Providing Reliable Communications over Static-node-assisted Vehicular Networks Using Distance-vector Routing

Takuya Yoshihiro¹ · Daichi Araki² · Hiroki Sakaguchi³ · Naoki Shibata³

Published online: 12 February 2018
© The Author(s) 2018. This article is an open access publication

Abstract

To support various practical applications that are expected to work over vehicular networks, it is important to develop a network infrastructure that provides quality communications on which applications can rely. However, since a vehicular network is essentially a delay tolerant network (DTN) in which connected paths to the destinations do not always exist, providing reliable communications on such a network is a challenging research goal. In this paper, we propose a new table-driven distance-vector routing protocol called RDV (Reliable Distance-Vector routing) for static-node-assisted vehicular networks. RDV provides the guarantee that the expected packet delivery ratio exceeds the preconfigured value by creating the necessary number of packet copies sent on the shortest path. Evaluation results show that RDV provides a stably high packet delivery ratio and outperforms other conventional routing schemes in both packet delivery ratio and overhead of packet copies.

Keywords Vehicular networks · Static-node assistance · Distance-vector routing · Delay tolerant communications

1 Introduction

Given that large numbers of vehicles are in movement around the world, such vehicles are a potential information infrastructure useful in the IoT era. Vehicles can act as sensors to observe the world, while simultaneously functioning as a networking platform to support communications among vehicles or their drivers as well. To put this useful infrastructure of vehicular networks to use, various practical applications have been considered for the improvement of social efficiency and the quality of people's experiences, with

applications in road-safety, cooperative congestion reduction, road management, navigation assistance, information sharing, infotainment, etc. [1].

However, unfortunately, many of these applications will not be deployed in practice, for several reasons. First, we currently have no appropriate packet routing and forwarding mechanisms that meet the requirements of various applications considered for vehicular networks. Many application proposals offer their own network protocols or architectures that do not support other applications. To promote various applications for the real world, a common platform would be required, over which various kind of applications work together. Second, the current vehicular network architectures proposed so far do not provide the quality communications that many applications rely on. Since vehicular networks by nature are opportunistic networks in which successful packet delivery depends on the amount and timing of vehicles moving on roads, providing reliable communications over vehicular networks that support various applications is a challenging research goal.

Of course, vehicular networks cannot provide the same quality communication as current network infrastructure such as the Internet. This means that we have to consider what quality vehicular networks can provide. However, vehicular networks fall into the category of

✉ Takuya Yoshihiro
tac@sys.wakayama-u.ac.jp

Daichi Araki
araki.daichi@g.wakayama-u.jp

Naoki Shibata
n-sibata@is.naist.jp

¹ Faculty of Systems Engineering, Wakayama University, 930 Sakaedani, Wakayama, 640-8510, Japan

² Graduate School of Systems Engineering, Wakayama University, 930 Sakaedani, Wakayama, 640-8510, Japan

³ Graduate School of Information Science, Nara Institute of Science and Technology, 8916-5 Takayama-cho, Ikoma, Nara, Japan

DTNs (Delay-Tolerant Networks) in which connected paths from senders to receivers do not always exist. We can expect the connected multi-hop communications only in limited situations such as urban areas even in daytime, where a sufficient number of vehicles are available to relay packets to their destinations. Since communication paths are easily disconnected,¹ our goal in vehicular networks becomes providing quality communications with a predefined expectation on delivery ratio as DTNs, while connected communications are possible only when the connected paths to destinations are available. Note that, even over this type of DTNs, we still have applications to provide delay-tolerant services such as advertisements of regional shops or restaurants, sharing local traffic jam information, data collection into sink servers located at static nodes or on the Internet, etc.

In the literature, we have a sequence of studies that treats vehicular networks as a kind of DTNs. As a representative example, several geographic routing proposals such as VADD [2] and MDDV [3] improved packet delivery ratio by incorporating the *carry-and-forward* behavior to enable opportunistic forwarding where vehicles usually hold packets and pass them only when they meet other vehicles. Since vehicles move with high speed and may meet others occasionally, this strategy would be suitable for vehicular networks. However, due to the nature of DTNs, packets are frequently lost and fail to reach their destinations, especially in case of sparse-vehicle scenarios, which is a serious problem, since an infrastructure should provide reliable services.

A significant improvement on packet reachability was made by introducing an idea to deploy static nodes at intersections. Ding et al. proposed a routing scheme called SADV (a Static-node-assisted Adaptive Data dissemination in Vehicular networks) [4] that improves the reliability of communications by deploying static nodes (i.e., stationary wireless nodes without wired connection) at intersections to support routing packets. In SADV, static nodes are placed at each intersection and they behave like link-state routers to assist packet forwarding, i.e., they measure the delivery delay between neighbor nodes, share them with other nodes, compute the shortest-delay paths, and help packets travel along the paths. By waiting for vehicles that carry packets in the right direction of their paths, each static node controls packets to enable them to reach their destinations with high probability. Furthermore, by sending two duplicated copies of a packet at each node to the shortest and the second shortest next-hops, SADV further improves the packet reachability.

¹Imagine that traffic signals easily generate a group of vehicles and vehicles in different groups could hardly communicate with each other. Traffic signals thus tend to divide a vehicular network into multiple network components.

Although SADV significantly improves the ratio of successful packet deliveries, there are several flaws for a reliable infrastructure that supports various services in practice. First, SADV assumes that every node has the up-to-date roadmap to search for locations of static-nodes and their neighbors. Since static nodes are not connected to the Internet, updating roadmaps at every node requires considerable maintenance cost. Second, SADV is assumed to locate static nodes densely at most intersections in the target region. Because SADV sends a single packet to a neighbor static node, the packet is expected to reach the neighbor for sure; otherwise, reliable delivery to the destination is not possible. However, a vehicle that carries a packet might stop or change its direction halfway. High reachability to neighbor nodes is achievable only if static nodes are placed densely in most of intersections. Third, SADV is not sufficiently reliable as an information infrastructure since it does not provide any mechanism to manage delivery ratio. In fact, the delivery ratio of SADV depends on the characteristics of traffic patterns or network topologies since SADV adopts the normal best-effort forwarding, although it is reinforced by a multi-path strategy. As a dependable infrastructure, it is expected to provide some kind of guarantee on packet delivery.

In this paper, we propose a new design of static-node-assisted vehicular networks called RDV (Reliable Distance-Vector Routing) that solves the above practical problems by introducing several new concepts in vehicular networks.

First, RDV deploys simple and light *distance-vector routing* instead of link-state routing to compute reliable paths between every pair of static nodes. Thanks to the favorable characteristics of distance-vector routing, neighbor discovery and shortest-path computation are both executed autonomously within the low overhead of control messages. Since all the participating static nodes are detected by routing protocols, RDV works without the help of roadmaps. Second, RDV incorporates *statistic-based vehicle selection* (SVS) that captures the statistic on local traffic patterns of vehicles to select with high probability vehicles that carry packets to the next static nodes. As a base of reliable routing, SVS computes the probabilities of each vehicle to reach other static nodes. Third, RDV introduces *adaptive packet duplication* (APD) that adaptively creates the required number of packet copies based on the probabilities computed by SVS to achieve a certain ratio of successful packet delivery. This function provides a better expectation value than a predefined value on packet delivery ratio even over unreliable links, which allows us to locate statistic nodes in only some of the major intersections to reduce deployment cost.

The reminder of the paper is organized as follows: In Section 2, we describe the related work on the routing techniques for vehicular networks. In Section 3, we present

the requirement and the design of RDV. After that we describe the specific mechanisms of RDV in Section 4. We evaluate the performance of RDV through simulation in Section 5. Then, we discuss several practical issues on RDV in Section 6. Finally, in Section 7, we conclude our work in this paper.

2 Related work

2.1 Geographic routing with DTN support

In urban areas, we can assume relatively high density of vehicles so that connected multi-hop paths to the destinations are available in many cases. In this situation, the main challenge is to cope with the dynamics due to the high mobility of vehicular networks. Since traditional MANET routing protocols are not suitable to handle the high mobility situations, geographic routing has been typically applied since the early stages of the studies on multi-hop vehicular networks.

