Virtual Vertex Routing (VVR) for Course-Based Vehicular Ad Hoc Networks

Hojin Lee, Youndo Lee, Taekyoung Kwon, and Yanghee Choi
School of Computer Science and Engineering
Seoul National University, Seoul, Korea
Email: {lumiere, ydlee}@mmlab.snu.ac.kr and {tkkwon, yhchoi}@snu.ac.kr

***Abstract*— In Vehicular Ad Hoc Networks (VANETs), geographic routing protocols appear to be a fascinating option since they have generally low delay and small routing overhead and GPS devices are becoming affordable. However, we reveal that geographic routing protocols suffer from routing holes when nodes are distributed only on lines such as cars on roads, trains on rails, and ships on courses. To tackle this problem, we propose a novel geographic routing protocol, Virtual Vertex Routing (VVR), which uses the information of the lines. Using graph formulation, we introduce a new concept, the proximity of a vertex (or a virtual vertex). An intermediate node in this proximity performs routing toward the destination by Floyd algorithm. For routing holes, we propose two countermeasures: greedy routing (VVR-GR) and face routing (VVR-FR). The latter can guarantee the packet delivery. Extensive simulations are performed to show that VVR outperforms GPSR and AODV.**

I. Introduction

A geographic routing protocol uses the location information of neighbor nodes and destinations. It chooses a locally optimal next hop in a distributed manner; i.e., it selects the neighbor node closest to the destination. Geographic routing protocols typically have the lower delay of packet delivery than reactive routing protocols [10] (AODV [5], DSR [6], TORA [7]). This is because they have no route acquisition delay such as RREQ flooding. In terms of the routing overhead, geographic routing protocols also surpass both reactive and proactive routing protocols [10] (DSDV [8], OLSR [9]) because control packets in geographic routing protocols are only HELLO messages which are small size containing sender's ID and sender's location only and are exchanged locally, i.e., within one hop.

Due to the above advantages, when the location information is accessible or available, geographic routing protocols are attractive. In Vehicular Ad Hoc Networks (VANETs), communication entities are vehicles. Cars with GPS equipments increase more and more, and almost all buses and ships are already equipped with GPS devices for the transportation and navigation services. Consequently, geographic routing protocols appear to be a fascinating option for VANETs.

In this paper, we reveal that existing geographic routing protocols are not appropriate for VANETs. They assume that nodes are located on a two-dimensional area uniformly at random. However, almost all vehicles lie on specific lines of the area, such as cars on roads, trains on rails, and ships on courses. In this situation, geographic routing protocols frequently fall into the local minimum (a routing hole), where no neighbor nodes are closer to the destination than the node itself. The frequent occurrence of routing holes degrades the performance of routing (delivery ratio, delivery delay) severely.

To tackle the above problem, we propose a novel geographic routing protocol - Virtual Vertex Routing (VVR). We assume that all the lines (e.g., roads, rails, or courses at sea) are known to every node. Actually, this information can be provided by the navigator system, the digital road map, or the map of courses which are usually embedded in vehicles. Using the line information, VVR greedily forwards packets to the intermediate target, which is updated at certain points of the path. We compare VVR with GPSR [1] [2] and AODV [5] which are the representative geographic routing protocol and reactive routing protocol, respectively. We exclude proactive routing protocols for comparison because the proactive schemes are not suitable for high mobility environments due to the large delay of routing table convergence [14].

The rest of this paper organized as follows. In Section II, we briefly explain the overview of geographic routing and illustrate the routing hole problem in VANETs. Our proposed scheme is described in Section III and the performance evaluation is presented in Section IV. Finally, the paper is concluded in Section V.

II. Preliminaries and Motivation

A. Greedy Forwarding and Routing Hole

If a node knows the locations of its neighbors and a destination, it can make a locally optimal, greedy choice in choosing a packet's next hop. It selects the next hop closest to the destination, which is called *greedy forwarding*. The locations of neighbors are obtained by exchanging periodic HELLO messages among nodes. A HELLO message typically includes the sender's ID and the sender's location.

Greedy forwarding is simple and does not need a routing table. However, it may fall into the local minimum (a routing hole), where no neighbor nodes are closer to the destination than the node itself. Thus, it does not guarantee delivery. A simple example of such a topology is shown in Fig. 1 where S and D are the source and the destination, respectivly. Note that the dashed curve is the circle centered at D with radius $\overline{DS}$, inside which nodes are closer to D than S. The dotted curve is the radio range of S. We denote the intersection area of the two

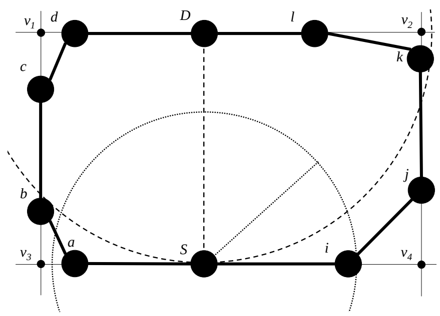


Fig. 1. A routing hole occurs at S.

