Journal of Systems Architecture 95 (2019) 1–8

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

Lane scheduling around crossroads for edge computing based autonomous driving

Changqing Xia[a,b,c,*], Xi Jin[a,b,c], Linghe Kong[a,d], Chi Xu[a,b,c], Peng Zeng[a,b,c]

[a] State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China
[b] Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences Shenyang, China
[c] Key Laboratory of Networked Control System, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China
[d] Shanghai Jiao Tong University, Shanghai, China

ARTICLE INFO

Keywords:
Edge computing
Autonomous driving
Scheduling
IoT
Crossroads.

ABSTRACT

Autonomous driving, which can free our hands and feet, is of increasing importance in our daily lives. However, the capacity of onboard computation and communication limits the rapid development of autonomous driving. To address this issue, this paper proposes a novel model named the edge computing-based lanes scheduling system (ECLSS) to study lane scheduling for each vehicle around crossroads with real-time edge devices. There are several edge computing devices (ECD) deployed around crossroads in ECLSS that can collect information from vehicles and road conditions with short-range wired/wireless transmissions. Based on the strong computing power of ECDs and their real-time transmission performance, we propose two centralized management lane scheduling methods: the searching for efficient switching algorithm (SESA) and special vehicles lane switching algorithm (SVLSA). These edge computing-based autonomous driving methods aim to achieve high speed passing through crossroads and guarantee that special vehicles can pass through crossroads within a certain time. Extensive simulations are conducted, and the simulation results demonstrate the superiority of the proposed approaches over competing schemes in typical lane switching scenarios.

1. Introduction

Currently, autonomous driving has become quite a hot topic in the research community and in industry. Depending on the collaboration of artificial intelligence, visual computing, radar, monitoring devices and global positioning systems, computers can operate motor vehicles automatically and safely without any human initiative. However, one of the largest barriers to the rapid development of autonomous driving is the limitation of onboard computation and communication. Obviously, upgrading onboard computers is not a good option since it has high equipment cost and energy consumption.

Cloud computing is another possible solution since it can provide services for communication, computing and storage facilities [1,2]. Cloud computing can achieve global scheduling for vehicles with information from vehicles and roadside units (RSUs) collaboratively. Several mobile cloud computing (MCC) frameworks have proposed technology for improving mobile devices with cloud services, such as CloneCloud [3], MAUI [4] and MobiCloud [5]. Furthermore, a vehicular cloud computing (VCC) approach based on MCC that provides driving quality of service (QoS) was proposed [6–8]. However, the response time between the vehicle and the cloud (response time in seconds) is much larger than the corresponding time of the vehicle operation (response time in microseconds) in both MCC and VCC, which are unacceptable in autonomous driving.

Edge computing has been studied as an extension of cloud computing that is capable of providing real-time communication performance and strong computing power at the edge of the network [9,10]. Since the bandwidth of the network has come to a standstill compared with the development of data computing speed, transmission time overhead has become the bottleneck for cloud-based autonomous driving. For example, the Boeing 787 generates 5 GB of data every second [11], and an autonomous vehicle will generate 4 TB of data per day in the near future [12]; however, the bandwidth is not large enough for data transmission (it will take 18 days to upload 4 TB of data with the 20 Mbps LTE network). Hence, the data need to be processed at the edge for a shorter response time, more efficient processing and a smaller network pressure [13]. Several related studies have been proposed for autonomous driving based on the concept of edge computing, such as [14–16]. However, these works have various limitations, such as a lack of lane change method under complex road conditions, and they do not address the

* Corresponding author: Peng Zeng, at Key Laboratory of Networked Control System, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China.
E-mail addresses: xiachangqing@sia.cn (C. Xia), zp@sia.cn (P. Zeng).

https://doi.org/10.1016/j.sysarc.2019.02.017
Received 15 October 2018; Received in revised form 13 February 2019; Accepted 15 February 2019
Available online 18 February 2019
1383-7621/© 2019 Published by Elsevier B.V.

requirement of special vehicles driving on the road. Furthermore, the existing works primarily consider the safety of straight line driving; however, crossroads have the most complex road conditions and require more attention in autonomous driving.

This paper systematically investigates edge computing in autonomous driving to schedule lane switching around crossroads, which is a critical issue in real life but has thus far received little attention in existing works. Furthermore, we also consider the requirement of special vehicles, such as their passing time. The challenges are as follows: (1) we must guarantee the safety of autonomous driving, especially the safety of switching lanes; (2) we want to address the need for vehicles to pass through crossroads quickly, especially special vehicles, such as an ambulance or a police car that must pass through crossroads within a certain time. In this study, we propose two lane scheduling methods around crossroads. We list our contributions as follows:

1. We propose a novel model for edge computing-based autonomous driving. Because of the complexity of traffic situations, our model mainly focuses on lane scheduling around crossroads for both ordinary and special vehicles. Based on the traffic and vehicle information, an edge computing device (ECD) can provide centralized scheduling of vehicle lane switches.
2. We define the safe distance to guarantee a vehicle's safety. Furthermore, we find that the additional time overhead of each switch is irrelevant to the position of the vehicle that switches when there are vehicles in front of it and behind it in the target lane.
3. We obtain the range of the time overhead to pass through the crossroads for any vehicle.
4. We propose two lane switching algorithms. By implementing the searching for efficient switching algorithm (SESA), we reduce the time overhead at crossroads on the premise that all vehicles can reach their target directions. For special vehicles, we then propose the special vehicles lane switching algorithm (SVLSA) to ensure that special vehicles can pass through crossroads before their passing deadlines. We demonstrate that our scheduling algorithms and analysis approach are better than the existing ones with extensive simulations.

The remainder of the paper is structured as follows. Section 2 reviews research on autonomous driving. The system configuration is presented in Section 3, and the problem statement of MEC-based autonomous driving is provided in Section 4. Scheduling algorithms for lane scheduling around the crossroads are proposed in Section 5. Section 6 illustrates the implementations of the proposed methods based on simulations. Conclusions are given in Section 7.

2. Related work

The concept of connected vehicles was proposed in 1996, and many researchers have been working in this area to improve the safety and convenience of driving [17–19]. introduced the concept of autonomous driving by allowing a vehicle to connect to the Internet and share internet access with other equipment. Then, many companies, research institutions and auto vendors showed great interest in autonomous driving [20–23]. Furthermore, some works have surveyed the current state-of-the-art on planning and control algorithms with a particular emphasis on the urban setting [24–26]. However, wireless network bandwidth and real-time performance are often the performance bottlenecks of cloud computing for connected vehicles.

