdsourcing system, on-demand, real-time, visual.**

I. Introduction

TODAY'S urban communications have been significantly facilitated by smartphone applications, where collecting comprehensive and real-time road information is the cornerstone function. Map services like Google maps provide not only turn-by-turn instructions to navigate a user to a strange place but provide traffic congestion conditions and estimated time of arrival as well. Current map services enable the street view function, which provisions real visual information of certain places to help users get intuitionistic feelings. The visual information conforms to human cognitive habitats since the image of landmarks or buildings could be more meaningful than those highly abstracted signs on the traditional map. In addition, current map services also provide congestion levels labeled by different colors through sensor data fusion [1], [2]. Waze is another kind of an app utilizing the crowdsourcing-based approach [3], [4] to serve drivers, which forms a social network of drivers to share road information. Waze allows drivers to file a real-time report on a scene such as traffic jams, accidents, and police traps, where a picture can be attached to the report to illustrate the real situation. The picture makes the report more informative, but the reporter has to be around the scene so that the truthfulness of the report can be guaranteed.

A picture is worth a thousand words in the context of road information collection and recognition. A picture of a landmark taken by a passing traveler can be more useful than the panoramic view of the landmark obtained months ago since it is perhaps under external remodelling. A picture taken by a traveler currently on the road combined with the red line segment on Google maps can be more meaningful for a user to realize the degree of traffic congestion, and the information delivered by the picture can be up-to-date compared with the color of the line segment waiting for update. However, the current street view service can only provide views surveyed by the camera man maybe weeks ago, and the Waze report can provide the visual information of a scene only if there is a reporter happening to be around.

This paper proposes a scene reporting system, i.e., Real-time View Share (RVShare), which provides on-demand and real-time visual information based on a crowdsourcing approach, which can further improve the user's quality of experience (QoE) for urban communications such as path planning and navigation. With RVShare, a user can request real-time visual information at a location, and the server will distribute the request to possibly passing vehicles so that the most recent picture of the location in a vehicle passenger's perspective can be provided to the user. We utilize the smartphone in the front of a vehicle to get access to vehicle's acceleration, orientation, and road's scene information for the evaluation of the RVShare system. More specifically, we have the following three-fold contributions.

Manuscript received December 6, 2015; revised March 4, 2016, April 12, 2016, and May 27, 2016; accepted July 12, 2016. Date of publication July 18, 2016; date of current version April 14, 2017. This work was supported in part by the National Natural Science Foundation of China under Grant 61532012, Grant 61325012, Grant 61271219, Grant 61521062, Grant 61428205, Grant 91438115, Grant 61572319, and Grant U1405251 and in part by the National Mobile Communications Research Laboratory, Southeast University under Grant 2014D07. The review of this paper was coordinated by Dr. Y. Ji.

X. Wang, T. Wang, Y. Guan, and X. Wang are with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: wangxiongsjtu@sjtu.edu.cn; c86668248@sjtu.edu.cn; yfguan69@yahoo.com; xwang8@sjtu.edu.cn).

L. Ding is with the Department of Electrical Engineering, University of California Los Angeles, Los Angeles 90024, USA (e-mail: stoner@sjtu.edu.cn).

Q. Wang is with the Department of Computer Science, Columbia University, New York, NY 10027 USA (e-mail: paoptu023@sjtu.edu.cn).

J. Xie is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA (e-mail: xiejin@sjtu.edu.cn).

X. Tian is with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai 200240, China, and also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: xtian@sjtu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVT.2016.2592685

0018-9545 © 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

First, we propose a view-sharing job distribution mechanism, where a wavelet-based vehicle prediction scheme and a tree-searching-based tracking scheme are developed to select vehicles for view sharing. If there are no vehicles near the requested view-sharing location to take pictures, our proposed vehicle selection scheme could trace vehicles in a larger area around and determine vehicles that may pass by the requested location. We fully exploit the advantage of wavelet analysis, which is an effective way for extracting the features of time-series data. To the best of our knowledge, it is the first time wavelet analysis has been integrated into vehicle maneuver prediction applications. Field experiments show that the wavelet-based prediction can achieve an average of 95.1% accuracy.

Second, we design a simple but effective incentive mechanism to encourage more travelers to participate into the view-sharing activity, which takes the special requirements of the view-sharing application into account. The quality of uploaded pictures is reflected in the rewarding stage of the incentive mechanism, which promotes the effectiveness of reward utilization and the quality of uploaded pictures. The quality is evaluated by comparing the uploaded picture with the current street view image, where the number of feature match points is counted. The number of feature match points can reflect the content and the shooting angle of the uploaded picture, both of which are relative to the picture's quality directly.

Third, we develop a processing flow for uploaded pictures to improve the accuracy of the picture quality evaluation in the view-sharing case. The current street view images usually have been processed, which results in many incorrect matches in the matching process, thus influencing the accuracy of picture quality evaluation. We perform ratio and symmetry tests, as well as the random sample consensus (RANSAC) method [5], to filter out wrong feature matches. We find that such simple but effective image processing approaches are very suitable for the view-sharing application, where we specify criteria for picture quality evaluation referable for other similar applications.

The remainder of this paper is organized as follows. Section II presents related work. Section III demonstrates the architecture of RVShare system and design challenges. We describe vehicle selection procedure to distribute view-sharing job among selected vehicles in Section IV. In Section V, the Vickrey–Clarke–Groves (VCG) auction-based incentive mechanism is designed to calculate payments for uploading pictures. In Section VI, we present image matching and picture quality evaluation. Section VII shows experiment and simulation results of the RVShare system. Conclusion remarks are drawn in Section VIII.

II. Related Work

Road Information Collection: Acquiring road information is a critical technique for applications such as traffic congestion alleviating and route planning, which attracts much attention from the research community. Wireless sensor networks are wildly used to sense road conditions, such as those systems utilizing smart cameras [6] and ultrasonic sensors [7]. However, the wireless sensor network cannot cover every corner of wide urban areas, and the deployment is costly. The rise of mobile crowdsensing paradigm makes it possible to cover a wider range of urban areas at low cost [8]. Arnaud *et al.* propose a probabilistic approach for road hazard warnings taking advantage of crowdsourced drivers [9], which, however, cannot evaluate the credibility of the driver's road report. Our work in this paper presents a crowdsourcing approach to collect road information, where smartphone cameras are utilized through coordination. Since our system implements three key functions, we hereby briefly survey existing efforts to help enable each of the functions.

Vehicle Prediction and Tracking: The probabilistic theory is usually utilized to perform vehicle maneuver prediction. In [10], the probabilistic finite-state machine and fuzzy logic are combined to recognize the vehicle's maneuver, but the prediction process is not reliable with accuracy lower than 80%. Firl *et al.* proposed a probabilistic approach based on advanced driver assistance system to predict the vehicle's lane-change maneuver [11], but the utilization of professional camera and radar is costly, and the accuracy is only 40% when prediction lead time reaches 1 s. In [12], Ohn-Bar *et al.* learn holistic features first and then predict driver's maneuvers based on the learning data, but an array of sensors are needed to capture driver's gestures and road conditions. Chen *et al.* propose a scheme to detect the vehicle's maneuver through the gyroscope sensor embedded in smartphones [13]. However, they cannot predict the vehicle's maneuver in advance. We propose a scheme to predict the vehicle's near future actions using smartphone sensors, which can achieve high prediction accuracy and relatively large prediction lead time.

A video-based approach is extensively applied to tackle the vehicle tracking issue. Hajimolahoseini *et al.* propose an algorithm to track the vehicle from videos captured by fixed cameras [14], whereas Xiong *et al.* propose a method to obtain the vehicle's trajectory in the video based on the pixel-level motion vector [15]. However, the video-based method is not flexible, and the cost is extremely high. The Global Positioning System (GPS)-based tracking scheme is more favorable with the help of a satellite system [16]; thus, we propose a tracking scheme to trace vehicle's trajectory with GPS data, which is flexible and less costly.

Incentive Mechanism: Game theory, auction theory, and contract theory are popular tools to design incentive mechanisms. The Stackelberg game and auction theory are used in [17] and [18]. Gao *et al.* use Lyapunov optimization and auction theory to calculate payments for participatory mobile crowd sensing workers under the constraint that workers will not quit [19]. However, few previous works consider the data quality in a real system. To this end, Wen *et al.* design a quality-driven auction-based incentive mechanism for indoor localization system in which quality of received signal strength (RSS) data is defined [20]. However, the data quality definition is only for evaluating RSS signals and not suitable for pictures. In this paper, we design an effective incentive mechanism that also considers inherent characteristics of the picture quality.

Image Feature Matching: An important part of our system is to match uploaded pictures with street view images to evaluate the picture quality, where the challenge is that the street view images are with strongly rendering by the service provider.