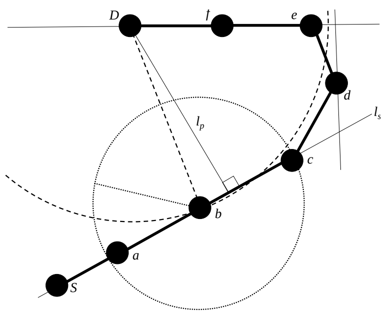


Fig. 2. A routing hole occurs at *b*.

circle $CandiArea_S^D$. Here, *S* is closer to *D* than its neighbors *a* and *i*. Although two paths ($S \rightarrow a \rightarrow b \rightarrow c \rightarrow d \rightarrow D$, $S \rightarrow i \rightarrow j \rightarrow k \rightarrow l \rightarrow D$) exist to *D*, *S* will not choose *a* or *b* to forward packets due to greedy forwarding.

To guarantee the delivery of packets in the presence of routing holes, face routing is proposed as a fallback solution [1] [2] [3]. It traverses faces to escape from the local minimum using the well-known right-hand rule. However, since face routing is performed blindly without considering actual shortest path, it may lead to much longer paths. Therefore, frequent routing holes degrade geographic routing performance as a side effect of face routing.

B. Effect of Node Placement on Routing Hole

If nodes are randomly distributed in a two-dimensional area, the probability that nodes exist in the $CandiArea$ is higher as the density of nodes increases. Accordingly, routing holes occur rarely and geographic routing is effective when the density of nodes is high enough [2] [4]. However, if nodes are located on specific lines such as cars on roads, trains on rails, and ships on courses, the occurrence of the routing holes is much more dependent on the layout of lines than the node density.

Let us illustrate how a routing hole happens in Fig. 2. First of all, there is a straight line (l_s) that represents a road or a course. From the perspective of D, we can draw a perpendicular line (l_p) to l_s. The intersection point of two lines is the nearest to D among all the points on l_s. As nodes are located on l_s, the nearer nodes is to the intersection point, the nearer is the nodes to the destination. Therefore, greedy forwarding is performed toward the intersection point.

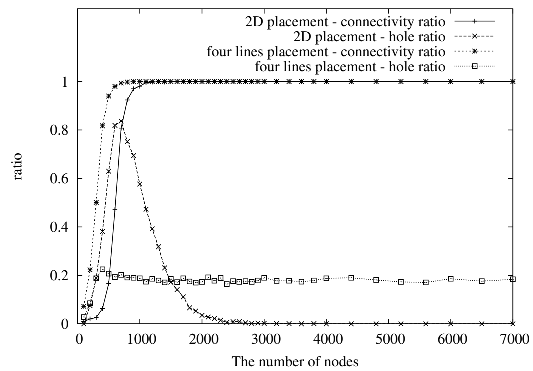


Fig. 3. The connectivity ratio and hole ratio with respect to the number of nodes.

Once a packet reaches the nearest node to the intersection point, greedy forwarding fails since there are no nodes in its $CandiArea$. In Fig. 2, the dashed curve is the circle centered at D with radius $\overline{Db}$ and the dotted curve is the radio range of b. S forwards to *a* and then *a* forwards to *b* greedily. Although a path ($b \rightarrow c \rightarrow d \rightarrow e \rightarrow f \rightarrow D$) exists to D, *b* cannot perform greedy forwarding due to no nodes within its $CandiArea$. In this situation, high node density does not help to solve the routing hole problem.

Fig. 3 shows routing hole and connectivity probabilities in two cases: (1) nodes are randomly located in a 2D area, and (2) nodes are located only on four lines. The radio range is one unit and the area size is 20 units x 20 units. In four lines placement, after two horizontal lines and two vertical lines are placed on the area randomly, nodes are located only on the lines randomly. A pair of a source and a destination is selected randomly 2000 times and the above performance metrics are averaged. The connectivity from a source to a destination increases in both of 2D area placement and four lines placement as the number nodes increases. Especially for the 2D area placement, the routing hole occurs rarely when the number of nodes is greater than 2500. However, in the four lines placement, the routing hole probability does not decline even when there are more than 2500 nodes. Consequently, this simulation reveals that existing geographic routing algorithms are not suitable for the case where nodes are placed only on lines because routing holes occur frequently even with the high node density.

III. Virtual Vertex Routing (VVR)

Existing geographic routing protocols experience routing holes frequently when nodes are placed only on lines. This is because they perform greedy forwarding blindly without considering the distribution of nodes.