Mobile edge computing (MEC) provides solutions to this dilemma by providing processing from the centralized cloud to the distributed edge devices that are close to the vehicles [15,27,28]. To meet the demands of both communication and computation, Xueshi Hou et al. present an overview of vehicles as the infrastructures for communication and computation in [14]. Considering the features of the cloud and MEC, Kengo Sasaki et al. propose an infrastructure-based vehicle control system that shares internal states between the edge and cloud servers, dynamically allocates computational resources and switches necessary computation on collected sensors according to network conditions to achieve safe driving [29]. Weisong Shi et al. conduct extensive work in terms of MEC-based autonomous driving system design, they propose building an open vehicular data analytics platform (OpenVDAP) for CAVs and an optimal workload offloading and scheduling strategy in [30]. Furthermore, based on OpenVDAP, they present CAVBench, the first benchmark suite for the edge computing system in the CAVs scenario, which is composed of six typical applications covering four dominant CAV scenarios that implement four datasets as standard inputs [31]. Moreover, a modular scheme, which is an integrated approach to teaching autonomous driving, is proposed in [32].

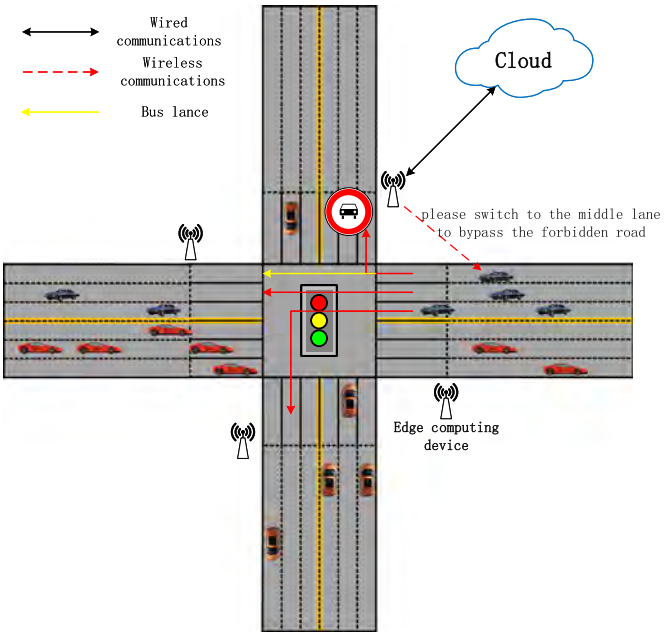


Fig. 1. The sketch of the crossroad.

Traffic at crossroads are the most complex; however, none of the existing works focus on this issue. Furthermore, the passing time of special vehicles is also a critical problem. Hence, the study of lane scheduling methods is urgently needed to ensure that different types of vehicles can pass through crossroads safely and quickly.

3. System configuration

In this section, we propose a novel autonomous driving system named edge computing-based lanes scheduling system (ECLSS) to ensure that vehicles can pass through crossroads safely and quickly. There are three layers in ECLSS.

- The bottom layer is composed of the vehicles. They can wirelessly interact with edge computing devices (ECD) which are deployed on the roads.
- The middle layer is composed of ECDs. They can perform information interaction between the vehicles (via wireless) and cloud (via wired). Furthermore, the requests of vehicles that have a real-time requirement are also performed in the ECD.
- The top layer is the cloud, which can provide long distance path planning and massive traffic information to the vehicles through ECD.

As depicted in Fig. 1, there are four ECD and many vehicles around the traffic post. The ECD have wired connections between each other, and they can obtain the surrounding traffic information (such as the duration of green lights(tl), the lanes states (ls)), the states of vehicles (such as speed (v), driving direction (d), current lane (cl), distance to crossroads (dc), navigation target and the type of vehicles (t)) from the

surrounding ECD and vehicles via wireless communications. Thus, when any vehicle drives into the crossroads, the ECD will send a suggestion of the speed and select the best lane for it depending on its direction, location and the surrounding traffic information. For example, a car drives from the right area (the vehicle connected to the ECD through the red dashed line) and wants to turn right at the crossing. When the corresponding ECD obtains the vehicle's states, the ECD will send a packet to the onboard autonomous driving system to "please switch to the middle lane to bypass the forbidden road" since the target road is closed and the right-most lane is the bus lane. Moreover, the replanned route is also in the packet (since path planning is beyond the scope of this paper, we just forward the planning result from the cloud).

To achieve a real-time response for each running vehicle, the scheduler of ECLSS is deployed in the ECD to calculate the optimal vehicle lane and transmit it with the traffic information to each vehicle wirelessly. Hence, we consider a task set in the ECD as $F = \{f_0, f_1 \dots f_n\}$, where n is the number of vehicles in the communication range of ECD (R). The vehicle ID is in an ascending sort order of the distance to the crossroads; hence, vehicle 0 is the vehicle nearest the crossroads (or the first vehicle that drives into the ECD's communication range), and n is the farthest one (or the last vehicle that drives into the ECD's communication range). The number of lanes is denoted as L; then, n^j is the last vehicle on lane j, and $|n^j|$ is the number of vehicles on lane j. Each task f_i is a series of operations that contain four processes: sensing road conditions, data upload from the vehicle to the ECD, algorithm computing and sending the result to each vehicle. Then, each task $f_i \in F$ is characterized by $\{tl_i, ds_i, v_i, d_i, cl_i, dc_i\}$, and the descriptions of these parameters are as follows:

- tl_i is the duration of a green light, where $tl_i = 0$ when the light is yellow or red. Furthermore, the vehicles can only go forward or turn left when the light is green;
- $ds_i = \{on, off\}$ is the direction state, in which the vehicle cannot go to its target direction when $ds_i = off$;
- v_i is the speed of i. In addition, the max speed for each vehicle in ECLSS is V;
- $d_i\{driving, right, left\}$ is the driving direction that vehicle i wants to pursue, which contains choices for keep driving, turn right and turn left;
- cl_i is the lane number that i drives in, and the vehicle can only turn left when the lane of l_i is a left-turn lane; otherwise, it must switch to the middle lane if it wants to drive forward. Furthermore, the total number of lanes is denoted as L;
- dc_i is the distance of vehicle i to the crossroads; and
- $t_i = \{ordinary, special\}$ is the type of vehicles. When i is a special vehicle, such as an ambulance, the other vehicles need to make way for it. Moreover, special vehicles can drive in any lane when the road is not closed.

Then, the driving vehicle can be described as the following: during t_i, car i drives in lane cl_i with a speed v_i with the target direction d_i. Then, the ECD will calculate the solutions for i, and update d_i^k, cl_i^k, tl_i^k, v_i^k and dc_i^k based on its collection information. For instance, when the ECD obtains $tl_i > \frac{dc_i}{v_i}$ and $ds_i = off$, the update can be explained according to the following instruction: please switch to the cl_i^k lane of direction d_i^k with the average speed v_i^k at the k^{th} switch; then, the remaining driving distance to pass the crossroads is dc_i^k after switching.

4. Lane scheduling scheduling problem

There are two potential constraints in autonomous driving: (1) the driving vehicles need to follow traffic rules; and (2) the primary task of autonomous driving is to ensure personal safety. Thus, challenges in lanes scheduling can be summarized as follows:

- A crash may occur when the vehicle switches its lane. How can lane switching be achieved safely?