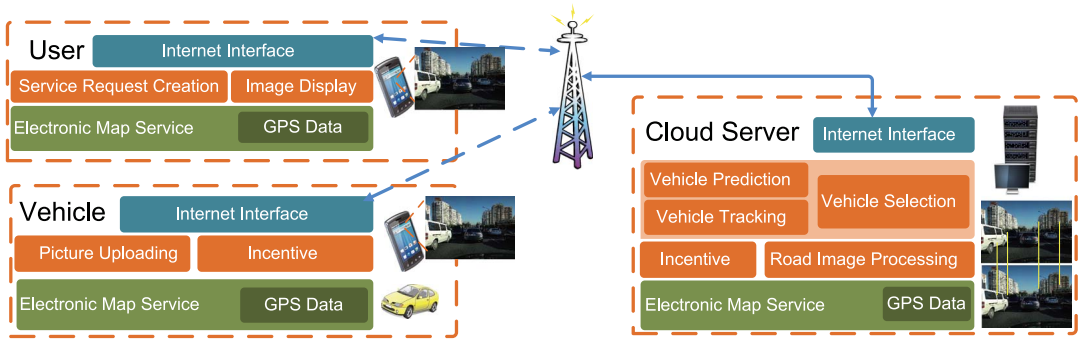


Fig. 1. RVShare system architecture.

This makes the traditional way of feature matching infeasible. Lowe proposes a scale-invariant local feature descriptor called scale-invariant feature transform (SIFT) [21], which presents outstanding performance over other feature descriptors when the image rotates or the scale of the image changes. To improve descriptor efficiency, Bay *et al.* [22] propose a speeded-up robust feature (SURF). Arandjelovic *et al.* [23] propose the RootSIFT feature descriptor based on SIFT feature descriptors. However, our experiment results show that simply realizing any of the given processing approaches cannot provide acceptable feature matching performance for the view-sharing application; therefore, a further process is necessary to improve evaluation accuracy.

III. System Architecture and Design Challenge

The RVShare system consists of three major parts: *User*, *Vehicle*, and *Cloud Server*, which are shown in Fig. 1. A walkthrough is described as follows:

Stage 1: Based on the *Electronic Map Service*, the user performs the *Service Request Creation* to the cloud server for road information acquisition at a requested location. The request is associated with the *GPS Data* and sent out through the *Internet Interface*.

Stage 2: The cloud server receives the user's request, and then carries out the *Vehicle Prediction* and the *Vehicle Tracking* near the requested location based on the *Electronic Map Service* and the *GPS Data* to finally determine vehicles that will pass by the requested location. Then, the cloud sever requests these vehicles to upload road pictures.

Stage 3: Vehicles receive the cloud server's request, and bid prices for uploading pictures.

Stage 4: The cloud server selects the winning vehicle set to upload road pictures and calculates the corresponding payments. Winning vehicles perform the *Picture Uploading* to the cloud server as well as obtaining payments.

Stage 5: The cloud server performs the *Road Image Processing* to match those uploaded images with Baidu street view. Then, the cloud server selects the highest quality image and sends it back to the user. The user receives and displays the image to acquire the real-time visual road information.

To realize the system, we have to overcome the following nontrivial challenges.

- The user is unwilling to wait for a long time after sending the request; therefore, we need to reduce the waiting time as much as possible. If there are no vehicles at the requested location, an intuitive method is to ask vehicles around to take road pictures for the user. However, these vehicles may not pass by the requested location. We propose a vehicle selection scheme to first search vehicles around the requested location and then select those who will arrive at the requested location, which can reduce the user's waiting time as much as possible.
- Vehicles are unwilling to upload road pictures since it consumes power and traffic flow. Moreover, only qualified pictures are useful to reflect the road situation. We propose a quality-based auction that calculates payments according to the picture quality and satisfies favored economic properties.
- Precisely verifying the content of uploaded picture directly is difficult since it needs to consider several requirements. To tackle this problem, we match uploaded pictures with Baidu street view for quality evaluation. However, Baidu street view images have been processed so that the matching process may yield many incorrect matches. We carry out ratio and symmetry tests and utilize the RANSAC method to remove wrong feature matches. Furthermore, we define the picture quality according to matching results.

IV. Vehicle Selection

Upon receiving a request, the cloud server first searches vehicles around the requested location and then find out whether those searched vehicles will pass by. Consider the example shown in Fig. 2, P denotes the location of a vehicle, D is the requested location, and A and B are two nearby intersections around the location. If the vehicle intends to pass by D, it needs to move toward D. Suppose the vehicle moves along the path illustrated by the red line, it has to turn right at A; otherwise, it will not pass by D. To find out whether a vehicle will pass by the requested location as soon as possible, we need to predict the vehicle's maneuver including turning left, turning right, and going straight at intersections in advance. Meanwhile, we need to track the vehicle to see whether it will move to the

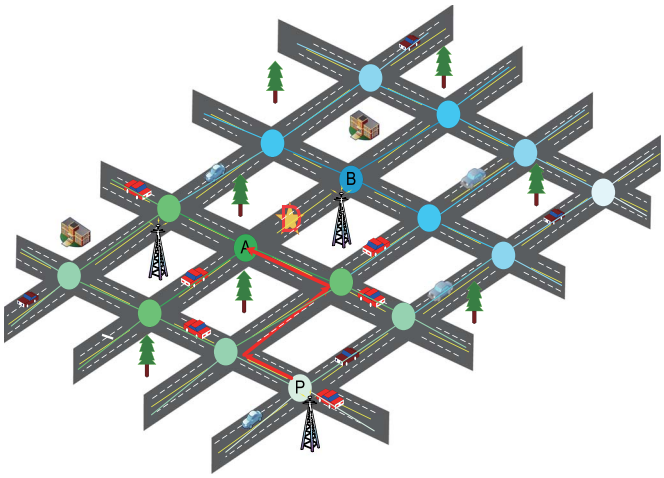


Fig. 2. Schematic of the vehicle selection process.

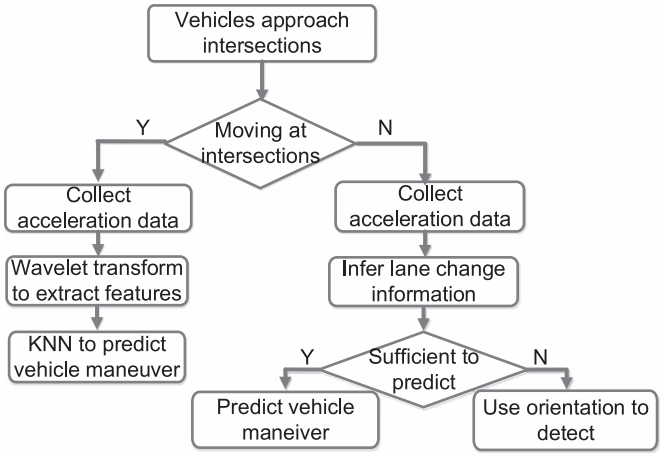


Fig. 3. Process of classified discussion.

requested location. We divide the vehicle selection process into the vehicle prediction part and the vehicle tracking part. Correspondingly, we propose vehicle maneuver prediction algorithm (VMPA) and vehicle trajectory tracking algorithm (VTTA) to predict vehicle's maneuver and track vehicles, respectively.

A. Vehicle Prediction

The cloud server will collect the vehicle's acceleration data when it approaches intersections and then analyzes the data to predict its maneuver. However, if the vehicle stops when traffic lights turn red, the acceleration data are of little avail for the maneuver prediction; therefore, we need to investigate situations of moving and stopping when approaching intersections, which we also call moving and stopping at intersections, respectively. When vehicles move at intersections, the cloud server gathers vehicle's acceleration data and then implement wavelet transform to extract its features, which are utilized to predict vehicle's maneuver. When vehicles stop at intersections, the cloud server first uses the lane-change information to predict vehicle's maneuver. If the lane-change information is not sufficient for the prediction, orientation data are leveraged to detect vehicle's maneuver. The direction of the collected acceleration data are perpendicular to the vehicle's moving direction and from the vehicle's left side to the vehicle's right side. Since we only gather acceleration data and orientation data when vehicles approach intersections with a low sampling rate, the data traffic is not high for cellular networks. We also illustrate this classified discussion for situations of vehicles moving and stopping at intersections in Fig. 3.

1) Moving at Intersections: In this case, the cloud server uses the vehicle's acceleration information to predict its maneuver. As shown in Fig. 4, X denotes the location of the intersection, and Y is the location for the server to predict the vehicle's maneuver. Acceleration data $\overline{\boldsymbol{a}} = \{a(1), a(2), \ldots, a(N)\}$ collected during the vehicle moving from Z to Y is a sequence of acceleration data to be analyzed. Correspondingly, denote t_p as the prediction lead time in which vehicle moves from Y to X, and t_i as the collection lead time in which $\overline{\boldsymbol{a}}$ is gathered. We will predict the vehicle's maneuver ahead of t_p time at Y,

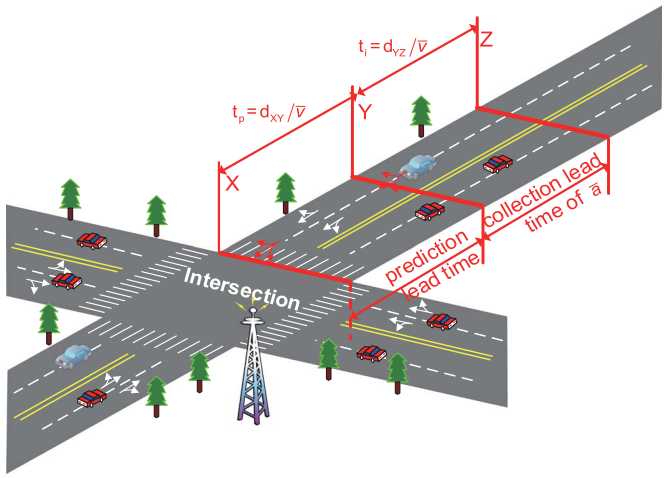


Fig. 4. Schematic of moving at intersections.

which actually occurs at X through processing $\overline{\boldsymbol{a}}$ collected in period t_i, thus reducing the user's waiting time.