GPCR [5] and GPSRJ+ [6] are both extensions of general geographic routing protocol GPSR [7] to fit greedy geographic routing into vehicular networks. Since vehicles move along roads, obstacles such as buildings often prevent packets from being forwarded in the right direction at intersections in urban areas. To solve the problem, GPSR [5] forwards packets once to the center of the intersections and then forwards each of them in its own directions. In GPSRJ+ [6], to cope with the same problem, beacons are forwarded within a two-hop range so that nodes watch two-hop neighbors to make better next-hop selection so as to forward packets to their own directions at intersections. GSR [8] is based on source routing in which nodes compute the shortest paths to the destinations using a route map and forward packets so as to follow the computed sequence of junctions. A-STAR [9] incorporates traffic information obtained from a dynamically or statically rated map in the anchor-based routing, which follows the sequence of junctions (i.e., anchors) so as to avoid low-traffic roads. One of their drawbacks is the poor performance in sparse-vehicle scenarios since they silently drop packets when no valid next-hop to their destinations is available.

On the other hand, many studies consider the sparse scenarios where connected paths to the destinations are rarely exist. In such cases, networks are regarded as DTNs. One of the most classical routing schemes in DTNs is the so-called epidemic routing [10], in which nodes simply pass a copy of each packet when they meet other nodes in order to make the packets more likely to reach their destinations. Yoo et al. analyzed and evaluated the performance of epidemic routing in vehicular networks [11]. However, since packets are simply copied and become flooded in the

network, the overhead of epidemic routing is extremely high in general.

To incorporate the DTN feature into geographic routing schemes in vehicular networks, the *carry-and-forward* behavior, i.e., behavior of vehicles carrying packets by themselves until favorable next-hop vehicles appear, is usually applied. VADD [2] introduced the *carry-and-forward* behavior, designed to achieve minimum-delay packet delivery even in sparse-vehicle scenarios. By introducing *carry-and-forward* behavior, vehicular networks are able to work as DTNs, while real-time communications are possible simultaneously if connected paths to the destinations are available.

Even if *carry-and-forward* behavior is introduced, forwarding packets along 'connected' paths, in which vehicles do not need to 'carry' packets, is the key strategy to improve communication performance in both delivery ratio and delay. Therefore, several techniques have been combined with geographic routing in order to choose the connected paths with high probability [3, 12, 13]. MDDV [3] forwards packets along a precomputed trajectory that was computed from static road information. GyTAR [12] considers dynamic traffic information such as the density of vehicles on each road to select the connecting roads as the forwarding paths at each intersection. SRPMT [13] introduces a concept of micro topology (MT), which is the topology among vehicles on each street, to estimate more precisely the connectivity of the street.

For all those dedicated studies, however, it is still difficult to provide guaranteed delivery within pure vehicular networks because the networks are essentially not always connected, and also because the mobility is too large to follow.

2.2 RSU-based network infrastructure

For improved reliability, a variation of packet dissemination techniques that makes use of static nodes called RSUs (Road-Side Units) has been proposed in the literature. The most typical deployment of RSUs is providing single-hop communications between RSUs and vehicles, which has already been standardized and deployed in practice [1]. As the most popular technology, we have DSRC (Dedicated Short Range Communications), which supports both V2V and V2I communications where a vehicle and a wire-connected Road-Side Unit (RSU) directly communicate with each other to provide various kind of services such as electronic payment, navigation assistance, etc. However, it takes a significantly high cost to cover a large geometric region with this kind of single-hop RSUs via direct communications, making it unrealistic in practice.

To enhance the coverage area of each wired RSU, several multi-hop data delivery techniques have been proposed.

D-Greedy [14] considers time-bounded delivery of packets from vehicles to RSUs that maximize the number of packets delivered within the specified time limitation by utilizing the traffic density and the length of each street. PROMPT [15] provides a beacon-based connectivity between vehicles and RSUs where beacons of RSUs are propagated within several hops to enable vehicles to choose delay-optimal paths to connect to one of the available RSUs. TSF [16] proposes to deliver packets from RSUs to a target vehicular with minimized delivery delay by delivering packets to a rendezvous point that is computed based on the prediction of vehicular movement.

Unfortunately, building many wired RSUs to cover a certain region still entails a high cost that is not negligible in practice. On the other hand, if we decrease the number of wired RSUs, communications will lose quality and reliability because a large number of hops is required to connect vehicles and RSUs, whereas such communications are supported by certain multi-hop routing schemes such as the geographic routing described in Section 2.1.

2.3 Static-node-assisted vehicular networks

Because non-wired static nodes can be placed far less expensively, static-node assistance to enhance packet delivery is a realistic means to build a platform for vehicular networks. However, we have only a few proposals on the static-node assistance in the literature.

SADV [4], which we described previously, is the representative work to assist vehicular networks with static nodes placed at intersections. SADV showed that reliable packet delivery is possible with the assistance of static nodes, by forwarding two packet copies to two next-hops corresponding to the first and the second shortest paths at each node. ETGR [17] also utilizes static nodes within a reactive routing scheme, in which RREQ is sent to the destination with assistance of static nodes. Static nodes actually improve the successful packet delivery ratio that supports delay-tolerant applications. However, several drawbacks described in Section 1 exist. Providing some guarantee on quality communications is, in practice, one of the most important issues.

Our proposed RDV is the first method to provide quality communications with a guarantee on the expected packet delivery ratio over vehicular networks assisted by unwired static nodes.

2.4 RSU technologies in practice

Several wired RSU technologies have already been deployed mainly for toll collection and advanced navigation systems. Since RSU has special requirements such as communications with high-speed vehicles, a set of

communication standards for DSRC (Dedicated Short-Range Communications) based on IEEE 802.11p [24] and IEEE 1609 [25] families has been dedicated for this purpose. Although the specifications are independently standardized in the U.S., Europe, and Japan, respectively, practical services with RSUs have been provided all over the world as, for example, ERP (Electric Road Pricing) in Singapore [26], and VICS (Vehicle Information and Communication Systems) in Japan [27].

However, one of the problems with the wired RSUs is the significant cost not only to build the hardware equipment but also for the wired/wireless connections. The wiring cost especially is very expensive because cables need to be laid along major streets in the whole area to provide services, while wireless connection also imposes non-negligible cost of using precious commercial bands for packet forwarding. Our approach for this problem is to off-load the traffic of local applications to local vehicular networks. RDV supports local packet forwarding with static nodes, which are possible to deploy inexpensively by utilizing the power supply for traffic lights as well as well-populated communication devices such as Wi-Fi.

3 RDV design

3.1 Requirements

We propose a new practical static-node-assisted routing scheme RDV for vehicular networks, a scheme which works as a reliable infrastructure that supports various applications. To enhance practicality and reliability, we first consider the requirements arising from practical scenes, and then design RDV to satisfy them by introducing certain new concepts and techniques. The requirements considered in this study are as follows:

- (a) **Reliable Packet Delivery:** As a network infrastructure to support various kind of services, especially used in commercial purposes, maintaining the state with a high packet delivery ratio has a significant importance. As we see on the Internet, high a packet delivery ratio even under best-effort forwarding can provide services that people rely on. In contrast, all V2V routing schemes in the literature work opportunistically and thus do not provide a sufficient packet delivery ratio. So as allow all applications on the networks, some kind of guaranteed packet delivery is beneficial.
- (b) **Low Communication Overhead:** The overhead of control messages in routing protocols is not expected to be large since communication capacity of vehicular networks is generally not large. When roads are filled with

vehicles, communication capacity for each vehicle is significantly limited. Unless control overhead is sufficiently low, the network would not work in such crowded states.

(c) Avoiding Dependency on Roadmaps: The topology of roads might change frequently in the real world due to development of new roads, temporary maintenance, etc., although the frequency depends on the locale. Since we do not suppose that a static node has a wired connection, nor that every node is required to update its map data every time the topology is changed, the operation to update roadmaps may take a few days. This could bring non-negligible operational costs, and should be avoided. Furthermore, topology changes may cause instability of networks due to an inconsistency of forwarding paths. For example, packet loops can be caused by unsynchronized routing tables independently computed by each node. To reduce the unfavorable effects coming from roadmaps, routing schemes are advantageous for being independent of roadmaps, and routes can be adapted to changes of topology and traffic patterns with time.

(d) Sparse deployment of static nodes: The cost of placing static nodes could be a real limitation in the actual deployment of static-node-assisted routing schemes. Given the many narrow roads in many cities, placing static nodes at all intersections is not realistic. If we adopt sparse deployment of static nodes and place nodes only at major intersections, we have to consider that a certain ratio of vehicles would get parked in a parking lot alongside the roads or enter into narrow roads. This would cause a loss of data on the 'link' between static nodes. Consequently, our scheme must work to provide reliable communications with a high delivery ratio even over such unreliable links.