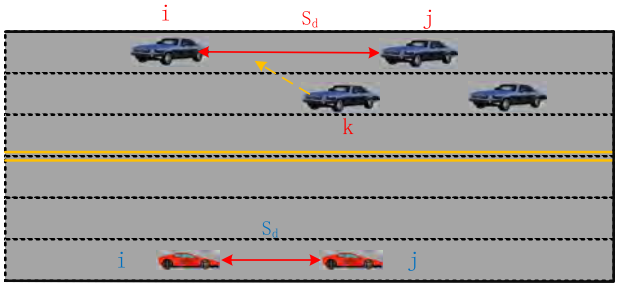


Fig. 2. The sketch of the safe distance.

- A vehicle can only turn at the crossroads when the traffic light of the corresponding lane is green. How can vehicles pass quickly and orderly?
- An ordinary vehicle must accommodate special vehicles, such as ambulances and fire trucks, so that they can pass through the crossroads quickly. How can lane switching for ordinary vehicles be scheduled accordingly?

To guarantee the safety of driving, we define the safe distance as follows:

Definition 1. Safe distance (Sd) is the minimum distance between two adjacent vehicles to ensure the safety of driving.

Fig. 2 is an example of the safe distance required when vehicle k wants to switch its lane to the left-turn lane; then, the distance between two adjacent vehicles i and j in the target lane must satisfy

$$\begin{aligned} &Sd \le dc_j - dc_k, \\ &Sd \le dc_k - dc_i, \\ &s.t. \\ &v_i \ge v_k \ge v_j. \end{aligned} \tag{1}$$

Similarly, the vehicles in the same lane are safe when the space of all vehicles and speed satisfies

$$\begin{aligned} &Sd \le dc_j - dc_i, \\ &s.t. \\ &v_i \ge v_j. \end{aligned} \tag{2}$$

We use the time overhead T to evaluate the performance of ECLSS, which is defined as the time at which the last vehicle arrives at its target direction that is determined the ECD schedule. Then, when the number of vehicles in the communication range R is $|n|$, the number of lane switches for i is K_i, the number of lane switches influencing i is I_i, the j^th influence on the time consumption is C_j, the travel distance at the k^{th} switch is Δ_i^k, and the time consumption for waiting at the signal light is $\aleph_i$. We can obtain the time overhead as

$$T = \max\left\{\sum_{k=0}^{K_i}\frac{\Delta_i^k}{v_i^k} + \sum_{j=0}^{j=I_i} C_j + \aleph_i, i \in n\right\}, \tag{3}$$

where Δ_i^k is

$$\Delta_i^k = dc_i^{k-1} - dc_i^k, k > 0. \tag{4}$$

Furthermore, each special vehicle i must pass the crossroads within a certain time s_i, that is,

$$\frac{\Delta_i^k}{v_i^k} + \sum_{j=0}^{j=I_i} C_j + \aleph_i \le s_i. \tag{5}$$

Hence, when the ECDs obtain traffic information F and the number of vehicles around the crossroads is n, our objective is scheduling the vehicles to arrive in the target directions with the minimum T under the constraints of Eqs. (1), (2) and (5).

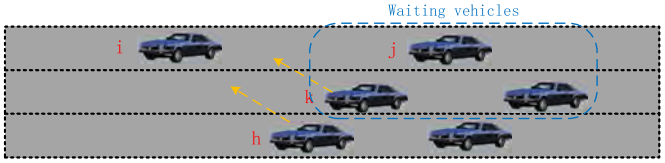


Fig. 3. An example of efficient lane switching.

For convenience of analysis, we ignore the extra distance caused by switching lanes since it is a small fixed value and it can be calculated by the Pythagorean Theorem easily.

5. Scheduling

To enhance the speed of vehicles that pass the crossroads and reduce T under the constraints of Eqs. (1), (2) and (5), we propose two lane scheduling algorithms (searching for efficient switching algorithm and special vehicles lane switching algorithm) in this section for the vehicles around the crossroads. The proposed methods can select the lane for each vehicle based on the global view of the ECD. The vehicle drives with the same speed as the one ahead of it in the range of $[0, V]$, and the distance between two vehicles in the same lane is set as Sd in ECLSS (this is easily implemented by adaptive cruise); then, our objective can be translated into reducing the time overhead of lane switching and make the vehicles pass crossroads quickly in ECLSS. That is, let n vehicles reach their target direction with a minimum T.

We divide each switch into three parts: (1) i and the vehicles behind i are stopped to the cars in front of i until the one that is ahead drives to a safe distance Sd (i will drive with speed V when there are no vehicles in front of it); (2) then, i begins to drive with the same speed as the cars ahead; and (3) when the distance between i and the vehicle behind i is Sd, the vehicles behind i begin to drive with the same speed as i. The time overhead of waiting for the k^{th} lane changing is denoted as β_i^k.

To reduce the time overhead, we must enhance the efficiency of lane switching and decrease the time to wait for the turn. As Fig. 3 shows, h can change its lane directly when k is waiting. Hence, we can find that the time overhead T is influenced by the number of lane switches K_i, the time overhead of waiting β_i^k, the distance to the crossroads dc_i and the time overhead of the traffic light $\aleph_i$. Since the time consumption for waiting at the traffic light is represented by the variables associated with $\sum_{k=0}^{K_i}\beta_i^k$, when red light takes α seconds and green light takes η seconds, we can calculate $\aleph_i$ as

$$\aleph_i = \left\lfloor \alpha \frac{\left(\sum_{k=0}^{K_i}\beta_i^k + \sum_{j=0}^{j=I_i} C_j + \frac{dc_i}{V}\right)}{\eta} \right\rfloor. \tag{6}$$

Then, we can rewrite Eq. (3) as

$$T = \max\left\{ \sum_{k=0}^{K_i}\beta_i^k + \frac{dc_i}{V} + \sum_{j=0}^{j=I_i} C_j + \left\lfloor \alpha\left(\sum_{k=0}^{K_i}\beta_i^k + \frac{dc_i}{V}\right)\right\rfloor, i \in n \right\}, \tag{7}$$

Hence, the key idea of our lane scheduling algorithms is to reduce T by a coordination approach that optimizes the relationship between K_i, I_i, C_j and β_i^k.

Lemma 1. *The influence of K_i on T is related to the distance of adjacent vehicles with i.*

Proof. We prove this lemma with a classification discussion based on the distance of adjacent vehicles. Based on Eqs. (1) and (2), we find that the switches do not increase the time overhead when the distance of adjacent vehicles is larger than Sd; that is, T remains unchanged. When i is the last vehicle in the current lane and the other lanes are faster than the current one, T is reduced; however, when the distance is less than Sd, the vehicle behind must decelerate to guarantee the safe driving distance; then, T is increased. Hence, the influence of K_i on T is related to the distance of adjacent vehicles with i. □

Lemma 1 reveals that the switches only influence i and the vehicles behind i. Thus, vehicle i should switch as much as possible in the situations in which no additional time overhead has been introduced when it drives in a lane that cannot provide access to its target direction ($cl_i \neq d_i \cap ds_i = on$) or the target direction is closed ($cl_i = d_i \cap ds_i = off$).