To predict the vehicle's maneuver, we need to extract features of $\overline{\boldsymbol{a}}$ and then classify the extracted features into the three categories of turning left, turning right, and moving straight. Since $\overline{\boldsymbol{a}}$ is time-series data, we utilize wavelet analysis [25] to process it, which turns out to be effective. Wavelet analysis provides frequency-domain information of acceleration in different scales that can serve as the features to distinguish one kind of maneuver from another. We also have considered other method such as fractal theory; however, wavelet analysis is more efficient for our system. To the best of our knowledge, this is the first time wavelet analysis is applied to vehicle maneuver prediction. Our experiment results show that the accuracy of prediction is 95.1% on average. Specifically, the wavelet transformation of $\overline{\boldsymbol{a}}$ is

$$\mathrm{DWT}_{m,n} = \langle a(t), \Psi_{m,n} \rangle \tag{1}$$

where $\langle , \rangle$ is the inner product, and

$$\Psi_{m,n}(t) = a_0^{-m/2} \Psi \left(a_0^{-m} t - n b_0 \right).$$

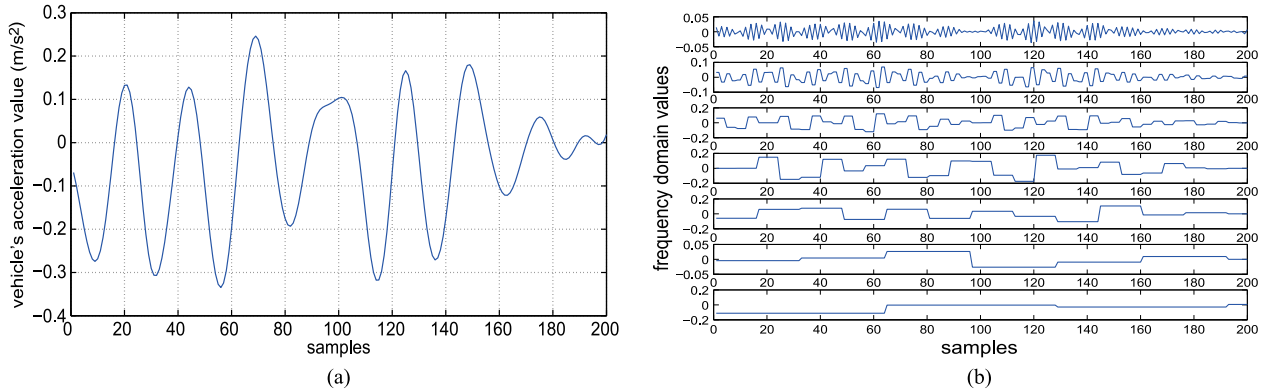


Fig. 5. Acceleration data and wavelet transform results. (a) Raw acceleration data. (b) Different scales in wavelet transform.

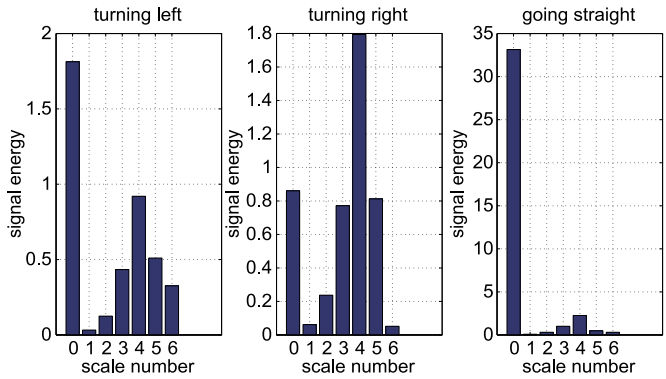


Fig. 6. Signal energy in different scale.

The expression $\Psi(a_0^{-m}t - nb_0)$ is known as Daubechies wavelet, more specifically "db1" wavelet; $a_0 = 2$; and $b_0 = 1$.

Our experiment results show that the six-scale discrete wavelet transform presents an outstanding performance of feature classification, where $m = 0, 1, \ldots, 6$ for $n = 0, 1, 2, \ldots, N$. Fig. 5 shows the original collected acceleration data, where $N = 200$, and the corresponding results for discrete wavelet transform when the vehicle is going to turn right. We can obtain that with the increase in wavelet scale, the variation tendency of the signal becomes increasingly obvious. Then, we acquire the energy of each signal $\mathrm{DWT}_{m,n}$, which is calculated as

$$E_m = \sum_{n=0}^{N} \|\mathrm{DWT}_{m,n}\|^2. \tag{2}$$

The corresponding energy eigenvector $\boldsymbol{E} = [E_0, E_1, \ldots, E_6]$ is in fact the feature of $\overline{\boldsymbol{a}}$. In Fig. 6, a typical signal energy in different scales for the corresponding three kinds of maneuvers are illustrated. Apparently, the characteristics for different maneuvers vary distinctly, which makes the inference process much more differentiable.

After obtaining these features, we implement k-nearest neighbor (KNN) with $k = 3$ to construct the classifier and input the training features sets to train the classifier. Finally, we leverage the trained classifier to infer the prediction result, which is the vehicle's maneuver.

2) Stopping at Intersections: In this case, a vehicle stops when it approaches the intersection due to the traffic lights. Such an event cannot be predicted by analyzing acceleration data features. We can take advantage of lane-change detection to remove some deviating vehicles. For the sake of clarity, we take roads with two lanes and three lanes as examples. For two-lane roads if locating in the left lane, a vehicle cannot turn right. As for three-lane roads, a vehicle locates in the right lane represents turning right, locates in the middle lane represents going straight, and locates in the left lane represents turning left. Therefore, if we can detect that a vehicle makes a left lane change when approaching an intersection, it will not turn right. Furthermore, when a vehicle runs on a three-lane road, if we detect that the vehicle makes a left lane change for twice when approaching the intersection, it is most likely to turn left. To this end, we carry out short-term lane-change detection when a vehicle is reaching an intersection.

Suppose upon a vehicle is L_o away from the nearest intersection, the cloud server starts to detect whether the vehicle will change lane using acceleration data $\boldsymbol{a}'$ in the range of a sliding window W'. If the vehicle does not change lane when it is L_d away from the intersection, stop the lane-change detection since the vehicle is too close to make a lane change. Here, we denote L_o and L_d as window distance and decision distance, respectively. Both L_o and L_d are learned from vehicle's lane change behavior. Specifically, define the lane change distance L_c as the radial distance along moving direction during a vehicle changing lane. Among several lane change behaviors, the minimum lane change distance is L_{cmin} and the maximum lane change distance is L_{cmax}. Then

$$L_o = 2 \times L_{\mathrm{cmax}},\ L_d = L_{\mathrm{cmin}}. \tag{3}$$

Before detecting a vehicle changing lane, filter out high frequency noise of the acceleration data, and get local extremum, then perform interpolation to extract its upper envelope. Typical envelopes of left lane change and right lane change are shown in Fig. 7.

According to the practical pattern of $\boldsymbol{a}'$, we judge vehicle changing lane when its acceleration satisfy the following conditions.

- In W', the maximum value $a'_{\max}$ is not less than a defined threshold a'_t: $a'_{\max} \geq a'_t$.

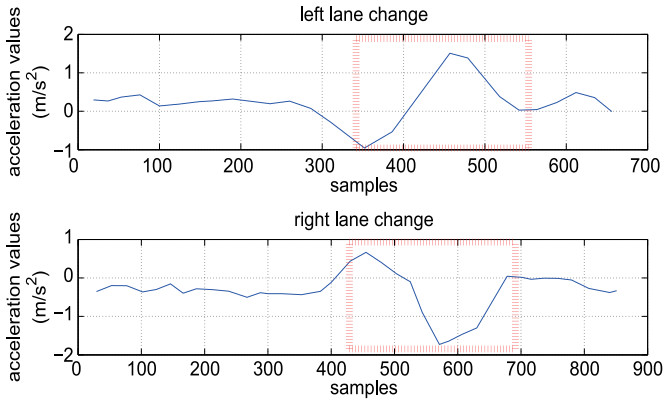


Fig. 7. Acceleration envelope of left and right lane changes.

- Difference between the maximum value $a'_{\max}$ and the minimum value $a'_{\min}$ is not less than a preset value d'_t: $a'_{\max} - a'_{\min} \geq d'_t$.
- Counting from beginning to end of $\boldsymbol{a}'$ in W', the location $L_{\max}$ of $a'_{\max}$ satisfies: $pW' \leq L_{\max} \leq W'$.