(e) Low Delivery delay: Delivery delay in general is an important measure of communication quality, which should also meet the conditions for vehicular networks. We expect to provide low-delay multi-hop communications in metropolitan areas which will also behave as a DTN in the case that no connected path is available. In both cases, we aim for a reasonably low-delay delivery.

We claim in this paper that the above requirements are largely satisfied by introducing distance-vector routing in combination with related techniques.

3.2 Design strategy

In RDV, static nodes exchange information in the manner of distance-vector routing protocols to compute their routing table for packet forwarding. RDV computes reliable paths among static nodes and forwards packets along them. By the nature of the distance-vector protocols, RDV achieves low control overhead to compute the shortest paths, in contrast

to the link-state routing which requires sharing full topology information among all nodes in the network. Our design of RDV can be regarded as the choice of taking the low messaging overhead of distance-vector routing rather than high messaging overhead of link-state routing, and rather than the high maintenance cost coming from maps.

The distance-vector strategy also benefits us by enabling each node to consider nodes placed several-hops away as the candidates of next-hops, whereas link-state routing only cares about neighbors within a '1-hop' distance as links in its shared network topology; otherwise, the number of links in the topology database would easily explode exponentially. As a result, in contrast to the existing link-state routing schemes, in RDV, a vehicle directly passes a packet to the next relay node without any support from other static nodes on the way; that is to say, when a vehicle visits a static node, it does not consult the route for the packets whose next relay is not the visiting node. This mechanism significantly reduces the communication overhead. Also, thanks to the function that identifies neighbors in the routing protocol, RDV does not require roadmaps to know the node locations and their neighbor relationships. In other words, RDV adaptively changes the paths according to the topology and the traffic conditions that change with time.

Let us consider here that the distance-vector routing in general is said to have the drawback of slow convergence in face of topology changes. However, since the traffic pattern generally changes in the cycle of one day, and which is sufficiently slow, RDV is able to follow it in practice. Consequently, in RDV, we look only at convergence speed, temporary events such as traffic accidents and road management.

For reliable packet delivery, RDV introduces two related techniques, i.e., *statistic-based vehicle selection* (SVS) and *adaptive packet duplication* (APD). For reliable packet delivery, static nodes first have to know the moving direction of each vehicle entering the communication range, in order to choose the vehicles to which to pass packets. However, this is actually difficult since vehicles may change their direction in intersections and in many cases the communication occurs before the direction change. To solve the problem, SVS investigates the statistics between in-coming and out-going directions (i.e., the previous and the next visiting nodes) at the intersection, and, to pass packets, chooses the vehicle that will go in the expected direction with high probability. Besides the local probabilities obtained by SVS along the path a packet travels, APD provides the function to make the required number of packet copies that will guarantee meeting the preconfigured expectations on packet delivery ratio reaching the destination. By combining SVS with APD, RDV guarantees meeting the expectations on end-to-end packet delivery ratio with the least overhead of packet copies.

Note that the mechanism for the reliable packet delivery also contributes to the sparse deployment of static nodes, because it achieves a high delivery ratio over unreliable links. RDV provides reliable communications among static nodes placed sparsely at the main large intersections by duplicating packets to secure a sufficient number of packets in order to achieve the preconfigured packet delivery ratio.

Finally, for low delivery delay as well as reliable communications, we introduce a routing metric based on the number of vehicles passing on the roads. The amount of traffic is of primary importance in transmitting a number of packet copies within a small delay. By computing the shortest paths in terms of the sum of the metrics, RDV computes the reliable paths that support stable communications.

3.3 Assumptions and definitions

We assume that the static nodes are not placed at every intersection, but only at major ones. Packets destined to arbitrary coordination are generated by vehicles or static nodes; however, for simplicity, we consider here the problem of delivering packets to the closest static nodes. To carry packets from the source vehicles to a static node, we can use any of several methods found in the literature, such as [14, 15]. Static nodes and vehicles are equipped with their own wireless communication cards that can be connected with each other. We are not concerned with the PHY and MAC protocols deployed on them, since we just assume that a vehicular node and a static node can carry out a small amount of communications when they meet.

As with several conventional routing schemes, vehicles in RDV carry out the carry and forward behavior. Specifically, a vehicle outside the communication range of any static node behaves in *road mode*, in which a vehicle forwards packets to the next static node by passing packets repeatedly to the vehicles running ahead along the road, while the forwarding vehicle carries packets by itself if it cannot find any vehicle to pass packets to. On the other hand, when a vehicle gets into the communication range of a static node, it behaves in *intersection mode*, in which each vehicle exchanges control messages and data packets with the static node of the intersection, as described in Section 4.

Note that the intersections where static nodes are placed must be selected carefully. Because RDV by nature cannot relay packets without static nodes, we must place static nodes at every intersection where we expect relay packets to be relayed, i.e., where we expect a change in the directions of packet forwarding. Avoiding major intersections is possible only if alternative paths exist. As a result, static nodes in RDV would be placed at most of major intersections, whereas we can omit minor intersections. More specifically, in determining the node locations, one

of the feasible strategies is to consider the major flows of vehicles and locate nodes at the intersections of those flows. All we can do in packet delivery over vehicles is to make packets surely reachable within the area covered by major vehicle flows; and the strategy above achieves this coverage. Since the number of intersections along vehicle flows is usually far smaller than the number of physical intersections on maps, locating nodes at only a part of the intersections will drive RDV. Also, even if the vehicle flows are so dense that the number of intersections for nodes increases, we can design algorithms to omit stationary nodes, while keeping the reachability of packet delivery.

To describe RDV schemes, we introduce several terminologies. A *neighbor* for a static node is a node that is reachable by vehicles without visiting other static nodes. For example, in Fig. 1, (n_1, n_2) , (n_2, n_3) , (n_3, n_4) , etc. are neighbors. If a static node is reachable with high probability from another static node, via a vehicle, we regard those two nodes as located on the same street, and the distance between them is measured by the unit *hop*, the number of hops being the number of static nodes that a vehicle traveling between them passes by. For example, the distance between n_1 and n_4 is 3 hops in Fig. 1. The distance between two static nodes is also measured by the unit *carry*, the number carries being the number of vehicles required to carry packets between them. For example, the distance between n_1 and n_4 is 1 carry, and that between n_1 and n_{10} is 2 carries. When a packet is sent from n_1 to n_{10} , it has to transfer vehicles at n_4 . We call the transferring node the *relay node*.

4 RDV routing protocols

4.1 Overview

We now present an overview of RDV from the viewpoint of technical mechanisms. RDV has several control messages exchanged among nodes, with which it computes several tables to deliver packets.

Fig. 1 An example topology

Table 1 Local Statistic Table of n_2

Previous Hop	Next Hop	Probability
n_1	n_3	94%
n_1	n_5	3%
n_1	n_6	3%
n_5	n_6	86%
n_5	n_1	7%
n_5	n_3	7%
:	:	:

SVS is the first mechanism to introduce here. SVS investigates the local statistics, based on which RDV can achieve the preconfigured expectations on packet delivery ratio. SVS utilizes *hello* messages to enable nodes to detect their neighbors, to which SVS can forward packets with one vehicle, i.e., each node can detect neighbors, say, on the same street, and exchange *statistic* messages to compute the *statistic table* at each static node. See Fig. 1 for an example scenario. Table 1 shows an example of the statistic table computed at node n_2 . The *statistic table* includes the probability of vehicle movements among neighbor nodes; the 1st row in Table 1 means that a vehicle that visits n_1 before n_2 will reach the next-hop n_3 with a probability of 94%. Note that the sum of the 1st-3rd row is 100% in Table 1, but in the case of sparse deployment of static nodes, the sum would be far lower than this value.

Those probability values are carried by *single-carry* messages to identify nodes on the same street, i.e., the nodes that, with high probability, are reachable using one vehicle. By sharing the probability values with the nodes on the same street, each node computes the *single-carry table* shown in Table 2. This table holds the information needed to deliver packets to each node on the same street. For example, the 2nd row means that the vehicle that visits n_1 before n_2 will reach n_4 , which is placed 2 hops away, with a probability of 90%, and the routing metric from n_2 to n_4 is 0.022, which is computed from the number of vehicles in a unit time and the probability.