Theorem 1. *The additional time overhead of each switch T' is irrelevant to the position of the vehicle that switches when there are vehicles in front and behind it in the target lane.*

Proof. When k wants to switch its lane and there are vehicles in front and behind vehicle k, we denote the vehicle in front of k as i and behind k as j. As Fig. 2 shows, the additional time overhead in the right-turn lane is generated when

$$Sd \geq dc_k - dc_i, \tag{8}$$

then, ECD sends messages to adjust the speed of these three vehicles to ensure the safety of switching, and the waiting overhead β_k is

$$\beta_k = \frac{Sd - (dc_k - dc_i)}{V}. \tag{9}$$

the additional time overhead on the right-turn lane is

$$\begin{aligned} T' &= \frac{Sd - (dc_k - dc_i) + Sd - (dc_j - dc_k)}{V} + \aleph_j \\ &= \frac{2Sd + dc_i - dc_j}{V} + \aleph_j. \\ &= \frac{Sd}{V} + \aleph_j \end{aligned} \tag{10}$$

Since $\aleph_j$ is irrelevant to dc_k, we can obtain that the additional time overhead T' is irrelevant to the position of the vehicle that switches when there are vehicles in front and behind it in the target lane. □

Based on the traffic conditions of the target lane, we can obtain β_i^k as

$$\beta_i^k = \begin{cases} \frac{Sd-(dc_k-dc_i)}{V} & \exists j, dc_i - dc_j \leq Sd \\ 0 & \nexists j, dc_i - dc_j \leq Sd \end{cases} \tag{11}$$

Obviously, there is no time consumption when no vehicle j in the target lane satisfies $dc_i - dc_j \leq Sd$. Furthermore, we call the condition of $\beta = 0$ the optimal switch, and the situation of $0 \leq \beta < \frac{Sd}{V}$ the efficient switch. Based on Eq. (11), we denote the vehicle corresponding to the i^{th} blocking as b_i, and β_{b_i} is the delay caused by b_i to j; then, the blocking caused by the switching in front of vehicle j is

$$\sum_{i=0}^{i=I_j} C_j = \sum_{i=0}^{i=I_j} \beta_{b_i}. \tag{12}$$

Based on Lemma 1 and Eq. (11), we can obtain the following result.

Theorem 2. *For each vehicle i, the range of the time overhead to pass the crossroads is* $[\frac{dc_i}{V}(1 + \lfloor\frac{\alpha}{\eta}\rfloor), \frac{((i+L-1)Sd+dc_i)(1+\lfloor\frac{\alpha}{\eta}\rfloor)}{V}]$.

Proof. Eq. (11) illustrates that the vehicle i can change lanes without a slowdown when there is no vehicle j in the range of $dc_i - dc_j \leq Sd$. Hence, considering the blocking time consumption by the traffic light, we can obtain that the minimum time overhead of vehicle i to pass the crossroads is

$$\frac{dc_i}{V}\left(1 + \left\lfloor\frac{\alpha}{\eta}\right\rfloor\right). \tag{13}$$

In contrast, when there is a vehicle j in the k^th target lane of i that satisfies $dc_i = dc_j$, i needs to wait $\frac{Sd}{V}$. Thus, in the worst case, vehicle i

needs to switch $K_i = L - 1$ times to arrive in the final lane. That is, the maximum time to switch overhead of i is

$$\frac{Sd(L-1) + dc_i}{V}. \tag{14}$$

Since the switching only occurs in front of i, it can cause a delay, and the maximum waiting cost is $\frac{Sd}{V}$. Then, the total blocking time consumption is

$$i \times \frac{Sd}{V}. \tag{15}$$

In addition, since i needs to wait for the green light to pass the crossroads, the maximum time overhead to pass the crossroads is

$$\frac{((i+L-1)Sd + dc_i)\left(1 + \left\lfloor \frac{\alpha}{\eta} \right\rfloor\right)}{V} \tag{16}$$

Hence, for each vehicle i, the range of the time overhead to pass the crossroads is $[\frac{dc_i}{V}(1 + \lfloor \frac{\alpha}{\eta} \rfloor), \frac{((i+L-1)Sd+dc_i)(1+\lfloor \frac{\alpha}{\eta} \rfloor)}{V}]$. □

When i in Theorem 2 is the last vehicle of n, we find that the range of the time overhead T is $[\frac{dc_n}{V}(1 + \lfloor \frac{\alpha}{\eta} \rfloor), \frac{((n+L-1)Sd+dc_n)(1+\lfloor \frac{\alpha}{\eta} \rfloor)}{V}]$. However, $T = \frac{((n+L-1)Sd+dc_n)(1+\lfloor \frac{\alpha}{\eta} \rfloor)}{V}$ is too pessimistic. Since the number of lane switches for each vehicle i is fixed by cl_i and d_i, we can optimize T by reducing $\sum_{j=0}^{j=I_i} C_j$.

Based on the above results, we propose the searching for efficient switching algorithm (SESA) to reduce T with both breadth and depth searches for an efficient switch. The pseudocode of the SESA is in Algorithm 1, where cl_i^* is the lane after switching once. Note that

Algorithm 1 Searching for Efficient Switching Algorithm.

Require: F;
Ensure: all vehicles reach their target directions;
1: calculate K_i for each vehicle i;
2: **while** $(cl_j \neq d_j \cap ds_j = on) \cap (cl_j = d_j \cap ds_j = off) = \emptyset, j \in n$ **do**
3: **for** $i = 0$ to $|n|$ **do**
4: **if** $cl_i \neq d_i$ **then**
5: **if** $\exists p \in n$ satisfies $cl_p \neq d_p \cap dc_p \in [dc_i - Sd, dc_i + Sd] \cap ((cl_p, cl_p^*) \cap (cl_i, cl_i^*))$ **then**
6: switch p and i;
7: **else if** $dc_i = (K_i - k)Sd$ **then**
8: switch p and i;
9: **end if**
10: **for** the lane cl_q which have vehicle switches **do**
11: **if** $\exists p \in n$ satisfies $cl_p \neq d_p \cap dc_p > dc_q$ **then**
12: switch p and i;
13: **end if**
14: **end for**
15: update vehicles information and calculate T by Eqs. 7 and 11;
16: **end if**
17: **end for**
18: **end while**
19: **return** T;

Algorithm 1 is the case of executing the SESA once, and the SESA is repeatedly executed to schedule the new detected vehicles.