Here, a'_t and d'_t are learned in the training stage, where a'_t and d'_t are the least of all $\boldsymbol{a'_{max}}$ and $\boldsymbol{d'_{max}}$, respectively. p is suggested to be assigned value in range $[0.2, 0.3]$. If we detect a vehicle making a lane change, we will utilize the support vector machine (SVM) to tell left lane change from the right lane change.

If the lane change does not help, when the vehicle is passing through the intersection, its maneuver can be judged by reading the orientation data $\boldsymbol{O_r}$ obtained from the smartphone sensor, which also helps to select the potential vehicles. We set a sliding window W forwarding one element over time when new sampled orientation data are generated and calculate the variance of acceleration in W, which is termed as $D(\boldsymbol{O_r})$. If variance $D(\boldsymbol{O_r})$ exceeds a threshold D_t, we consider that the vehicle is making a turn. The orientation data ranges from 0 to 360° and may jump from 0 to 360 when the vehicle heads toward the north direction and makes a turn simultaneously; therefore, it is necessary to minus 360° called orientation compensation when such jumping occurs. After detecting the vehicle making a turn and compensating orientation jumping, the cloud server will check the orientation change to differentiate left turns from right turns. We illustrate the process as follows.

- If $D(\boldsymbol{O_r}) < D_t$ at intersections, the vehicle goes straight.
- If $D(\boldsymbol{O_r}) \geq D_t$ at intersections, compensate orientation jumping if necessary. Denote the average of orientation data before and after turning as O_b and O_a, respectively:
  - — If $O_a - O_b \geq O_t$, the vehicle turns right;
  - — If $O_a - O_b \leq -O_t$, the vehicle turns left.

Here, O_t is the threshold for judging the event of turning left or right with unit degree. D_t and O_t are suggested to be assigned 225° and 15°, respectively.

Based on the given analysis, we present our VMPA in Algorithm 1, which outputs whether the vehicle is turning left, turning right, or going straight. Lines 1–6 are to detect the vehicle's distance to the incoming intersection. If the vehicle arrives at Z, collect its acceleration data until it arrives at Y in Lines 8–9. Lines 10–13 first process the vehicle's acceleration data and then train the classifier and infer the maneuver result. If the vehicle stops at the intersection, we first determine whether we can eliminate vehicles through lane-change detection in Lines 15–20. If not, we collect the orientation data to judge its maneuver in Lines 21–29.

Algorithm 1: Vehicle Maneuver Prediction Algorithm (VMPA)

```
1  Detect the vehicle's distance L to intersection;
2  if vehicle moves at intersections then
3      Obtain d_XY, d_YZ from GPS;
4      Calculate t_p, t_i;
5      if L > d_XY + d_YZ then
6          Keep detecting;
7      else
8          while L > d_XY do
9              Collect ā;
10         Carry out wavelet transform according to Eq. (1);
11         Obtain E according to Eq. (2);
12         Implement KNN to E to train classifier;
13         Leverage the trained classifier for prediction;
14 else
15     Detect the vehicle's distance L to intersection;
16     if L < L_o then
17         Open the sliding window W';
18         while L > L_d do
19             Detect lane change;
20             Eliminate vehicle from list if deviating;
21         if can not eliminate vehicle from list through lane
           change detection then
22             Open the sliding window W for O_r;
23             if D(O_r) < D_t then
24                 vehicles goes straight;
25             else
26                 if O_a − O_b ≥ O_t then
27                     vehicle turns right;
28                 else if O_a − O_b ≤ −O_t then
29                     vehicle turns left;
```

B. Vehicle Tracking

We track vehicles to find out whether they finally arrive at the requested location. To this end, we build a searching tree to manage the potential vehicle list. Considering the example shown in Fig. 2, we take A and B as tree roots, which are the two nearest intersections to the requested location D. Suppose the acceptable waiting time of the user initiating the view-sharing request is $T_{\max}$. If the vehicle theoretically takes at most $T_{\max}$ to reach D according to the road's velocity distribution, all possible paths between the vehicle's current location to D are considered tree branches. After a tree is built in this way, the cloud server only needs to monitor vehicles on the

tree to see if they will pass by D eventually. Such vehicles are added to the potential vehicle list. Based on the previous vehicle maneuver prediction scheme, vehicles initially moving away from D will be eliminated from the list, and then, we can get the final selected vehicles which will go to the requested location.

- Implement VMPA to predict the vehicle's maneuver in advance and eliminate the deviating vehicles.
- Assume there is a virtual vehicle at the searching tree's leaf node at the very beginning. It heads toward the tree's root along branches with the road's average velocity. If the potential vehicle falls far behind the virtual vehicle, eliminate it from the potential vehicle list.

Since we have discussed VMPA in detail in Section IV-A, we just need to explain the process of eliminating a vehicle that falls far behind the virtual vehicle. Still take the example illustrated in Fig. 2, suppose there are m intersections around the requested location D that are denoted $n_1, n_2, \ldots, n_m$. For each road between two adjacent intersections, we divide it into several segments with length equaling to ΔL. If the last part does not reach ΔL, we also count it as a segment. As for each vehicle, we utilize a tuple (n_i, n_j, k, d) to represents its location information, where n_i, n_j indicate the road intersections, and the vehicle moves from n_i to n_j. k denotes the road segment number counting from n_i, and d represents the distance between the vehicle and the beginning of the segment. Assume that a vehicle V is initially located at position P, as does its virtual vehicle V'. V' heads toward D, and moves with the road's average velocity. Every Δt time, the cloud server will check the location of V and calculate the location of V' so that the data traffic for trajectory tacking is sparse enough. Given that V' arrives at a location $P' = (n'_i, n'_j, k', d')$ after several Δt rounds, If V is in the range from $(n'_i, n'_j, k', 0)$ to D, then V does not fall far behind or V will be removed from the potential vehicle list.

Overall, we illustrate the vehicle tracking process in Algorithm 2. Lines 1–3 check if there are vehicles around the requested location and then send a request to these vehicles for uploading pictures. Otherwise, we put vehicles near the requested location into the potential vehicle list in Lines 5–6 and track these vehicles to see whether they move away from D or fall behind in Lines 7–13.

Algorithm 2: Vehicle Trajectory Tracking Algorithm (VTTA)

```
1  Collect vehicle's GPS data near the requested location;
2  if Vehicle at the requested location then
3      Add to final potential list [N];
4  else
5      Search vehicles around the requested location;
6      Generate the potential vehicle list [V];
7      while [V] is not empty do
8          Track vehicles in [V];
9          Eliminate the deviating or falling behind vehicles
           from [V];
10     if [V] is empty then
11         Inform the user road picture is not available;
12     else
13         [N] ← [V];
14 Send request to [N] and determine the payments;
```

V. Picture Selection and Payment Calculation

The cloud server will send a request to the selected vehicles $[N]$ for uploading road pictures. However, vehicles are unwilling to upload pictures without any compensation. To incentivize vehicles to upload road pictures, we design a simple but effective incentive mechanism $\mathcal{M}(\boldsymbol{c})$ based on the VCG auction [26], [27], where the quality of uploaded picture is taken into consideration. Supposing the vehicle $i, i \in [N]$ claims price b_i for picture x_i, then the cloud server will determine a winning set $[N'] \subset [N]$ to upload pictures and pay p_i to $i \in [N']$. We assume that the job of uploading the picture x_i is associated with a cost c_i, which is the private information of i. Denote $q(x_i)$ as the quality of x_i evaluated by our proposed image processing flow, which will be discussed in detail in Section VI-B. We can obtain the value $v([N'])$ of all uploaded pictures defined as a diminishing return, which is frequently utilized in many previous works such as [17], [20]

$$v\left([N']\right) = \lambda \ln\left(1 + \sum_{i \in [N']} q(x_i)\right) \tag{4}$$

where λ is a system parameter. The cloud server's utility u_c is the difference between the value of collected pictures and the corresponding payments, i.e.,

$$u_c = v\left([N']\right) - \sum_{i \in [N']} p_i(b_i, c_{-i}) \tag{5}$$

where $p_i(b_i, c_{-i})$ is the payment to vehicle i when its bid strategy is b_i, and other's bid strategy profile is c_{-i} for $[N] \setminus \{i\}$. For vehicle i, its utility u_i is

$$u_i(b_i, c_{-i}) = \begin{cases} p_i(b_i, c_{-i}) - c_i, & \text{if } i \in [N'] \\ 0, & \text{otherwise} \end{cases} \tag{6}$$

the overall social welfare $S([N])$

$$\begin{aligned} S\left([N]\right) &= \sum_{i \in [N]} u_i + u_c \\ &= \lambda \ln\left(1 + \sum_{i \in [N']} q(x_i)\right) - \sum_{i \in [N']} c_i. \end{aligned} \tag{7}$$

Note that the cloud server cannot get access to the cost c_i of vehicle i; therefore, it treats the claimed price b_i as the corresponding cost c_i. Consequently, we need to rewrite (7) in terms of b_i, i.e.,

$$S\left([N]\right) = \lambda \ln\left(1 + \sum_{i \in [N']} q(x_i)\right) - \sum_{i \in [N']} b_i. \tag{8}$$

We aim to maximize the social welfare in (8) and meanwhile expect $\mathcal{M}(\boldsymbol{c})$ to satisfy several favored properties.