The information in the *single-carry* table is further advertised over the network using *multi-carry* messages.

Table 2 Single-carry Table of n_2

Dest.	Prev.	Dist.	Prob.	Metric
n_3	n_1	1 hop	94%	0.021
n_4	n_1	2 hops	90%	0.022
n_6	n_5	1 hops	86%	0.059
n_7	n_5	2 hops	80%	0.063
:	:	:	:	:

Table 3 Routing Table of n_2

Dest.	Relay	Prev.	Dist.	Prob.	Metric
n_3	n_3	n_1	1 carry	92%	0.021
n_4	n_4	n_1	1 carry	90%	0.022
n_7	n_7	n_5	1 carry	80%	0.063
n_{10}	n_4	n_1	2 carries	90%	0.085
:	:	:	:	:	:

Multi-carry messages are processed in the same manner as distance-vector routing to compute the shortest paths and the *routing table* for every destination of the network. Table 3 is an example of a *routing table*, which includes the information to send out packets to their destination. For example, the 4th row means that the destination n_{10} is reachable using 2 vehicles with 72% probability and that the next relay node along the shortest paths to n_{10} is n_4 . A vehicle that visits n_1 before n_2 will reach the next relay n_4 .

RDV delivers every packet using the routing table, in which the duplicated copies of the packet are sent out. In the example of Fig. 1, if n_6 sends a packet to n_9 , the packet first travels to the next relay n_2 , then visits n_4 , and finally reaches n_9 . Note that the direct link (n_6, n_9) is not used in this case because the metric of link (n_6, n_9) is high. The metric is computed from the amount of traffic and the delivery ratio, and small roads tend to have large routing metrics. Packet copies are made in the per-next-hop manner. For example, since a vehicle reaches n_7 from n_2 with 80% probability, n_2 has to send 3 packets if it expects 99% probability of reaching n_7 .

4.2 Behavior of vehicles and static nodes

In RDV, packets at a static node travel to their destinations via several vehicles and relay nodes. When a vehicle enters into the communication range of a static node, the vehicle first sends a set of control messages to the node, and then the node sends a set of control messages in turn. After this message exchange, the vehicle sends all required data packets to the node, and then the node sends data packets to the vehicle in turn. This transaction is carried out for every vehicle that visits a static node.

A vehicle and a static node exchange a fixed number of messages to limit the overhead of control messages. Specifically, a vehicle passes to a node two *hello* messages, one *statistic* message, N_{carry} *single-carry* messages, and $N_{carry}M_{MC}$ *multi-carry* messages in a transaction, where N_{carry} and M_{MC} are preconfigured values; N_{carry} represents the maximum number of hops included in a 1-carry distance, and M_{MC} represents the maximum number of *multi-carry* messages loaded in a vehicle. Similarly, a node passes to a vehicle one *hello* message, one *statistic* message,

one *single-carry* message, and M_{MC} *multi-carry* messages. With these values N_{carry} and M_{MC} , RDV limits the load of control messages per vehicle to within several KBytes.

When a vehicle is not in the communication range of any static node, it works as a *road mode*; The vehicle passes packets to the vehicles running ahead of it so as to forward packets to their next relay node along the road.

4.3 Computing local statistic tables for SVS

The local statistic table is held by each node as the key component of SVS, which is used when a vehicle visits the node to obtain the probability of the vehicle reaching each neighbor node. Since vehicles in general have high probability of continuing straight in intersections, the incoming direction is useful to predict the direction in which the vehicle will exit the intersection. RDV constructs the local statistic table using two types of messages, *hello* and *statistic* messages.

Note that, naturally, we have several exceptional cases in which many vehicles do not proceed straight in intersections. However, in the majority of the cases, we have major flows in which incoming directions determine the outgoing directions with high probability. In this work, we assume that major intersections have such major flows that determine outgoing directions. Of course, we may have inconvenient cases in which there is little correspondence in number between incoming and outgoing vehicles. In those cases, we would generate a large number of packet copies, sufficient for the preconfigured packet delivery ratio. Or, we could use additional information such as the vehicles' positions, acceleration directions, turn signals, etc. to estimate their direction, as discussed in Section 6.

4.3.1 Hello messages

A *hello* message is generated by a static node when a vehicle visits it, and passed to the vehicle. A *hello* message carries the address of the generator node to the nodes within a two-hop distance. Consequently, every node gets to know the addresses of all two-hop neighbors.

Each vehicle usually carries two *hello* messages, i.e., the latest two *hello* messages received. To maintain the latest messages, every vehicle, when it visits a static node, passes the two *hello* messages it holds to the node and receives a *hello* message from the node in return.

By collecting *hello* messages, each static node is able to count the number of in-coming vehicles in pairs of nodes (the previous node and the node preceding it). See Fig. 1 again for example. Node n_3 can count the number of vehicles coming through n_1 and n_2 in a given time unit. Similarly, n_3 can count the vehicles coming through n_5 and

n_2 , or n_6 and n_2 . If we let $c(n_a, n_b, n_c)$ be the number of vehicles that visit n_c after n_a and n_b , n_c can count the vehicles for every pair of neighbor nodes n_b and the two-hop nodes n_a .

4.3.2 Statistic messages

A *statistic* message is used to notify neighbors of the count information obtained through *hello* messages. When a vehicle visits a static node, the vehicle passes the holding *statistic* message to the node and, in return, receives the *statistic* message originated by the node. A *statistic* message includes all the count information on the originated node, i.e., the values $c(n_a, n_b, n_c)$ for every valid combination of n_a, n_b and n_c where n_c is the originated node, n_b is a neighbor of n_c , and n_a is a two-hop neighbor node of n_c . By means of *statistic* messages, every node gets to know the count information on all neighbors, i.e., n_b collects the values $c(n_a, n_b, n_c)$ for every valid pair of neighbors n_a and n_c . In case of Fig. 1, by receiving the *statistic* message from n_3 , n_2 gets to know the count information $c(n_5, n_2, n_3)$, $c(n_1, n_2, n_3)$ and $c(n_6, n_2, n_3)$.

With the information obtained from *statistic* messages, each static node computes the local statistic table. Let $p(n_a, n_b, n_c)$ be the probability that a vehicle visiting n_b after n_a will visit n_c . The statistic table of n_b is composed of the probabilities $p(n_a, n_b, n_c)$ for all possible combinations of n_a and n_c , where n_a and n_c are the neighbors of n_b . Each probability is computed from the count values collected by *statistic* messages with the following formula:

$$p(n_a, n_b, n_c) = \frac{c(n_a, n_b, n_c)}{c(n_a, n_b, *)}, \quad (1)$$

where $c(n_a, n_b, *)$ denotes the number of all vehicles that visit n_b after n_a in a unit time regardless of the next-hop node, which can be counted at node n_b .

4.4 Computing routing tables

The routing table at each node is computed in the manner of the distance-vector routing protocol. In this type of routing protocol, each node advertises a control message periodically that includes the information regarding reachable destinations, i.e., tuples of three values, a destination, the next-hop and the distance to the destination. In our routing scheme for vehicular networks, we exchange control messages among static nodes to perform the same computation as the distance-vector protocol. However, because broadcasting to all neighbor static nodes is not possible in vehicular networks, and also because we would like to identify streets to send the 'distance-vector' messages to all the static nodes within a 1-carry distance, we have to modify the

general distance-vector protocol in such a way that it works on vehicular networks.

For the first problem, i.e., broadcasting, we choose to carry each message using a distinct vehicle for each direction. However, since we do not assume a reliable link between static nodes, RDV duplicates the message and passes the duplicates to several distinct vehicles even those moving in the same direction to improve reliability of message delivery. Simultaneously, to limit the overhead from control messages, we restrict the number of messages loaded in a vehicle. As a result, RDV achieves a stable exchange of control messages within a limited message overhead.

The second problem, i.e., street identification, is solved by utilizing a special message called a *single-carry* message. By identifying static nodes that are located on the same street using the *single-carry* message, RDV enables nodes to contact all the other nodes on the same street using a single vehicle, which reduces the traffic between vehicles and nodes, and improves the capacity of the network.

4.4.1 Single-carry messages

As mentioned above, a node in RDV identifies the static nodes that are located on the same street and are reachable using a single vehicle with high probability. The basic role of *single-carry* messages is to propagate along a street the probability information stored in the local statistic table so as to compute the packet delivery probability between static nodes located several hops away.