The SESA first calculates the lane switch order for each vehicle by both vehicle and traffic information (line 1) in the transmission range of the ECD; then, it judges whether each vehicle is driving in the lane to its target direction, and it will return T when the judgment is true (line 2). If there are vehicles that need to change lanes, the SESA will search for efficient switches to reduce T with both breadth and depth searches (lines 3–18). When the lane of vehicle i does not lead to its destination, it will search for the vehicles that want to switch in the range of $[dc_i - Sd, dc_i + Sd]$ in the related lanes; however, the current switch of i is not an efficient switch for the case in which no vehicles can switch in the same block of time (lines 3–6). Then, the switches that i makes can occur only when $dc_i = (K_i - k)Sd$ (lines 7–9). When the SESA determines vehicles that switch and their corresponding lanes (both the current lane and the lane after switching), it will reduce T by a depth search for an efficient switch (10–14). The vehicle information and T will be updated by Eqs. (7) and (11) (lines 15–18). Finally, the SESA returns T (line 19). The time complexity of the SESA is $O(n \log n)$.

Theorem 3. *The time overhead T can reduce at most $\lfloor \frac{y(L-1)}{L} \rfloor + \lceil \frac{\lfloor \frac{y(L-1)}{L} \rfloor}{\eta} \rceil \alpha$, where y is the number of efficient switches.*

Proof. The time consumption for each switch is in the range of $0 \leq \beta < \frac{Sd}{V}$. Hence, the maximum time overhead reduction occurs when $\beta = 0$. Since there are at most $(L-1)$ efficient switches that can occur simultaneously, the maximum time savings of efficient switches is

$$\left\lfloor \frac{y(L-1)}{L} \right\rfloor. \tag{17}$$

In addition, the time savings may cause the vehicles to pass the crossroads before another red light occurs; that is, the additional time savings of the traffic light is

$$\left\lceil \frac{\left\lfloor \frac{y(L-1)}{L} \right\rfloor}{\eta} \right\rceil \alpha. \tag{18}$$

Hence, the time overhead T can reduce at most $\lfloor \frac{y(L-1)}{L} \rfloor + \lceil \frac{\lfloor \frac{y(L-1)}{L} \rfloor}{\eta} \rceil \alpha$, where y is the number of efficient switches. □

Considering the characteristics of special vehicles, we then propose a special vehicles lane switching algorithm (SVLSA) to guarantee that special vehicles can pass the crossroads before their deadlines. The pseudocode of the SVLSA is in Algorithm 2.

Algorithm 2 Special Vehicles Lane Switching Algorithm.

Require: F;
Ensure: that the special vehicles that pass the crossroads satisfy Eq. 5;
1: calculate K_i for each vehicle i;
2: **while** $(cl_j \neq d_j \cap ds_j = on) \cap (cl_j = d_j \cap ds_j = off) = \emptyset, j \in n$ **do**
3: **for** $i = 0$ to $|n|$ **do**
4: **if** $t_i = special$ **then**
5: $s_i = s_i - \frac{dc_i}{V}$;
6: calculate $\aleph_i$;
7: **if** $\exists p \in n$ satisfies $cl_p \neq d_p \cap dc_p < dc_i \cap (cl_p = cl_i \cup cl_p^* = cl_i)$ **then**
8: calculate β_p by Eq. 11;
9: **if** $s_i - \beta_p - \aleph_i \geq 0$ **then**
10: update $\aleph_i$ and the remaining s_i;
11: switch p;
12: **end if**
13: **end if**
14: **else if** $\exists p$ satisfies $t_p = special \cap dc_p \geq dc_i \cap (cl_i = cl_p \cup cl_i *= cl_p)$ **then**
15: calculate $\aleph_i$ and β_i by Eqs. 6 and 11;
16: **if** $s_p - \beta_i - \aleph_i \geq 0$ **then**
17: update $\aleph_i$ and the remaining s_i;
18: switch p;
19: **end if**
20: **else**
21: execute as under SESA;
22: **end if**
23: **end for**
24: **end while**

Table 1
Simulation parameters.

Parameter	Description
n	The number of vehicles that need to be scheduled
L	The number of lanes
α	Red light duration
η	Green light duration
S	The number of special vehicles
T	Time overhead of n vehicles that pass the crossroads
K	Total number of switches
D	The number of vehicles that miss their directions

The SVLSA is mainly focused on the switches associated with special vehicles. For the case in which i is a special vehicle, the SVLSA first calculates the remaining time for passing the crossroads, and the other vehicles can only switch into cl_i or out of cl_i when the blocking time satisfies Eq. (5) (lines 4–13). For the ordinary vehicle that wants to switch into cl_i or out of cl_i, the SVLSA will decide whether to switch i or not by assessing the remaining time for passing the crossroads (lines 14–19). The remaining vehicles will execute as under the SESA. The time complexity of the SVLSA is $O(n\log n)$.

Since the vehicles cannot switch their lanes when the remaining time of a special vehicle is not sufficient (lines 8 and 15), the vehicles may miss their directions in the SVLSA to ensure that special vehicles can travel through the crossroads within their time constraint.

Theorem 4. *Vehicle p must miss its target direction when it satisfies* $(dc_p < dc_i) \cap (s_i - \beta_p < \frac{Sd_i}{V})$, *where i is the special vehicle in the same lane as p.*

Proof. When an ordinary vehicle p drives in front of a special vehicle i in the same lane$(dc_p < dc_i) \cap cl_i = cl_p$, the SVLSA will decide whether p can change its lane or not according to Eq. (5). That is, p can switch its lane only when the special vehicle can pass the crossroads within its time constraint s_i. When considering the time cost solely and the traffic light, vehicle p can switch its lane only when it satisfies $s_i - \beta_p \geq s_i - \beta_p - \aleph_i \geq \frac{Sd_i}{V}$. Hence, vehicle p must miss its target direction when it satisfies $(dc_p < dc_i) \cap (s_i - \beta_p < \frac{Sd_i}{V})$, where i is the special vehicle in the same lane as p. □

6. Experimental results

In this section, we compare our methods with the lane change in a real-life driving situation. We denote the switching method in an actual real-life scenario as lane switching without scheduling (LSWS). In LSWS, the vehicle changes its lane only based on its own information, such as its direction and the speed of the vehicle in front of it. Furthermore, the vehicle will change its lane immediately in LSWS when it wants to switch and does not consider the deceleration of other vehicles. That is, the vehicles schedules under LSWS adopt the local optimal method. All algorithms are implemented in C language. These programs run on a Windows machine with 3.4GHz CPU and 8GB memory.

To illustrate the applicability of our algorithms, for each parameter configuration, several test cases are generated randomly. For each test case, we set the safe distance as $Sd = 10m$, and the maximum speed is $V = 10$ m/s. In each case, there is one left-turn lane and one right-turn lane, and the number of special vehicles is in the range of [0,3], the red light duration is $\alpha = 20$ s and the green light duration is also $\beta = 20$ s. The traffic light changes to green at the beginning of each case. Moreover, there is at most one direction in the state of *off* in each case, and the vehicles that want to drive in this direction will change their target direction to keep driving along the road. Representative simulation parameters are summarized in Table 1.