Truthful: For every winning vehicle $i \in [N']$ with cost c_i and claimed price b_i, it cannot gain a larger utility by claiming a higher or lower price than its cost c_i

$$u_i(b_i, c_{-i}) \leq u_i(c_i, c_{-i}) \tag{9}$$

definition of $u_i(c_i, c_{-i})$ is similar to $u_i(b_i, c_{-i})$ in (6).

Individual Rational: For any vehicle i uploading picture x_i, the corresponding utility is greater than 0

$$u_i = p_i - c_i \geq 0. \tag{10}$$

Denote $[N_{-i}]$ and $[N'_{-i}]$ as the vehicle set and the winner set when vehicle i is excluded, respectively. We can obtain the maximal social welfare $S([N'_{-i}])$ for $[N_{-i}]$. We describe the detailed quality-based auction $\mathcal{M}(\boldsymbol{c})$ in Algorithm 3. Line 1 selects subset $[N']$ to maximize social welfare, and Lines 4–7 demonstrate that p_i is calculated according to the impact of i to social welfare.

Algorithm 3: Quality Based Auction

Input: Final potential vehicle set $[N]$, Bid $\{(b_i, x_i)|i = 1, 2, ..., N\}$

Output: Winning vehicle set $[N']$, Payment $p_i, i = 1, 2, ..., N$

1 $[N'] \leftarrow \arg\max_{[N']} \lambda \ln(1 + \sum_{i\in[N']} q(x_i)) - \sum_{i\in[N']} b_i$;
2 **for** $i \in [N]$ **do**
3 $\quad p_i = 0$;
4 **for** $i \in [N']$ **do**
5 $\quad [N_{-i}] \leftarrow [N] \setminus \{i\}$;
6 $\quad [N'_{-i}] \leftarrow \arg\max_{[N'_{-i}]} \lambda \ln(1 + \sum_{j\in[N'_{-i}]} q(x_j)) - \sum_{j\in[N'_{-i}]} b_j$;
7 $\quad p_i = S([N']) - S([N'_{-i}]) + b_i$;

Next, we will analyze the properties of the proposed quality-based auction $\mathcal{M}(\boldsymbol{c})$.

Theorem 1: Quality-based auction $\mathcal{M}(\boldsymbol{c})$ is truthful.

Proof: We will show that each vehicle has no incentive to claim a wrong cost. For a vehicle, there are four outcomes.

Case 1: Vehicle i wins the auction when claiming $b_i = c_i$ and still wins when claiming $b_i \neq c_i$. In this case, the payment is

$$\begin{aligned} p_i &= S([N']) - S\left([N'_{-i}]\right) + b_i \\ &= \lambda \ln\left(1 + \sum_{i\in[N']} q(x_i)\right) - \sum_{[N']\setminus\{i\}} b_i - S\left([N'_{-i}]\right) \end{aligned}$$

which is irrelevant to b_i, and vehicle i cannot gain higher utility when lying for cost.

Case 2: Vehicle i wins the auction when claiming $b_i = c_i$ but loses when claiming $b_i \neq c_i$. In this case, the vehicle i gets a utility $u_i \geq 0$ when winning the auction but receives $u_i = 0$ when losing the auction. Therefore, we obtain $u_i(b_i, c_{-i}) \leq u_i(c_i, c_{-i})$.

Case 3: Vehicle i loses the auction when claiming $b_i = c_i$ but wins when claiming $b_i \neq c_i$. It demonstrates that $i \notin [N']$ when $b_i = c_i$; therefore, its utility $u_i = 0$. If i claims $b_i \neq c_i$ and wins the auction, then utility u_i is

$$\begin{aligned} u_i &= S([N']) - S\left([N'_{-i}]\right) + b_i - c_i \\ &= \lambda \ln\left(1 + \sum_{i\in[N']} q(x_i)\right) - \sum_{j\in[N']\setminus\{i\}} b_j - c_i \\ &\quad - S\left([N'_{-i}]\right) \\ &= S([N']) - S\left([N'_{-i}]\right) \\ &\leq 0 \end{aligned}$$

where $S([N'])$ is social welfare on $[N']$ when i claims truthful price. The last inequality holds because when $b_i = c_i$, vehicle i loses the auction, and $S([N])$ is maximized when winning set is N'_{-i}. Therefore $u_i(b_i, c_{-i}) \leq u_i(c_i, c_{-i})$.

Case 4: Vehicle i loses the auction when claiming $b_i = c_i$, and still loses when claiming $b_i \neq c_i$. Therefore, the utility $u_i(b_i, c_{-i}) = u_i(c_i, c_{-i}) = 0$, which implies $u_i(b_i, c_{-i}) \leq u_i(c_i, c_{-i})$.

Above all, the quality-based auction $\mathcal{M}(\boldsymbol{c})$ is truthful. ■

Theorem 2: Quality-based auction $\mathcal{M}(\boldsymbol{c})$ is individual rational.

Proof: If vehicle i loses, then $p_i = 0$, and utility $u_i = 0$. Therefore, we just consider the case where vehicle i wins the auction. According to theorem 1, vehicle i will claim price b_i as its cost c_i for higher utility. $\forall\, i \in [N']$, the utility u_i is

$$\begin{aligned} u_i &= p_i - c_i \\ &= S([N']) - S\left([N'_{-i}]\right) + b_i - c_i \\ &= S([N']) - S\left([N'_{-i}]\right) \\ &\geq 0 \end{aligned}$$

the last inequality holds because, in the quality-based auction, we select the subset $[N']$ that maximizes overall social welfare. ■

Theorem 3: The utility of cloud server $u_c \geq 0$.

Proof: According to (5), we obtain that

$$\begin{aligned} u_c = &\ v([N']) - \sum_{i\in[N']} p_i(b_i, c_{-i}) \\ = &\ v([N']) - \sum_{i\in[N']} \left(S([N']) - S\left([N'_{-i}]\right) + b_i\right) \\ \geq &\sum_{i\in[N']} \left(v([N']) - v([N'] \setminus \{i\})\right) \\ &- \sum_{i\in[N']} \left(S([N']) - S\left([N'_{-i}]\right) + b_i\right). \end{aligned}$$

The last inequality holds because $v([N']) = \sum_{i=2}^{N'} (v([i']) - v([(i-1)']))$, and $\lambda \ln(1 + x)$ is concave. Thus, $\lambda \ln(1 + x_1 + \Delta x) - \lambda \ln(1 + x_1) \geq \lambda \ln(1 + x_2 + \Delta x) - \lambda \ln(1 + x_2)$ when $x_1 < x_2, \Delta x > 0$. Then, we get

$$\begin{aligned} u_c &\geq \sum_{i\in[N']} \left(\sum_{j\in[N']} b_j + S\left([N'_{-i}]\right) - v([N'] \setminus \{i\}) - b_i\right) \\ &= \sum_{i\in[N']} \left(S\left([N'_{-i}]\right) - S([N'] \setminus \{i\})\right) \\ &\geq 0. \end{aligned}$$

(a) (b)

Fig. 8. Image matching example. (a) Image of some place. (b) Corresponding Baidu street view image.

The last inequality holds because $S([N'_{-i}])$ is maximized when i is excluded. ■

Moreover, the cloud server's profit does not only depend on the pictures' value, and apps such as Waze can make profit by advertisement and other means.

VI. Processing of the Real-Time Road Images

Each winning vehicle will upload a road picture through cellular networks, but we only need qualified ones. Road pictures do not have to be high resolution and should be evaluated from the following aspects: 1) location where the road picture is taken (the distance from the camera location to the requested location should be close enough); 2) content of road picture, (there may be completely unrelated pictures, such as an all black picture, because the camera is obscured); 3) the shooting angle of road picture (a picture taken off the road or deviated from the road direction is unqualified since it cannot reflect the required road information); and 4) the picture sharpness (blurred pictures clearly do not meet the requirements). We neglect the time factor because taking a picture is associated with a timestamp, which can verify the picture's real-timeliness.

It is difficult to directly evaluate a road picture. Since existing Baidu street view images have already perfectly satisfied all the above requirements and cover all the major cities with 360° views, an efficient method is to compare the uploaded pictures with Baidu street view images of the corresponding positions. The more matches they have, the more likely the uploaded picture is qualified. However, Baidu street view images have been processed so that their brightness and hue are very different from the real situation. We leverage the RootSIFT [23], [24] algorithm to match the image that can yield more matches and then evaluate picture quality according to the matching result.