For a specific description, see Fig. 2, where a vehicle is moving from a static node n_1 to n_5 . Every time the vehicle visits a static node, it receives the *single-carry* message of the node and carries it to the succeeding N_{carry} hops. In this example, the vehicle receives values $p(n_3, n_2, n_1)$, $p(n_4, n_3, n_2)$, and $p(n_5, n_4, n_3)$. Simultaneously, by collecting all *single-carry* messages carried by every visiting vehicle, each static node obtains the probability information issued by all the static nodes within

N_{carry} -hop distance. In this example, n_4 obtains the values $p(n_3, n_2, n_1)$ and $p(n_4, n_3, n_2)$ from the vehicle, and n_4 has $p(n_5, n_4, n_3)$ in its statistic table. From those values, n_4 is able to compute the probability $p(n_5, n_4, n_1)$, which is the probability of reaching n_1 if n_4 passes a packet to a vehicle coming from n_5 , computed as $p(n_5, n_4, n_1) = p(n_3, n_2, n_1) \cdot p(n_4, n_3, n_2) \cdot p(n_5, n_4, n_3)$. In general, with *single-carry* messages, every node n_b can compute $p(n_a, n_b, n_c)$ for every pair of its neighbors n_a and any node n_c that is located on the same street within N_{carry} -hop distance.

The specific process of RDV to treat *single-carry* messages is as follows: when a vehicle visits a static node, it first passes all the *single-carry* messages to the node, and receives the *single-carry* message of the node in return. Then, the vehicle deletes the oldest message from among the $N_{carry} + 1$ holding messages to keep the latest N_{carry} *single-carry* messages. The node, say n_b , on the other side, computes the probabilities $p(n_a, n_b, n_c)$ for all valid combinations of a neighbor n_a and a node n_c that are located within N_{carry} -hop distance of n_b . If the probability is higher than a certain threshold T_{SC} , the results are stored in the single-carry table.

4.4.2 Multi-carry messages

Once a node gets to know the set of static nodes reachable in a 1-carry distance with *single-carry* messages, the information is advertised in the manner of a distance-vector routing protocol using *multi-carry* messages. The *multi-carry* messages advertise a set of reachable destinations to all the nodes located within a 1-carry distance. By exchanging the *multi-carry* messages repeatedly, every node constructs its routing table, which includes all the reachable destinations with the corresponding next-hops, distances and delivery probabilities.

For the specific process, see Fig. 3, which illustrates the example scenario for computing the routing-table entry at n_1 - n_4 for the destination n_d . Node n_1 knows that n_d

Fig. 2 Single-carry Messages

Fig. 3 Multi-carry Messages

is reachable within a 1-carry distance, so it advertises a *multi-carry* message including this information by passing the message to the vehicle traveling on the road to n_2 . The message includes the destination-node address together with the corresponding relay node, the distance, and the probability of reaching the destination. Thus, by receiving the message along the road, n_2 and n_4 get to know that nd is reachable in a 2-carry distance by relaying packets at n_1 . Similarly, the message from n_2 notifies n_3 that nd is reachable in a 3-carry distance over the relay node n_2 . Note that the message from n_2 is also received by n_4 , but this message is ignored since n_4 has received the *multi-carry* messages issued by n_1 , which implies the existence of shorter paths (i.e., 2-carry distances) to nd . In this way, RDV always selects the shortest path among the possible paths to the same destination. We introduce below, in Section 4.6, a routing metric to re-define the distance between static nodes used in the shortest-path computation. This redefining can be treated in the same way by utilizing the metric-based distance instead of a hop-based distance.

Table 4 is the routing table computed at n_4 of this example. Note that, in this table, the probability of reaching each destination appears. Since *multi-carry* messages include the probability of packets traveling from the relay node to the destination, using the information in its own single-carry table, a node x that receives the message can compute the probability of traveling from x to the destination. From the probability in the routing table, RDV determines the number of packet copies to send out to

control the end-to-end packet delivery ratio as described in Section 4.5.

The specific operations to process *multi-carry* messages are as follows. When a vehicle visits a static node, the vehicle first passes all the $N_{carry}M_{MC}$ loaded messages to the node, and in turn receives M_{MC} messages from the node. Then, the vehicle deletes the oldest M_{MC} messages, which are received at the N_{carry} -th previous node, thereby limiting the number of messages loaded by $N_{carry}M_{MC}$. Note that a static node may have a larger number of *multi-carry* messages than M_{MC} , the limitation on messages to pass. In that case, the node selects M_{MC} messages randomly and passes them to each vehicle visiting the node. On receiving new *multi-carry* messages, a static node first updates its database with the messages and then recomputes its routing table in accordance with distance-vector protocols. Note that, although this recomputation occurs for every visiting vehicle, the computational cost is not large because it is done by simply comparing two (i.e., old and new) entries per destination to select the least distant one. After the routing-table computation, the node prepares the updated *multi-carry* messages to pass them to visiting vehicles.

4.5 Adaptive packet duplication (APD)

To achieve the preconfigured expectations on packet delivery ratio, RDV has a mechanism called APD that sends out several copies of a packet to secure the preconfigured probability of packets reaching their destinations. RDV controls the delivery ratio in a relay-node manner. That is, the source and each relay node in the shortest path to the destination compute the required delivery ratio for the next relay node, and send the required number of packet copies to meet the ratio. Every packet in RDV has a next-relay-node field in the packet header, so that the transmitted packets are received by the relay node unless they are lost for some reason. If the relay node receives one of the copies, the packet is considered to have reached the relay node, and the relay node ignores copies of the same packet arriving after that. By performing this for each relay node, RDV

Table 4 Routing Table of n_4

Dest.	Relay	Dist.	Prev.	Prob.
n_d	n_1	2 carries	n_5	0.72
n_1	n_1	1 carry	n_5	0.8
n_2	n_2	1 carry	n_5	0.9
n_3	n_2	2 carries	n_5	0.81
n_5	n_5	1 carry	n_2	0.95

finally achieves the preconfigured end-to-end delivery ratio for every destination.

Specifically, let P be the preconfigured delivery ratio which suffices, and P_{relay} be the required delivery ratio for the next relay node. Then, $(P_{relay})^c \geq P$ must be satisfied where c is the distance for carrying to the destination. Thus, we let $P_{relay} = P^{\frac{1}{c}}$ in RDV. On the other hand, let C_{relay} be the number of copies and p be the probability of reaching the next relay node that is stored in the routing table. Then the following formula holds:

$$1 - (1 - p)^{C_{relay}} \geq P_{relay}. \quad (2)$$

Consequently, the number of required packet copies C_{relay} to reach the relay node with probability larger than P_{relay} is expressed as

$$C_{relay} = \text{ceil} \left(\frac{\log(1 - P_{relay})}{\log(1 - p)} \right), \quad (3)$$

where $\text{ceil}(\cdot)$ is the ceiling function.

The behavior of nodes and vehicles with duplicated packets is as follows. When a packet is received by a static node (or generated at the node), it first refers to the routing table to know the required number of copies C_{relay} , and enqueues the packet and C_{relay} pair to the its output queue. When a vehicle visits the node, the node examines the pairs in the queue starting from the tail of the queue, and judges whether the packet is to be passed to the vehicle. Specifically, the following steps are taken for each pair in the output queue until the buffer of the vehicle becomes full. (1) The node first refers the routing table to obtain the previous node of the vehicles passing the packet. (2) If the previous node of the visiting vehicle matches the previous node in the routing table, it passes a copy of the packet, and increases the count of copies passed to vehicles. (3) If the count is equal to C_{relay} , it removes the packet from the queue.

When a vehicle leaves an intersections, it passes packets to the vehicles ahead of it repeatedly to reduce the delivery delay in reaching destinations. Here, note that a vehicle does not receive the copies of the same packet that the node has in its buffer.

Once a packet is received by a relay node, it records the packet for a certain time and ignores the copies of the same packet received at the node. By receiving the same packet only once, RDV prevents sending wasteful packet copies to the next relay node; that is, for each packet, the node sends exactly the same number of copies as is sent by C_{relay} to the next relay node. By repeating to pass packets to the relay nodes, packets finally reach their destinations.