Fig. 4 shows the relationship between the time overhead T and the number of vehicles n for $L = 5$ lanes and $S = 2$ special vehicles. The number of vehicles ranged from $n = 20$ to 60. We assume the vehicles are evenly distributed in each lane, and the maximum passing time of a special vehicle $s_i = 15s$. The time overheads of these three methods are increasing with the number of vehicles simultaneously. However, the rates of increase reveal an enormous difference. That is because at the beginning, all the vehicles can pass the crossroads in the first green light under the three methods; since the vehicles under the SVLSA may miss their directions and drive with V to guarantee Eq. (5), this phenomenon of a missed direction slows down the increasing rate of T. With the SESA, the vehicles search efficiently, and all the vehicles will reach their intended directions; hence, the increasing rates of the three methods satisfy $LSWS > SESA > SVLSA$. When the vehicles cannot drive to their destinations in the first green light, the remaining vehicles must wait until the next green light; thus, T increases sharply, such as at $n = 40$ when $\frac{K}{n} = 0.25$.

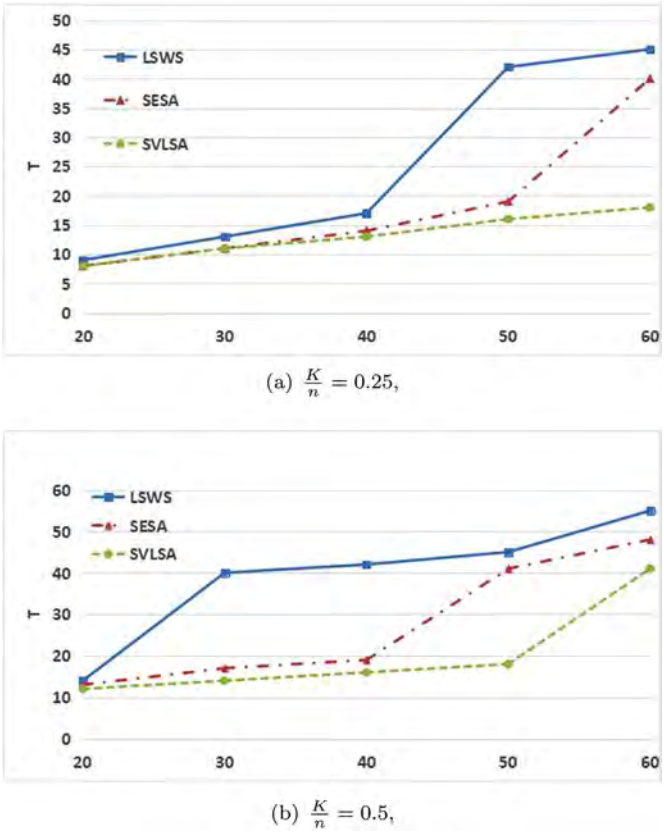


Fig. 4. The relationship between the time overhead T and the number of vehicles n.

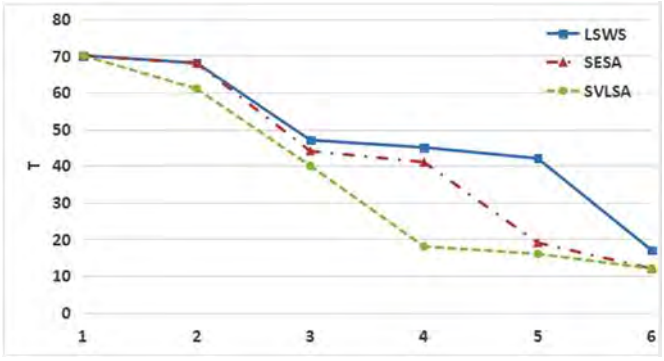


Fig. 5. The relationship between the time overhead T and the number of lanes L.

Fig. 5 shows the relationship between the time overhead T and the number of lanes L for $n = 50$ vehicles, $K = 13$ and $S = 1$ special vehicle.

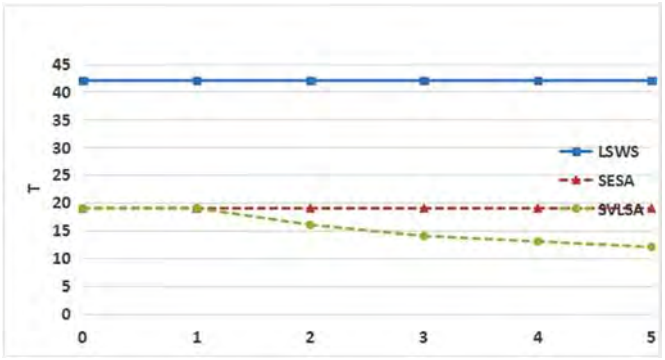


Fig. 6. The relationship between the time overhead T and the number of special vehicles S.

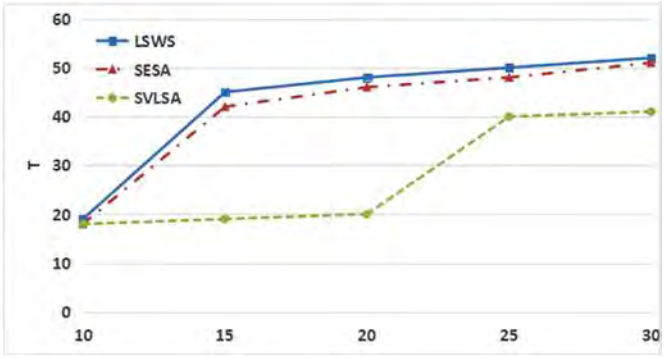


Fig. 7. The relationship between the time overhead T and the number of switches K.

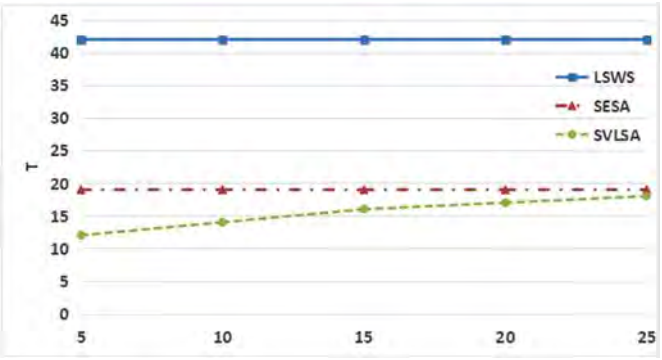


Fig. 8. The relationship between the time overhead T and the maximum passing time of special vehicle s_i.