A. Image Matching

To illustrate the image matching process, we use two images in Fig. 8 as an example. Fig. 8(a) is an image taken at an intersection, and Fig. 8(b) is the corresponding Baidu street view image. The first step is to use RootSIFT feature detector to detect feature points in both images and compute their SIFT descriptors. For each feature point in the first image, we utilize KNN $(k = 2)$ to find two best matches in the second image based on the distance between descriptors and *vice versa* for the second image. If the Euclidean distance between a feature descriptor and its best match is much smaller than the second best match, the first match is very likely to be a correct match. Thus, we accept the best match and reject the second best match. On the contrary, if the two best matches are at similar distances to the feature descriptor, they may both be wrong matches since a good match is distinctive; then, we reject both matches. We carry out this process by checking whether the ratio between the distance to the best match and the distance to the second best match is smaller than a threshold. In practice, we find 0.75 to be a proper threshold that can reject incorrect matches while keeping correct ones. Now, we have two match sets: one from the first image to the second image and the other from the second image to the first image. A good match of two feature points should be symmetrical, which means they should be the best matching point of the other in both match sets.

Corresponding points in two image planes should obey the epipolar constraint, which means the match of a given point in another image plane must lie on this point's epipolar line. The epipolar constraint between images can be represented by the fundamental matrix that can be calculated through RANSAC method [5] from the match sets. Mathematically, for a feature point, if the distance between its corresponding point and its epipolar line exceeds the threshold, this match pair should be rejected. Therefore, the last step is to obtain the fundamental matrix through RANSAC method and reject the matches which do not obey the epipolar constraint. We always select the one who has the largest number of matches with its corresponding Baidu Street view image and send it back to the requested user. However, we should first set an evaluation criterion, which is to decide whether the road picture meets basic requirements. Otherwise, if all picture candidates are not qualified, even the one with the largest number of matches should not be accepted.

B. Picture Quality

Baidu street view was established a few years ago; therefore, the street view images may much differ from off-the-shelf real road scenes. Each time the cloud server will select the best road picture, which is then transmitted to the user and stored in the cloud server as a standard image for future use. The next time a user requests a road picture of the same place, candidate pictures will be matched with the standard image instead of the Baidu street view. This process can ensure that the reference images for evaluation also stay a certain degree of consistency. Thereby, we should consider two cases: matching with street view and matching with standard road images. Experiment results also show that the number of matches with street view image is generally much less than with standard image so that the evaluation criteria for the two cases should be different.

For the case of matching with standard road images, using the number of match points to determine whether the image meets basic requirements is not a reliable approach. This is because uploaded images' sizes, pixels, and contents may be different, and it is difficult to identify a specific threshold to distinguish qualified pictures from unqualified pictures. For example, since the feature point is often extracted from buildings, road signs, street lamps, and other objects, if there are few buildings in the image, the number of matches for feature points will be small. With such an observation, we use the ratio between the number of matches and the number of detected feature points (if the

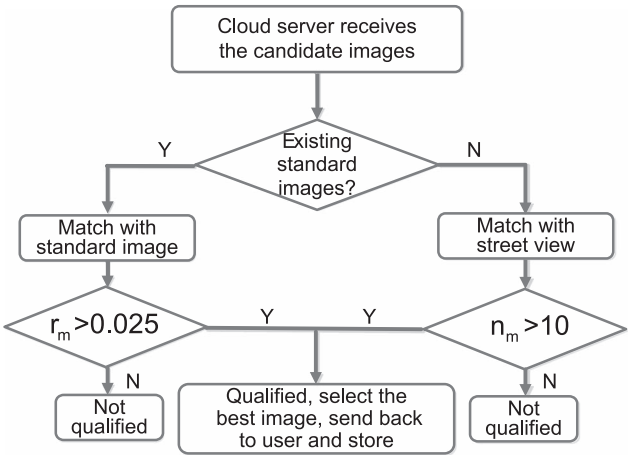


Fig. 9. Evaluation process.

TABLE I
RESULTS OF VEHICLE MANEUVER PREDICTION ACCURACY

maneuver	turn right	turn left	go straight
Accuracy	100%	84.6%	100%

number of detected feature points in two pictures are not equal, use the smaller one) as the criterion and set a corresponding threshold α_r. According to experiment results, we set α_r as 0.025. Images whose ratio between the number of matches and the number of feature points is greater than 0.025 are considered qualified road images. As for matching with Baidu street view, we use the number of match points as the criterion because we can only obtain a small number of matches. Based on the experiment results, we find images with more than ten matches with the street view images are qualified to reflect the road information; therefore, threshold $\alpha_n = 10$. The thresholds α_r and α_n are set the values according to the image matching tests when we consider that the uploaded images can fundamentally reflect real road conditions for the two different cases.

Taking two different cases into consideration, we define the quality of picture x_i separately as follows:

$$q(x_i) = \begin{cases} \frac{r_m}{\alpha_r}, & \text{matching with standard image} \\ \frac{n_m}{\alpha_n}, & \text{matching with street view} \end{cases} \tag{11}$$

where r_m is the ratio between the number of matches and the number of detected feature points, and n_m is the number of matches. The whole process of evaluation is shown in Fig. 9.

VII. EXPERIMENTS AND SIMULATION

We fix a NEXUS 4 smartphone in front of the car to collect acceleration, GPS, and orientation data. The sampling rate of the smartphone is 50 Hz. Acceleration, GPS, and orientation data are used for the vehicle maneuver prediction. Meanwhile, we take road pictures and match them with the corresponding Baidu street view for quality evaluation. Moreover, we carry out extensive simulation for vehicle tracking and the proposed quality-based auction by utilizing real data sets.

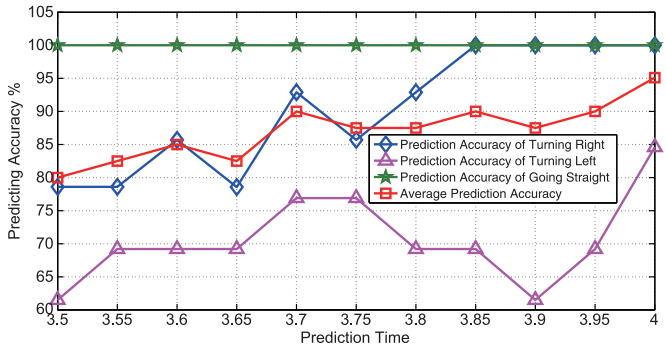


Fig. 10. Accuracy prediction lead time diagram.

TABLE II
RESULTS OF LANE-CHANGE DETECTION ACCURACY

lane change	right	left
Accuracy	100%	100%

TABLE III
RESULTS OF VEHICLE MANEUVER DETECTION ACCURACY

maneuver	turn right	turn left	go straight
Accuracy	100%	95%	100%

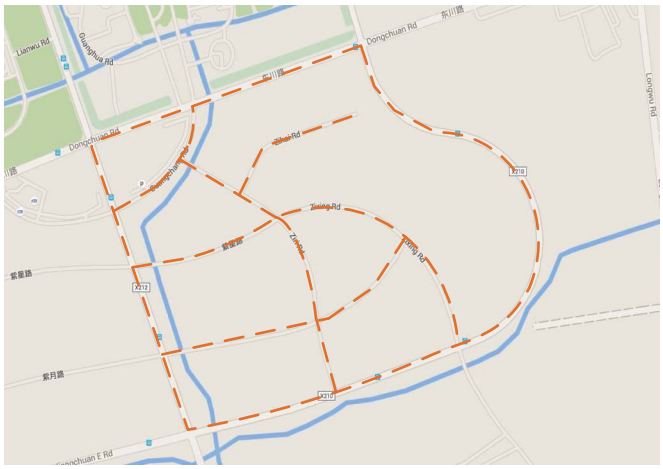


Fig. 11. Simulation map of VTTA.

A. Vehicle Selection Experiments

The test for the vehicle selection can be divided into two parts: VMPA in Section IV-A and VTTA in Section IV-B. With regard to VMPA, we collect real acceleration and orientation data of moving vehicles to analyze its performance. As for VTTA, we take advantage of real road parameters to design the simulations.

1) VMPA Experiments: In the experiment of VMPA, we use acceleration data to predict the vehicle's maneuver when it moves at intersections while utilizing orientation data to detect the vehicle's maneuver when it stops at intersections. We collect 56 sets of acceleration data for the vehicle maneuver prediction in which 19 are set for turning right, 19 are set for turning left, and 18 are set for going straight, respectively. Meanwhile, we also gather the corresponding 56 sets of orientation data when the vehicle turns left, turns right, or goes straight at intersections

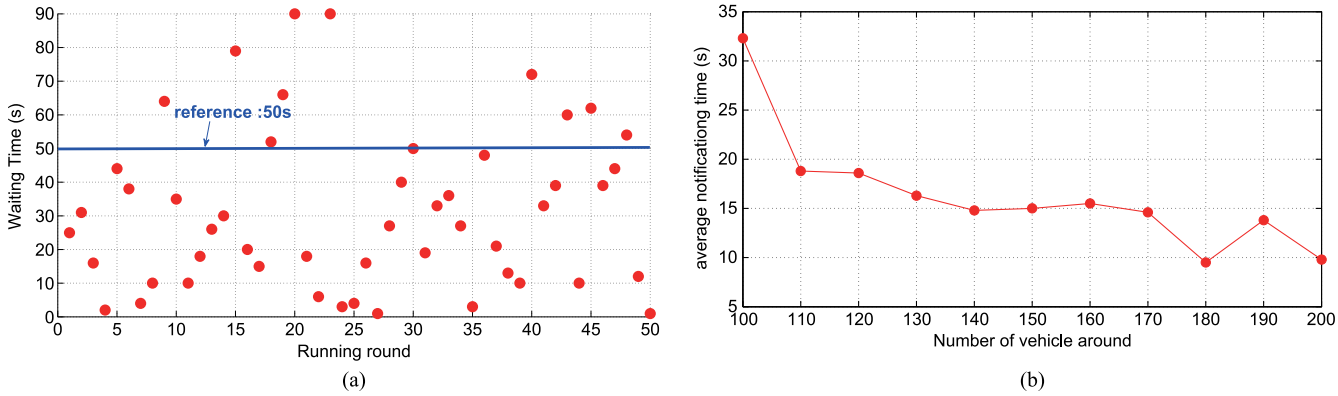


Fig. 12. Notification time. (a) Scattering diagram of the notification time. (b) Impact of the number of vehicles for notification time.