RDV also uses the TTL field to avoid wasteful transmissions. When a new relay node is set to a packet, the distance in hop-count to the relay node is set as TTL. Since

the TTL value is decreased each time the carrying vehicle visits a node, packets that failed to reach their relay nodes are silently dropped when TTL reaches 0.

4.6 Traffic-amount-based routing metrics

We designed RDV so as to provide reliable packet delivery with low end-to-end delay. To achieve this in sparse networks, we introduce a routing metric based on the bandwidth of links, i.e., based on the density of vehicles in roads. The metric is measured for every pair of nodes within 1-carry distance, and used in the shortest-path computation. Note that busy roads are suitable for RDV to provide reliable and low-delay communications since RDV transmits multiple copies of a packet into a single path.

For an example calculation of the metric, suppose a pair of nodes n_s and n_r are located in a 1-carry distance. Node n_s first finds the previous-hop n_p that has the highest probability to reach n_r , i.e., $n_p = \text{argmax}_{n_x} p(n_x, n_s, n_r)$. Then, the number of vehicles in the time unit that can carry packets from n_s to n_d is $c(n_p, n_s, *)p(n_p, n_s, n_r)$, where $c(n_p, n_s, *)$ is the number of vehicles from n_p to n_s observed by *hello* messages. Now we utilize the traditional custom used in the Internet routing to incorporate link bandwidth into the shortest-path computation; link costs in link-state routing protocols are set as the inverse of the link bandwidth. Consequently, n_s computes the routing metric of a link (n_s, n_r) as

$$M(n_s, n_r) = \frac{1}{c(n_p, n_s, *)p(n_p, n_s, n_r)}. \quad (4)$$

The metric values are computed at every node n_s when the protocol computes $p(n_p, n_s, n_r)$ in processing *single-carry* messages. In sending *multi-carry* messages, RDV includes the corresponding metrics in the messages. This means that we just change the weight of each link in the path computation. The weight of each link in the normal RDV is 1 whereas the weight is replaced with the metric value in the metric-based RDV. As a result, RDV computes the shortest-paths in terms of the sum of metric values for more optimized path selection.

Note that some smoothing preprocess such as taking the moving average should be applied on $c(n_p, n_s, *)$ in order to prevent heavy fluctuation of either metric values or communication paths. Also, since the metric transits as time passes, it may cause the so-called count-to-infinity problem in distance-vector routing schemes. However, we can avoid the count-to-infinity problem by introducing the technique using predecessor-node information [18]. Omitting some detail here, this technique is easily incorporated into the proposed scheme.

5 Evaluation

5.1 Methods

We evaluate the proposed method through simulation. The objective of our evaluation is to measure the communication performance of RDV in a relatively sparse scenario. We have, specifically, two concerns about the performance of RDV. The first one is whether RDV provides reliable communication within the managed delivery ratio, and within the relatively small overhead of copied packets. The second concern is the convergence speed of RDV paths in face of topology changes or traffic trends in measuring the flexibility of RDV in dynamic environment. We prepared two scenarios corresponding to these two questions.

As a simulation environment, we developed a network simulator with C++ language to simulate several routing protocols in relatively sparse vehicular networks. We are not concerned about PHY and MAC protocols since the network behaves mostly as a DTN. We only assume that a certain amount of data exchange is certain when a vehicle visits a static node.

We obtained a real road map of Kyoto from Open Street Map [19] and removed smaller streets to retrieve the major roads, as shown in Fig. 4 (we see below the locations of points x , s_1 , s_2 , d_1 , d_2). SUMO [20] is used for vehicle traffic generation on the map. We find 104 intersections in all. At all of these we placed traffic signals and static nodes, leaving static nodes at only major intersections. We have 20 road-ends, and at every road-end we generate vehicles with a constant time interval for which the maximum speed is set at 50[Km/h]. We adopt a probabilistic mobility model: at intersections, 90% of vehicles go straight, and 5% turn right, 5% left. If vehicles are not allowed to go straight at an intersection, 50% turn left and the rest right. Each vehicle has a buffer (i.e., a transmission queue) that holds 128 data packets. In RDV, each static node has a transmission queue for each previous hop (remember that RDV recognizes previous hops to select which vehicles to pass packets to), the size of the queue being 1280 packets each. Nodes and vehicles have a communication range of 50[m], so vehicles work as *intersection mode* if they are within the range of a static node, and as *road mode* otherwise.

Also, in RDV, every node updates its statistics and the routing tables every time the node exchanges messages with a vehicle. This allows the most sensitive updates of the statistics and the routing table to be made, while we can make choices about less frequent updates to reduce the computational load.

We compared the proposed RDV scheme with SADV, GPSR, and Epidemic routing. In the implementation of SADV, we measured, in the preliminary tests, the average delay of each link, i.e., of each pair of neighbor static nodes,

Fig. 4 Kyoto city map used in simulation

and used the fixed values in the shortest-path computation. In SADV, we set TTL as the length of the shortest path plus 2 and 4, which allows using paths 2- and 4-hops longer than the shortest path as detour routes. Note that the delivery ratio improves as TTL increases because it allows detour routes to reach destinations, whereas the overhead grows due to increase of packet copies. Note that when TTL=4, it achieved the best delivery ratio in our preliminary tests, meaning that larger TTL values do not bring visible improvement. In GPSR, we incorporated the *carry-and-forward* behavior to improve delivery ratio in sparse scenarios; that is to say, a vehicle passes packets when suitable neighbors exist. Without suitable neighbors the vehicle holds the packets. In GPSR and Epidemic, for fair comparison, static nodes at intersections participated in routing as the same 'node' as vehicles that disseminate packets. In RDV, we set the expected delivery ratio at $P = 99\%$, $M_{MC} = 20$, $N_{carry} = 10$, and $T_{SC} = 70\%$.

We started transmitting data packets for 100[sec] after the stable routing tables were obtained. In the first scenario, we measured the communication performance. During the simulation time we generated 10000 packets exchanged among randomly-selected static node pairs with a constant time interval. To consider the uncertainty of vehicle behavior under the sparse deployment of static nodes, vehicles were made to leave major roads and enter minor streets or parking lots along the road, and then return again to the major roads. Because we do not have static nodes at minor intersections in the sparse scenario, we cannot trace a part of the vehicles that escapes into minor streets, etc.,

so that we treat these cases as probabilistic phenomena. For this purpose, at each section of a road between two static nodes (i.e., at each links), we probabilistically selected vehicles and flushed (deleted) all packets and messages held by them. This flush operation pretends that, as one vehicle leaves the road, another vehicle simultaneously joins. We let the probability of flushing at each road section be 0%, 1.0% and 2.5% to measure the effect of uncertainty of vehicle movement. We generated 3–8 vehicles per minute per road-end, which is a relatively sparse traffic state. We took the average of 10 repetitions to show the results.

In the second scenario, with which we measured the effect of topology changes, different from the first scenario, we generated two flows: from node s_1 to d_1 , and s_2 to d_2 , as shown in Fig. 4. We assume that an accident occurs at intersection x , and thereafter blocks all roads connecting to x thereafter. We observe the route changes after the accident to estimate the required time for path convergence, and simultaneously measure the delivery delay and the packet

loss ratio to see the effect of the accident on communication performance.

5.2 Results on communication performance

In Fig. 5, we compare the communication performance under variation of vehicle density, in which delivery ratio, delivery delay, and the number of packet copies are shown for each value of flush ratio 0.0%, 1.0%, and 2.5%. We first provide here the information that the average length of RDV paths between source and destination nodes in *carry* (i.e., the number of vehicles required) is 3.92 carries in the case of flush ratio 0%, 4.40 carries for 1%, and 5.02 carries for 2.5%. Note that the distance in *hop* (i.e., the number of static nodes in the path) over which a single vehicle can carry packets is shortened as flush ratio increases.

We see in Fig. 5a, b and c that RDV maintains 99% packet delivery in every flush ratio, where packet delivery of SADV degrades as flush ratio increases. This clearly

Fig. 5 Comparison of communication performance

Fig. 6 Performance of traditional methods

indicates that RDV achieves the preconfigured packet delivery ratio (i.e., 99% in our scenario) even under the uncertainty of vehicle movement. These figures also show that the TTL value on SADV contributes to delivery ratio improvement, which is because larger TTL values allow packets to reach their destinations along longer paths, in compensation for paying a larger overhead of packet copies.