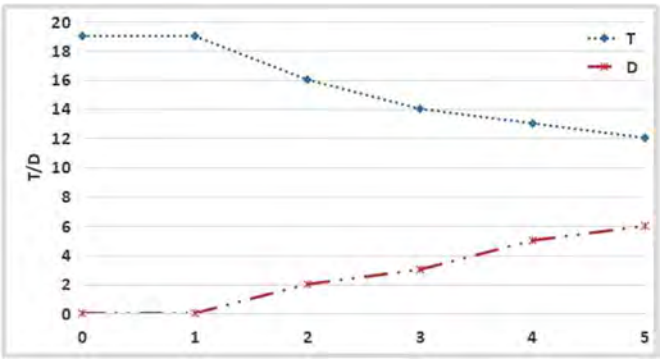


Fig. 9. The relationship between the number of vehicles that miss their directions D and the number of special vehicles S.

The number of lanes between $L = 1$ and $L = 6$ are considered. When there is only one lane, all the vehicles pass the crossroads through this lane; thus, the three methods have the same time overhead; however, as the lanes increase, the switching cost is optimized under the SESA (β) and the switching is restricted ($s_i - \beta - \aleph_i \geq \frac{Sd_i}{V}$) under the SVLSA. Hence, T reduces and satisfies $LSWS > SESA > SVLSA$.

The relationship between the time overhead T and the number of special vehicles S for $L = 5$ lanes, $K = 13$ and $n = 50$ vehicles is shown in Fig. 6. The number of special vehicles between $S = 0$ and $S = 5$ is considered. The time overheads are fixed under both *LSWS* and the *SESA* as S changes. For the *SVLSA*, it has the same time overhead as the *SESA* at $S = 0$ and $S = 1$. Considering the passing time of special vehicles, the *SVLSA* sacrifices the opportunity of some vehicles to reach their target directions. That is, the SVLSA reduces β and $\sum_{i=0}^{i=I_j} C_j$ as S increases. Therefore, the time consumption is inversely proportional to S.

Fig. 7 shows the relationship between the time overhead T and the number of switches K for $L = 5$ lanes, $n = 50$ vehicles and $S = 2$ special vehicles. The time overhead increases faster in Fig. 7 than that in Fig. 4, which illustrates that the switching consumes more time than straight driving. Similar to the above result, the *SVLSA* continues to have the lowest time overhead than the other two approaches.

Fig. 8 shows the relationship between the time overhead T and the maximum passing time of special vehicle s_i for $n = 50$ vehicles, $L = 5$ lanes and $S = 2$ special vehicles. The maximum passing time s_i is changed from 5 to 25 in units of 5. The time overhead is increased with s_i. The reason is that special vehicles have more time to pass the crossroads; consequently, the other vehicles can switch and generate additional time consumption. When this value reaches 25, the special vehicle consequences are not as imminent (all switches can occur and satisfy Eq. (5)) and degrade into the *SESA*.

In comparison from Fig. 8, we simulate the relationship between the number of vehicles that miss their directions D and the number of special vehicles S for $n = 50$ vehicles, $L = 5$ lanes and $K = 13$ in Fig. 9. Fig. 8 illustrates that the time overhead is cut with the SVLSA approach by sacrificing the number of vehicles that reach their target directions. Furthermore, the more the lanes occupied by special vehicles, the more the ordinary vehicles miss their target directions.

7. Conclusion

In this paper, a novel model named the edge computing-based lane scheduling system is proposed to study lane scheduling for vehicles (both ordinary and special vehicles) around the crossroads. Several edge computing devices (ECD) are deployed around the crossroads in ECLSS that can collect information from vehicles and road conditions. Three layers are considered in our model (vehicles, ECDs and the cloud), and we mainly focus on the second layer (ECD) since there are some related works that focus on the other two. Based on the centralized management of ECD, two lane scheduling methods for edge computing-based autonomous driving (the SESA and SVLSA) have been proposed to achieve high speed passing through the crossroads and guarantee that the special vehicles can pass the crossroads within a certain time, respectively. Extensive simulations are performed, and the simulation results demonstrate the superiority of the proposed approaches over competing schemes in a real-life lane switching scenario (LSWS).

The proposed ECLSS can be further extended by introducing deployed infrastructures, such as cameras, the traffic cloud and base stations. The lane scheduling methods for algorithms both with and without special vehicles can be extended to solve the scheduling issue when such infrastructures are deployed. The inherent issue is how to achieve on-demand computational and communicational resource allocation between these infrastructures to meet the additional requirements of different vehicles, which can offer higher computing power, but short link durations and bandwidth limitations. Moreover, the details of lane

switching in ECLSS are an efficient for further reducing the time overhead of vehicles passing crossroads, such as the discussion of the details of the blocking time consumption. Finally, weather is a critical factor that affects trafficÂ information; therefore, determining an approach to analyze trafficÂ information with both cloud and edge computing to dynamically adjust safe distances is also important. Solving the above problems is another potential direction for future research.

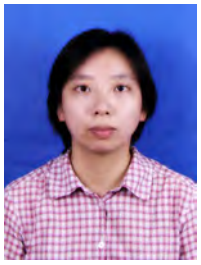
Acknowledgment

This work was partially supported by the Liaoning Provincial Natural Science Fund Project (20180520029,20180540114), the National Natural Science Foundation of China (61502474,61803368,61533015), the independent subject of State Key Laboratory of Robotics, and the Youth Innovation Promotion Association CAS.