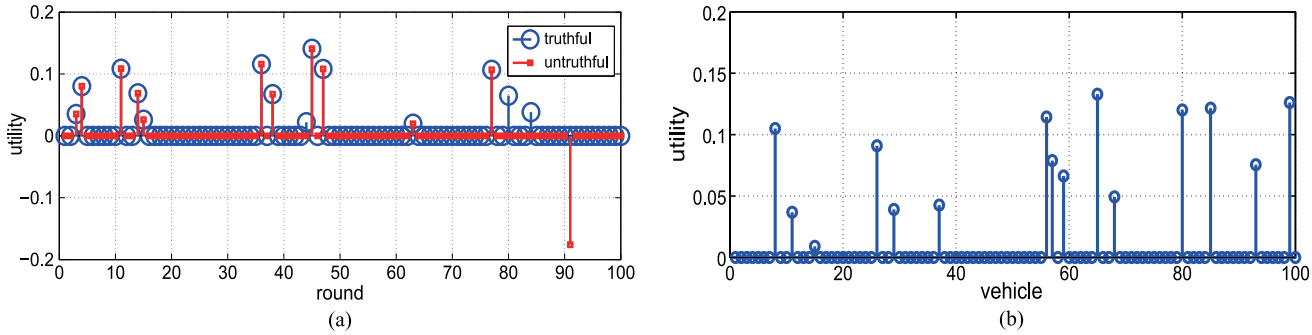


Fig. 13. Economic properties. (a) Truthfulness. (b) Individual rationality.

to test detection accuracy. For the case where vehicles stop at intersections, we collect 29 sets of acceleration data in which nine are set for left lane change, 11 are set for right lane change, and the other nine are set for no lane change.

When vehicles move at intersections, we set the prediction lead time t_p as 4 s and the collection lead time t_i as 2 s. Then, we implement KNN and choose five sets of acceleration data for each maneuver as the training sets to train the classifier. After the training process, we input the rest of the acceleration data to the classifier and get prediction accuracy results shown in Table I. The prediction accuracy is the ratio between the number of correct prediction and the number of total tests. According to Table I, we can achieve at least 84.6% prediction accuracy and 95.1% on average with 39 correct times out of 41 total times.

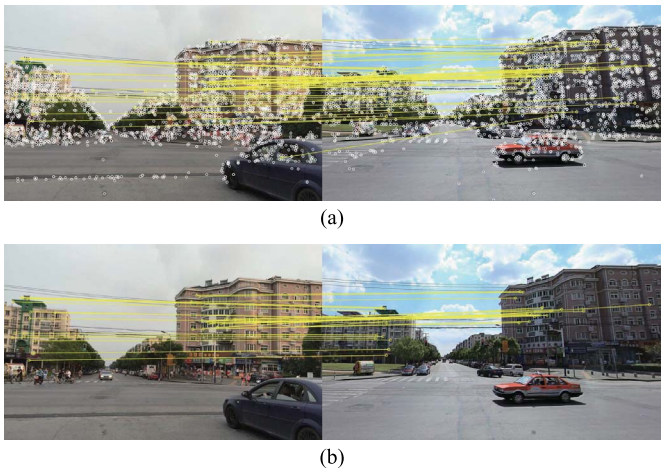
We also consider the GPS localization error, which affects the calculation of t_p and t_i according to Fig. 4, to test the robustness of VMPA. We range t_p from 3.5 to 4 s with an interval of 0.05 s each time. Therefore, we can get 11 groups of prediction accuracy results, which are drawn in Fig. 10. We can see that, even if the error of prediction lead time reaches 0.5 s, which means 6.9 m of GPS localization error when $\overline{v} = 50$ km/h, the average accuracy still maintains 80%. Therefore, our proposed vehicle prediction scheme is robust for localization error, which is several meters in most times. Fig. 10 also shows that acceleration data collected ahead of 4 s is more representative for vehicle maneuver prediction.

When vehicles stop at intersections, we first attempt to remove deviating vehicles through lane-change detection, and the accuracy of lane-change detection is shown in Table II. We carry out detection experiments of the vehicle's maneuver when it passes through the intersection. We utilize $\boldsymbol{O_r}$ to judge whether the vehicle turns left, turns right or goes straight with the results shown in Table III. From Table III, we obtain that the detection process is highly reliable with 96.4% accuracy on average.

2) VTTA Simulation: We use parameters of real town roads, as shown in Fig. 11, to simulate VTTA and test its efficiency.

First we build a searching tree structure for Fig. 11 and set the user's acceptable time as 100 s and the number of vehicles around as 100, where the vehicles randomly turn right, turn left, and go straight at intersections. Then, we run the simulation for 50 times to obtain the user's actual notification time defined as the time interval between the time a user sends its request and the time the cloud server predicts the first vehicle will pass by the requested location. We draw the results in Fig. 12(a). We can see that most of the points lie below the reference line of 50 s, which means the user's notification time is usually within 50 s.

In addition, we also investigate the impact of the number of vehicles around and increase the number from 100 to 200 with an interval of 10 each time. To alleviate randomness, we run the algorithm ten times for each vehicle number case and get the mean value. Fig. 12(b) demonstrates the results which reflect that, with the increase in the vehicle number, the average notification time tends to decrease. The results also show that prediction of vehicle's maneuver can reduce the notification time significantly.



(a)

(b)

Fig. 14. Image matching results. (a) Image matching result after ratio test and symmetry test: 54 matches. (b) Image matching result after RANSAC filtering: 30 matches.

B. Quality-Based Auction Simulations

Suppose the number of selected vehicles is $N = 100$, and their claimed prices are uniformly distributed in $[0, 1]$ with corresponding picture quality ranging in $[3, 4]$ uniformly. λ is set as 2.

1) Truthfulness: We randomly select a vehicle in $[N]$, e.g., vehicle 1, and let it randomly chooses a price in $[0.5b_1, 2b_1]$. We run Algorithm 3 for 100 times, and compare the utility between truthfully and untruthfully claiming price, which is shown in Fig. 13(a). We can see that the vehicle can always gain higher utility when truthfully claiming than lying.

2) Individual Rationality: Then, we plot the utility of each vehicle in Fig. 13(b). According to Fig. 13(b), we find that $u_i \geq 0 \ \forall \, i \in [N]$.

C. Image Matching Simulations

We match the road picture with the corresponding Baidu street view. Initially, we detect 3094 feature points in Fig. 8(a) and 2999 feature points in Fig. 8(b). However, after the ratio test and the symmetry test, only 54 matches remain. In Fig. 14(a), the white circles are initially detected feature points, whereas the yellow lines connect matched points. Fig. 14(b) demonstrates the result after RANSAC filtering. Obviously, after RANSAC, many incorrect matches are eliminated with only 30 matches left, which means the picture quality is 3.

VIII. Conclusion

In this paper, we build an RVShare system to acquire visual real-time road information, where the VMPA and VTTA are proposed to predict the vehicle's maneuver and track vehicle, respectively. The combination of vehicle prediction and vehicle tracking can select vehicles for view-sharing job distribution and reduce the user's waiting time. A quality-based auction is designed to incentivize selected vehicles to upload qualified pictures. Furthermore, the uploaded picture is matched with Baidu street view for quality evaluation. Extensive experiments demonstrate that the RVShare system is efficient and robust.

References

[1] A. Ananda, G. Ramaduraia, and L. Vanajakshia, "Data fusion-based traffic density estimation and prediction," *J. Intell. Transp. Syst., Technol., Plan., Oper.*, vol. 18, no. 4, pp. 367–378, 2014.

[2] J. Chen, K. H. Low, Y. Yao, and P. Jaillet, "Traffic state estimation based on data fusion techniques," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 3, pp. 901–921, Sep. 2015.

[3] K. Ali, D. Al-Yaseen, A. Ejaz, T. Javed, and H. S. Hassanein, "Crowdits: Crowdsourcing in intelligent transportation systems," in *Proc. IEEE WCNC*, 2012, pp. 3307–3311.