As for the overhead of packet copies, we see in Fig. 5d, e and f that RDV has a much lower overhead than SADV, which is because RDV sends the copies only along the shortest paths, whereas SADV sends them over a wide geometric region between the source and the destination nodes. Note also that the overhead of SADV reduces as the flush ratio increases, which is because many packets that will create their own copies are lost before creating them. Conversely, in RDV, the overhead increases as the flush ratio increases since RDV generates a larger number of packet copies to achieve the preconfigured delivery ratio.

Figure 5g, h and i show the performance on delivery delay. Although the delay reduces as the vehicle density increases, RDV always takes larger values than SADV. Moreover, in RDV the delay increases as the flush ratio increases while it remains at almost the same value in SADV. This is because RDV has to forward packets with the vehicles that follow the same path. When a vehicle carrying a packet is lost, an RDV node to receive the packet has to

wait for the next vehicle moving along the same road, which increases the delivery delay. Note that, since the network is basically a DTN, this difference in delivery delay would not be of great importance in practice, although improvement here is one of the future tasks for RDV.

In Fig. 6a, b and c, we show the delivery ratio of two traditional routing schemes GPSR (with *carry-and-forward* behavior) and epidemic routing in the same scenario. We see that the performance of these two in delivery ratio is very poor, indicating that they do not work to provide reliable services on a sparse scenario in vehicular networks.

Figure 7 shows the performance of RDV and SADV under pressure of communication load. We see specifically, the delivery ratio when we increase the load; that is, we see the number of generated packets when fixing the vehicle density at 4 [vehicles/min]. When the flush ratio is 0% and 1%, the delivery ratio ranges around high values over 95%. RDV keeps the preconfigured delivery ratio 99% until the network saturates, i.e., until the generated packets exceed the network capacity. In the case of the 1% flush ratio, the network saturates at a load lower than 0%, which is because RDV creates a larger number of packet copies to achieve the preconfigured delivery ratio. In the case of the 2.5% flush ratio shown in Fig. 7c, we see that the performance of SADV dramatically decreases, whereas RDV shows a trend similar to the cases of a 0-1% flush ratio. This is because SADV uses multiple paths to reach the destination, and if

Fig. 7 Performance over communication load

the flush ratio exceeds a certain level, the packets on all the paths may be lost. This means that SADV is vulnerable to the flush operation, i.e., vulnerable to the uncertainty of vehicular movement.

From the results above, we clarified that RDV performs the best among all the other existing schemes, in both packet delivery ratio and packet-copy overhead. RDV has the ability to keep a high delivery ratio even under the uncertainty of vehicle behavior, making it a reliable infrastructure for various practical services.

5.3 Results on path-convergence performance

We measure the performance of RDV on the speed of path convergence in a scenario with a topology change as intended. We generated two flows: Flow-1 goes from s_1 to d_1 and Flow-2 goes from s_2 to d_2 , as shown in Fig. 4. Each flow begins to generate packets at a rate of 12 [packets/min] at time 3600[Sec] after beginning simulation. We assume that a traffic accident occurs at intersection x at time 4100[Sec] and blocks all streets connecting to x . We compared the three cases where $M_{MC} = 10, 15$ and 20 .

Figure 8 shows the number of lost packets in each of two flows, the two compared for each value of M_{MC} , presented here for each time interval on packet generation. Figure 8a and b is the case $M_{MC} = 10$, in which packet loss due to the accident lasts about 400 [sec] and 300 [sec], respectively. Since we lost all packets (i.e., one packet per 5 [secs]) during that time period, we see that RDV requires about 300–400 [sec] convergence time to update paths from old to new conditions. Figure 8c and d show the corresponding

results in the case of $M_{MC} = 15$. We see that the convergence time significantly reduces as M_{MC} increases. This is because increasing the capacity of each vehicle to carry *multi-carry* messages enhances the propagation speed of topology-change information. Figure 8e and f show the case of a larger capacity: $M_{MC} = 20$. In this case, the convergence time is almost the same as that of $M_{MC} = 15$, which indicates that the value $M_{MC} = 15$ provides the sufficient capacity to support *multi-carry* messages for this network of the size of Kyoto City.

Note that the values of M_{MC} must be determined considering a trade-off between message load and the convergence time. In the case of this scenario with the size of Kyoto City, the value $M_{MC} = 15$ was proved to be sufficient. Under these conditions, the load of $N_{carry}M_{MC} = 10 \times 150 = 150$ *multi-carry* entries on each vehicle is required. Because the load for the message entries is actually small, we have only to use the large value of M_{MC} to provide sufficient capacity for *multi-carry* information.

From the results, we see that the convergence time that RDV requires is about 100–200 [sec] with the sufficiently large values of M_{MC} . This scale of convergence time would be acceptable in the real case of accidents. Note that the convergence time differs according to the time duration, based on which the statistic tables are built. This value should be determined considering the trade-off between statistical reliability and convergence time. In this scenario, we adopt 25 [min] for this duration, which is the smallest value that made the values in statistic table stable in our preliminary test.

Fig. 8 Performance on paths convergence

As additional discussion on the suitable parameter values, we also note that the network load depends on several other parameters such as T_{SC} and N_{carry} . T_{SC} , which is defined as the minimum delivery ratio to establish a single link, controls the number of duplicated packets. First, T_{SC} must be small enough to maintain networks connected even without clear traffic patterns, as happens, for example, in rural areas. (Note that a large T_{SC} would not allow networks for a successful delivery ratio in many cases.) Conversely, under this constraint, it is preferable to use a large T_{SC} value to lower the traffic load of duplicated packet copies. Note too that N_{carry} should also be sufficiently large to allow RDV to create links of sufficient length along major streets in which most vehicles move with typical traffic patterns. Using long links in urban scenarios where we have a large amount of data traffic will significantly reduce the number of packet exchanges between static nodes and vehicles. Administrators should determine the parameter values by considering the above concerns.

6 Discussions on practical deployment of RDV

There are several issues on practical deployment of RDV. We first discuss its scalability. If RDV is deployed in a large area in which each packet is relayed by many vehicles, it may require a very large number of packet copies to achieve the preconfigured delivery ratio. (Note that SADV also suffers from the scalability problem due to packet copies as well.) RDV is able to solve this scalability problem by connecting a small number of static nodes to wired networks, the Internet for example. Suppose we prepare one ‘gateway’ that connects to the Internet for each ‘unit’ area. Note that we intend to prepare the minimum required number of gateways and that only one gateway per unit area will suffice to enhance RDV scalability. By providing short-circuits among all wired static nodes, the diameter of vehicular networks is greatly reduced, so that the length of the paths on which packets travels reduces, as does the number of packet copies. This significantly improves the scalability. Specifically, in RDV, we can advertise the IDs (e.g., IP addresses) of wired nodes as ‘default gateways’ that forward all packets with unknown routes to the destinations. By limiting the distance-vector route propagation within a certain distance, and by locating gateways with the proper density, the gateways connected by wired networks can cover a large area by connecting all vehicles within a limited distance. As a result, we can deploy RDV in a large area without the scalability problem.

Second, we discuss here the last-mile connection so as to deploy RDV in practice. Since RDV provides communications between static nodes, it must work with

other routing protocols that support first and last mile connections. As the first mile connection from source vehicles to a static node, vehicles can directly connect to the static nodes they meet. Or, if we apply multi-hop forwarding, several proposals are available, such as D-Greedy [14] or PROMPT [15] that provide multi-hop connection to RSUs for vehicles.

In contrast, regarding last-mile connections, we have to discuss which communication types to support. The first one we would consider is *geocast*, in which information is delivered to all vehicles found in the *broadcast area*, which is geometrically specified by the source node. *Geocast* is one of the most important types of communications, since many delay-tolerant applications would expect this type of communications. To support geocast with RDV, we can combine any packet dissemination methods with RDV, whereby packets are delivered first to all the static nodes within the broadcast area, and the static nodes propagate the packets to the entire broadcast area. Packet delivery to multiple static nodes is easily done by sending the packet separately to each static node, or by extending RDV to deliver packets to support multicast. As a dissemination scheme from the static nodes to destination vehicles, we can use the flooding-based schemes surveyed in [21]. Or, we can deploy several *geocast* routing schemes [22] to provide delay-tolerant dissemination from a static node to the vehicles around it.