References

[1] S. Kumar, S. Gollakota, D. Katabi, A cloud-assisted design for autonomous driving, in: Proceedings of the Edition of the Mcc Workshop on Mobile Cloud Computing, 2012, pp. 41–46.
[2] M. Gerla, E.K. Lee, G. Pau, U. Lee, Internet of vehicles: from intelligent grid to autonomous cars and vehicular clouds, in: Proceedings of the Internet of Things, 2016, pp. 241–246.
[3] B.G. Chun, S. Ihm, P. Maniatis, M. Naik, A. Patti, Clonecloud: elastic execution between mobile device and cloud, in: Proceedings of the Conference on Computer Systems, 2011, pp. 301–314.
[4] E. Cuervo, A. Balasubramanian, D.K. Cho, A. Wolman, S. Saroiu, R. Chandra, P. Bahl, Maui:making smartphones last longer with code offload, in: Proceedings of the International Conference on Mobile Systems, Applications, and Services, 2010, pp. 49–62.
[5] D. Huang, X. Zhang, M. Kang, J. Luo, Mobicloud: building secure cloud framework for mobile computing and communication, in: Proceedings of the IEEE International Symposium on Service Oriented System Engineering, 2010, pp. 27–34.
[6] M. Shojafar, N. Cordeschi, E. Baccarelli, Energy-efficient adaptive resource management for real-time vehicular cloud services, IEEE Trans. Cloud Comput. PP (99) (2016) 1.
[7] M. Whaiduzzaman, M. Sookhak, A. Gani, R. Buyya, A survey on vehicular cloud computing, J. Netw. Comput. Appl. 40 (1) (2014) 325–344.
[8] T. Mekki, I. Jabri, A. Rachedi, M.B. Jemaa, Vehicular cloud networks: challenges, architectures, and future directions, Veh. Commun. 9 (2016) 268–280.
[9] W. Yu, F. Liang, X. He, W.G. Hatcher, C. Lu, J. Lin, X. Yang, A survey on the edge computing for the internet of things, IEEE Access 6 (99) (2018) 6900–6919.
[10] P. Mach, Z. Becvar, Mobile edge computing: a survey on architecture and computation offloading, IEEE Commun. Surv. Tutor. PP (99) (2017) 1.
[11] Boeing 787s to create half a terabyte of data per flight, says virgin atlantic. accessed on dec. 7, 2016. [online]. available: https://datafloq.com/read/self-driving-carscreate-2-petabytes-data-annually/172.
[12] p. Nelson just one autonomous car will use 4,000 gb of data/day[j]. network world. idg enterprise, framingham, 2016.
[13] W. Shi, J. Cao, Q. Zhang, Y. Li, L. Xu, Edge computing: vision and challenges, IEEE Internet Things J. 3 (5) (2016) 637–646.
[14] X. Hou, Y. Li, M. Chen, D. Wu, D. Jin, S. Chen, Vehicular fog computing: a viewpoint of vehicles as the infrastructures, IEEE Trans. Veh. Technol. 65 (6) (2016) 3860–3873.
[15] J. Feng, Z. Liu, C. Wu, Y. Ji, Ave: autonomous vehicular edge computing framework with aco-based scheduling, IEEE Trans. Veh. Technol. PP (99) (2017) 1–1.
[16] Y. Sun, X. Guo, S. Zhou, Z. Jiang, X. Liu, Z. Niu, Learning-based task offloading for vehicular cloud computing systems (2018) 1–7.
[17] Y. Feng, K.L. Head, S. Khoshmagham, M. Zamanipour, A real-time adaptive signal control in a connected vehicle environment, Transp. Res. Part C 55 (2015) 460–473.
[18] M. Wang, H. Shan, R. Lu, R. Zhang, Real-time path planning based on hybrid–vanet-enhanced transportation system, IEEE Trans. Veh. Technol. 64 (5) (2015) 1664–1678.
[19] N. Lu, N. Cheng, N. Zhang, X. Shen, Connected vehicles: solutions and challenges, Internet Things J. IEEE 1 (4) (2014) 289–299.
[20] Apple carplay - the ultimate copilot. available: https://www.apple.com/ios/carplay/.
[21] Dueros for smart automobile unit solution. available: https://dueros.baidu.com/en/html/2017/jjfasj0608/10.html.
[22] Ford will have a fully autonomous vehicle in operation by 2021,available: https://corporate.ford.com/innovation/autonomous-2021.html.
[23] Y. Zhou, o. Tuzel, Voxelnet: end-to-end learning for point cloud based 3D 535 object detection[c], in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4490–4499.
[24] b. Schoettle, m. sivak, A survey of public opinion about autonomous and self driving vehicles in the US, the UK, and Australia, University of Michigan, 2014 Technical report no. umtri-2014-21. Retrieved 540 from: http://deepblue.lib.umich.edu/bitstream/handle/2027.42/108384/103024.pdf.
[25] S. Shalevshwartz, S. Shammah, A. Shashua, Safe, multi-agent, reinforcement learning for autonomous driving (2016) 1–13.
[26] B. Paden, M. p, S.Z. Yong, D. Yershov, E. Frazzoli, A survey of motion planning and control techniques for self-driving urban vehicles, IEEE Trans. Intell. Veh. 1 (1) (2016) 33–55.
[27] Y.L. Hu, C.Y. Wang, C.K. Kao, S.Y. Chang, D.S.L. Wei, Y. Huang, I.Y. Chen, S.Y. Kuo, Toward fog-based event-driven services for internet of vehicles: design and evaluation, International Conference on Internet of Vehicles. Springer, Cham. (2017) 201–212.
[28] M.Q.L. Liu, X. Zhang, W. Shi, Safeshareride: edge-based attack detection in ridesharing services, in: Proceedings of the third ACM/IEEE Symposium on Edge Computing (SEC), Bellevue, WA, 2018, pp. 25–27.
[29] K. Sasaki, N. Suzuki, S. Makido, A. Nakao, Vehicle control system coordinated between cloud and mobile edge computing, in: Proceedings of the Society of Instrument and Control Engineers of Japan, 2016, pp. 1122–1127.
[30] Q. Zhang, Y. Wang, X. Zhang, L. Liu, X. Wu, W. Shi, H. Zhong, Openvdap: an open vehicular data analytics platform for cavs, in: Proceedings of the IEEE International Conference on Distributed Computing Systems, 2018, pp. 1310–1320.
[31] X.W. Yifan Wang, Liu Shaoshan, W. Shi, Cavbench: a benchmark suite for connected and autonomous vehicles, in: Proceedings of the Third ACM/IEEE Symposium on Edge Computing (SEC), Oct. 25–27, 2018, Bellevue, WA., 2018.
[32] J. Tang, S. Liu, S. Pei, S. Zuckerman, L. Chen, W. Shi, J.L. Gaudiot, Teaching autonomous driving using a modular and integrated approach, in: Proceedings of the IEEE Computer Software and Applications Conference, 2018, pp. 361–366.

Changqing Xia received the Ph. D. degree from Northeastern University, China in 2015. He is currently an assistant professor at Shenyang Institute of Automation, Chinese Academy of Sciences. His research interests include wireless sensor networks and real-time systems, especially the real-time scheduling algorithms, and smart energy systems.



Xi Jin received the Ph. D. degree from Northeastern University, China in 2013. She is currently an associate professor at Shenyang Institute of Automation, Chinese Academy of Sciences. Her research interests include wireless sensor networks and real-time systems, especially the real-time scheduling algorithms, and worst case end-to-end delay analysis.

Linghe Kong is currently an associate professor with Department of Computer Science and Engineering at Shanghai Jiao Tong University. Before that, he was a postdoctoral researcher at Columbia University, McGill University, and Singapore University of Technology and Design. He received his Ph.D. degree from Shanghai Jiao Tong University, 2012, his Master degree from TELECOM SudParis 2007, and his B. E. degree from Xidian University, 2005. His research interests include wireless communication, sensor networks, mobile computing, Internet of things, and smart energy systems.

Chi Xu received the Ph.D. degree from University of Chinese Academy of Sciences, China, in 2017. He has been with Shenyang Institute of Automation, Chinese Academy of Sciences, China, since 2013, and currently is serving as an assistant professor. He is a member of IEEE, China Computer Federation (CCF), and Chinese Association of Automation (CAA). He also serves as a 3GPP Delegate for RAN1. His research interests include cognitive radio networks, wireless sensor networks, industrial internet, and 5G.

Peng Zeng received his Ph. D. degree from Shenyang Institute of Automation, Chinese Academy of Sciences. He is currently a professor at Shenyang Institute of Automation, Chinese Academy of Sciences. His research interests include industrial communication and wireless sensor networks.