[4] N. Souliotis, A. Tsadimas, and M. Nikolaidou, "Real-time information about public transport's position using crowdsourcing," in *Proc. ACM PCI*, 2014, pp. 1–6.

[5] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *ACM Commun.*, vol. 24, no. 6, pp. 381–395, 1981.

[6] M. Dubsk, A. Herout, R. Jurnek, and J. Sochor, "Fully automatic roadside camera calibration for traffic surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1162–1171, Jun. 2014.

[7] Y. Jo, J. Choi, and I. Jung, "Traffic information acquisition system with ultrasonic sensors in wireless sensor networks," *Int. J. Distrib. Sens. Netw.*, vol. 16, no. 1, pp. 14 050–14 069, 2014.

[8] W. Z. Khan, Y. Xiang, M. Y. Aalsalem, and Q. Arshad, "Mobile phone sensing systems: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 402–427, 1st Quart. 2013.

[9] G. Arnaud, C. Marie, G. Dominique, and D. Xavier, "A probabilistic approach for data quality assessment of road hazard warnings in crowdsourced driving navigation systems," in *Proc. Transp. Res. Arena*, 2014, pp. 1–10.

[10] T. Hlnhagen, I. Dengler, A. Tamke, T. Dang, and G. Breuel, "Maneuver recognition using probabilistic finite-state machines and fuzzy logic," in *Proc. IEEE IV*, 2010, pp. 21–24.

[11] J. Firl and Q. Tran, "Probabilistic maneuver prediction in traffic scenarios," in *Proc. ECMR*, 2011, pp. 89–94.

[12] E. Ohn-Bar, A. Tawari, S. Martin, and M. M. Trivedi, "Predicting driver maneuvers by learning holistic features," in *Proc. IEEE IV*, 2014, pp. 719–724.

[13] D. Chen, K. T. Cho, S. Han, Z. Jin, and K. G. Shin, "Invisible sensing of vehicle steering with smartphones," in *Proc. ACM MobiSys*, 2015, pp. 1–13.

[14] H. Hajimolahoseini, R. Amirfattahi, and H. Soltanian-Zadeh, "Robust vehicle tracking algorithm for nighttime videos captured by fixed cameras in highly reflective environments," *IET Comput. Vis.*, vol. 8, no. 6, pp. 535–544, 2014.

[15] Y. Xiong, X. Lu, Z. Zhu, and W. Zeng, "Vehicle tracking in video based on pixel level motion vector," *Multimedia Signal Process.*, vol. 246, pp. 200–206, 2012.

[16] I. M. Almomani, N. Y. Alkhalil, E. M. Ahmad, and R. M. Jodeh, "Ubiquitous GPS vehicle tracking and management system," in *Proc. IEEE AEECT*, 2011, pp. 1–6.

[17] D. Yang, G. Xue, X. Fang, and J. Tang, "Crowdsourcing to smartphones: Incentive mechanism design for mobile phone sensing," in *Proc. ACM Mobicom*, 2012, pp. 173–184.

[18] X. Zhang, G. Xue, R. Yu, D. Yang, and J. Tang, "Truthful incentive mechanisms for crowdsourcing," in *Proc. IEEE INFOCOM*, 2015, pp. 2830–2838.

[19] L. Gao, F. Hou, and J. Huang, "Providing long-term participation incentive in participatory sensing," in *Proc. IEEE INFOCOM*, 2015, pp. 2803–2811.

[20] Y. Wen *et al.*, "Quality-driven auction based incentive mechanism for mobile crowd sensing," *IEEE Trans. Veh. Technol.*, vol. 64, no. 9, pp. 4203–4214, Sep. 2015.

[21] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[22] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features," in *Proc. ECCV*, 2006, pp. 404–417.

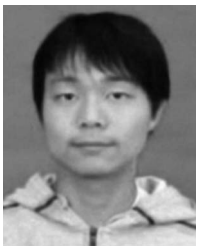
[23] R. Arandjelovic and A. Zisserman, "Three things everyone should know to improve object retrieval," in *Proc. IEEE CVPR*, 2012, pp. 2911–2918.

[24] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1615–1630, Oct. 2005.

[25] T. Amr, *Survey on Time-Series Data Classification*. [Online]. Available: http://tarekamr.appspot.com/pdfs/tsdm2012.pdf

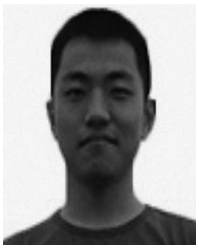
[26] W. Vickrey, "Counterspeculation, auctions, and competitive sealed tenders," *J. finance*, vol. 16, no. 1, pp. 8–37, 1961.

[27] E. Clarke, "Multi-part pricing of public goods," *Public Choice*, vol. 11, no. 1, pp. 17–23, 1971.



Xiong Wang received the B.E. degree in electronic information engineering from Huazhong University of Science and Technology, Wuhan, China, in 2014. He is currently working toward the Ph.D. degree in electronic engineering with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China.

His current research interests include crowdsourcing, network economics, and mobile computing.



Lei Ding received the B.E. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015. He is currently working toward the M.S. degree in electrical engineering with the Department of Electrical Engineering, University of California Los Angeles, Los Angeles, CA, USA.

His research interests include intelligent transportation and mobile computing.

Qi Wang received the B.E. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015. She is currently working toward the M.S. degree in computer science with the Department of Computer Science, Columbia University, New York, NY, USA.

Her research interests include image processing and mobile computing.

Jin Xie received the B.E. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2015. He is currently working toward the M.S. degree in electrical and computer engineering with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, USA.

His research interests include intelligent transportation and mobile computing.

Tianyi Wang received the B.E. degree in electronic engineering in 2014 from Shanghai Jiao Tong University, Shanghai, China, where he is currently working toward the M.S. degree in electronic engineering.

His research interests include dynamic time-division multiple access and mobile computing.

Xiaohua Tian (M'16) received the B.E. and M.S. degrees in communication engineering from Northwestern Polytechnical University, Xi'an, China, in 2003 and 2006, respectively, and the Ph.D. degree from Illinois Institute of Technology, Chicago, IL, USA, in 2010.

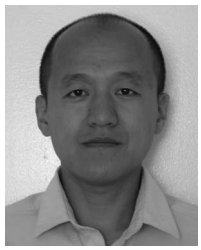
Since June 2013, he has been with the Department of Electronic Engineering, Shanghai Jiao Tong University, as an Assistant Professor with the title of SMC-B Scholar. He is also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing, China.

Dr. Tian has served as a Technical Program Committee (TPC) member for the Wireless Networking, Cognitive Radio Networks, and Ad Hoc and Sensor Networks Symposium of IEEE Global Communications Conference during 2011–2015; as a TPC member for the Ad Hoc and Sensor Networks and Next Generation Networking Symposium of the IEEE International Conference on Communications in 2013 and 2015, respectively; as a Best Demo/Poster Award Committee member of the IEEE International Conference on Computer Communications (INFOCOM), a TPC Cochair for the International Workshop on Internet of Things of the IEEE/CIC International Conference on Communications in China (ICCC), a TPC Cochair for the Ninth International Conference on Wireless Algorithms, Systems, and Applications on Next Generation Networking Symposium and the Local Management Chair for IEEE ICCC in 2014; and as a TPC member for the IEEE International Conference on Computer Communications (IEEE INFOCOM) in 2014 and 2015. He serves as an editorial board member on the computer science subsection of the journal *SpringerPlus* and the guest editor of the *International Journal of Sensor Networks*.

Yunfeng Guan received the Ph.D. degree in information engineering from Zhejiang University, Hangzhou, China, in 2003.

He is currently an Associate Professor with the Institute of Wireless Communication Technology, Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China. He mainly engaged in the study of digital TV systems and wireless access systems. He is also one of the main drafters of the China digital terrestrial television transmission standard GB20600-2006 (DTMB) and other important national DTV standards. He has developed a series of chips for the DTMB demodulator, digital satellite television demodulator, and HD-Decoder SoC. He is the author of more than 20 articles and a holder of 28 patents for inventions. He also finished a series of National and local major research projects.

Dr. Guan received the First Prize Shanghai Science and Technology Development Award in 2005 and 2006, the Second Prize National Science and Technology Development Award in 2008, the Shanghai Venus Award in 2009, and the Shanghai technology Top Ten Youth Award nomination in 2010.



Xinbing Wang (SM'12) received the B.S. degree (with honors) in automation from Shanghai Jiao Tong University, Shanghai, China, in 1998; the M.S. degree in computer science and technology from Tsinghua University, Beijing, China, in 2001; and the Ph.D. degree with a major in electrical and computer engineering and minor in mathematics from North Carolina State University, Raleigh, NC, USA, in 2006.

He is currently a Professor with the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China.

Dr. Wang served as a member of the Technical Program Committees of several conferences, including The ACM Annual International Conference on Mobile Computing and Networking in 2012, The ACM International Symposium on Mobile Ad Hoc Networking and Computing in 2012–2014, and the IEEE International Conference on Computer Communications in 2009–2014. He has served as an Associate Editor for IEEE/ACM TRANSACTIONS ON NETWORKING and IEEE TRANSACTIONS ON MOBILE COMPUTING.