Unicast, which delivers packets to a specific vehicle, is a rather challenging type of communication in delay-tolerant networks. Wu, et al. showed in [23] that the wired infrastructure strongly supports the *unicast* communication in vehicular networks in terms of delivery ratio. RDV would support *unicast* as well. As a last-hop communication scheme, TSF [16] has been proposed. It improves delivery ratio by selecting a rendezvous point as the destination place of packets which the vehicles will likely approach in consideration of its trip trajectory.

We would like to discuss as well the effect of vehicle density on communication performance. Our simulation focuses on the case of sparse vehicle scenarios where networks basically behave as DTNs, and multi-hop communications are basically not possible. We emphasize the main difficulty of the conventional routing schemes in the sparse-vehicle cases so that our contribution is seen clearly. In contrast, in case of urban scenarios where vehicle density is high, since the opportunity for multi-hop communications increases, packet delivery ratio could improve greatly under any routing schemes. Although RDV still effectively works to satisfy high expectation for delivery ratio, the difference with conventional schemes is less. In dense scenarios, the main criteria to measure communication performance change to those for real-time communications: throughput, collision ratio, etc. Since

Fig. 9 An inconvenient case

those measurements are deeply related to MAC or PHY layer protocols, research on for dense vehicle scenarios requires another solid body of studies. This should be an interesting task for the future.

Finally, we discuss the possibility of more intelligent decisions on relay-vehicle selection. Although RDV is formulated as a pure topological routing scheme in which no practical information (position, moving direction, maps, turning signals, or route information in navigation systems) is used, such information could improve packet delivery performance. It is possible to design routing schemes that incorporate such information. However, note that, even still, guaranteeing the expectations on packet delivery ratio is not possible: the packet duplication mechanism of RDV and its base mechanisms will continue to play their important role in providing reliable communications.

Still, it will be valuable to note that RDV has a limitation that will be overcome with the addition of practical information of the type just mentioned. See Fig. 9 for a problem case for RDV, in which n_7 is hardly visited since vehicles that visit n_1 before n_2 will then go to n_3 with high probability. So, even if street (n_2, n_5) has a certain amount of traffic, packets hardly ever reach n_7 because the probability of vehicles turning at n_2 to reach n_7 is low. This is true too when the destination of packets is on some small streets. Consequently, it is essential that the reachability of a destination in RDV depends on the network topology and the traffic patterns around the destination. Solving this problem in combination with the above feasible information is one of the important tasks for future work.

7 Conclusion

We proposed a new routing scheme RDV for static-node-assisted vehicular networks that provides reliable communications so as to support certain practical applications. Unlike the conventional link-state-based approaches, RDV adopts the distance-vector routing that enables us to handle the probability of packets reaching the nodes located at several-hops away on the same street. By creating the

required number of packet copies based on the statistics obtained from traffic observations at each node, RDV guarantees achieving the expected delivery ratio even in the sparse scenarios.

Simulation results showed that RDV provides reliable communications with lower overhead of packet copies than the conventional schemes, even in the presence of uncertain behavior of vehicles under sparse deployment of static nodes. Through this study, we showed that, with the assistance of static nodes located at major intersections, we can realize the reliable communication with a guarantee on the expected packet delivery ratio in the sparse scenarios.

As for the future work, we have to bear in mind that the delivery delay in RDV is relatively long due to the need to wait for a next vehicle along the single shortest path to forward packet copies. Multi-paths utilization is a possible way to reduce delivery delay. This would also provide load balancing functionality. Another interesting task is to enhance the pure topological routing scheme RDV with other feasible information such as position, maps, moving directions, etc., in order to lower the overhead to provide reliable routing.

Acknowledgements We sincerely appreciate the great support of Professor David Alfred Sell for beneficial advices and proofreading on our work.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. Karagiannis G, Altintas O, Ekici E, Heijenk G, Jarupan B, Lin K, Weil T (2011) Vehicular networking: a survey and tutorial on requirements, architectures, challenges, standards and solutions. *IEEE Commun Surv Tutor* 13:4
2. Zhao J, Cao G (2008) VADD: vehicle-assisted data delivery in vehicular ad hoc networks. *IEEE Trans Veh Technol* 57(3):1910–1922
3. Wu H, Fujimoto R, Guensler R, Hunter M (2004) MDDV: a mobility-centric data dissemination algorithm for vehicular networks. In: *Proc. of VANET'04* pp 47–56
4. Ding Y, Xiao L (2010) SADV: static-node-assisted adaptive data dissemination in vehicular networks. *IEEE Trans Veh Technol* 59:5
5. Lochert C, Mauve M, Fusler H, Hartenstein H (2005) Geographic routing in city scenarios. *ACM Sigmobile Mob Comput Commun Rev* 9(1):69–72
6. Lee K, Haerri J, Lee U, Gerla M (2007) Enhanced perimeter routing for geographic forwarding protocols in urban vehicular scenarios. In: *Proc. IEEE Globecom workshops*, pp 1–10

7. Karp B, Kung HT (2000) GPSR: greedy perimeter stateless routing for wireless networks. In: Proc. Mobicom'00, pp 243–254
8. Lochert C, Hartenstein H, Tian J, Fussler H, Hermann D, Mauve M (2003) A routing strategy for vehicular ad hoc networks in city environment. In: IEEE Proc. of intelligent vehicles symposium, pp 156–161
9. Seet BC, Liu G, Lee BS, Foh CH, Wong KJ, Lee KK (2004) A-STAR: a mobile ad hoc routing strategy for metropolis vehicular communications, networking 2004, lecture notes in computer science, vol 3042. Springer, pp 989–999
10. Vahdat A et al (2000) Epidemic routing for partially connected ad hoc networks. Duke University Tech Report CS-2000-06
11. Yoo J, Choi S, Kim CK (2009) The capacity of epidemic routing in vehicular networks. IEEE Commun Lett 13(6):459–461
12. Jerbi M, Senouci SM, Meraihi R, Ghamri-Doudane Y (2007) An improved vehicular ad hoc routing protocol for city environments. In: Proceedings of IEEE international conference on communication (ICC), pp 3972–3979
13. Zhang XM, Chen KH, Cao XL, Sung DK (2016) A street-centric routing protocol based on micro topology in vehicular ad hoc networks. IEEE Trans Veh Technol 65(7):5680–5694
14. Skordylis A, Trigoni N (2008) Delay-bounded routing in vehicular ad-hoc networks. In: Proc. MobiHoc'08
15. Jarupan B, Ekici E (2010) PROMPT: a cross-layer position-based communication protocol for delay-aware vehicular access networks. Ad Hoc Netw 8:489–505
16. Jeong J, Guo S, Gu Y, He T, Du DHC (2012) Trajectory-based statistical forwarding for multihop infrastructure-to-vehicle data delivery. IEEE Trans Mob Comput 11:10
17. Liya X, Chuanhe H, Jinhai W, Junyu Z, Lianzhen Z (2014) An efficient traffic geographic static-node-assisted routing in VANET. J Netw 9:5
18. Cheng C, Riley R, Kumar SPR, Garcia-Luna-Aceves JJ (1989) A loop-free extended Bellman-Ford routing protocol without bouncing effect. In: Proc. of SIGCOMM'89, pp 224–236
19. Open Street Map, <https://www.openstreetmap.org/>
20. Behrisch M, Bieker L, Erdmann J, Krajzewicz D (2011) SUMO – simulation of urban mobility: an overview. In: Proc. of SIMUL2011, pp 23–28
21. Panichpapiboon S, Pattara-atikom W (2012) A review of information dissemination protocols for vehicular ad hoc networks. IEEE Commun Surv Tutor 14:3
22. Maihofer C (2004) A survey of geocast routing protocols. IEEE Commun Surv Tutor 6(2):32–42
23. Wu Y, Zhu Y, Li B (2012) Infrastructure-assisted routing in vehicular networks. In: Proc. IEEE Infocom'12
24. IEEE 802.11p, Amendment to Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std. IEEE 802.11 version 2007, (2007)
25. IEEE 1609.1-1609-4, trial-use standard for wireless access in vehicular environments (WAVE). IEEE Std. IEEE, 1609, 2006
26. Electronic road pricing, https://en.wikipedia.org/wiki/Electronic_Road_Pricing (referred on July 25th, 2017)
27. Fujimoto A, Sakai K, Hamada MOS, Handa S, Matsumoto M, Takahashi K (2008) Toward realization of SMARTWAY in Japan. In: Proc 15th World Congress on intelligent transportation systems (ITSWC'08)