We assume that all the courses are known to every node. Actually, this information is provided by the navigator system, digital road map or the map of courses which are usually embedded in vehicles. Moreover, the location of a destination is assumed to be known by a location service [11] [12]. This assumption is typical in geographic routing protocols.

Algorithm 1 Initialization

s_i: vertices of a source edge
d_j: vertices of a desination edge
S: a source node, *D*: a destinatin node

1: $floyd\text{-}table \Leftarrow Floyd(G)$
2: **if** a new packet is generated at the source **then**
3: $\quad (srcVtx, dstVtx) \Leftarrow \underset{\text{arg } s_i, d_j}{\min}\{dist(S, s_i)$
$\quad + floyd\text{-}path\text{-}dist(s_i, d_j) + dist(d_j, D)\}$
4: $\quad VVR.srcVtx \Leftarrow srcVtx$
5: $\quad VVR.nextVtx \Leftarrow srcVtx$
6: $\quad VVR.dstVtx \Leftarrow dstVtx$
7: $\quad VVR.locDst \Leftarrow the\ location\ of\ D$
8: $\quad$ greedyForwardingTo($VVR.nextVtx$)
9: **end if**

A. Model Description

The network is composed of vertices, edges and nodes. Vertices are geographical crosspoints between courses. Edges are segments of courses which are demarcated by two adjacent crosspoints. Nodes are real entities to communicate with each other. For example, in Fig. 1, v_1, v_2, v_3, and v_4 are vertices, $\overline{v_1v_2}$, $\overline{v_3v_4}$, $\overline{v_1v_3}$, and $\overline{v_2v_4}$ are edges, and *S*, *D*, *a*, *b*, *c*, *d*, *i*, *j*, *k*, and *l* are nodes. From the previously known course information, we can make the graph, $G = (V, E)$ where *V* is the set of vertices and *E* is the set of edges.

B. VVR Basic Mechanisms

1) Initialization: The pseudo code of this procedure is presented in Algorithm 1. The shortest paths of all pairs between vertices in graph G are calculated by the Floyd algorithm [13] whose complexity is $O(n^3)$ where n is the number of vertices (line 1). The overhead of this calculation is trivial because of two reasons; first, it is performed rarely only when the graph G is changed (e.g., new roads are added or existing roads are destroyed.) and second, the number of vertices is even smaller than that of nodes.

We define the edges on which the source node (S) and the destination node (D) are located as the source edge and the destination edge, respectively. Every edge has two vertices. Let the two vertices of the source edge be s_1 and s_2. Let the two vertices of the destination edge be d_1 and d_2. Recall that the source knows the location of the destination. Then, S chooses the source vertex ($srvVtx$) from s_1 or s_2, and the destination vertex ($dstVtx$) form d_1 or d_2 by evaluating the following equation in the four cases

$$\min\{dist(S, s_i) + floyd\text{-}path\text{-}dist(s_i, d_j) + dist(d_j, D)\},$$
$$i, j = 1\ or\ 2$$

where $dist(a, b)$ is the Euclidian distance between a and b and $floyd\text{-}path\text{-}dist(x, y)$ is the distance of the shortest path, calculated by Floyd algorithm, between $vertex\ x$ and $vertex\ y$ (line 3). To substantiate the VVR protocol, the header of each packet includes the following fields: $VVR.srcVtx$, $VVR.nextVtx$, $VVR.dstVtx$, and $VVR.locDst$, which are set to $srcVtx$, $srcVtx$, $dstVtx$, and the location of D, respectively (lines 4-7). Lastly, S performs greedy forwarding to $VVR.nextVtx$ (line 8).

Algorithm 2 Vertex Change

$lookup\text{-}floyd\text{-}table(v_1, v_2)$: return the next vertex of the shortest path between v_1 and v_2 by $floyd\text{-}table$ look-up

1: **if** a packet is relayed from the previous hop node **then**
2: $\quad$ **if** in the proximity of $VVR.nextVtx$ **then**
3: $\quad\quad$ **if** $VVR.nextVtx \neq VVR.dstVtx$ **then**
4: $\quad\quad\quad VVR.nextVtx \Leftarrow$
$\quad\quad\quad lookup\text{-}floyd\text{-}table(VVR.nextVtx, VVR.dstVtx)$
5: $\quad\quad$ **else**
6: $\quad\quad\quad VVR.nextVtx \Leftarrow null$
7: $\quad\quad$ **end if**
8: $\quad$ **end if**
9: **end if**
10: **if** $VVR.nextVtx \neq null$ **then**
11: $\quad$ greedyForwardingTo($VVR.nextVtx$)
12: **else**
13: $\quad$ greedyForwardingTo($VVR.locDst$)
14: **end if**

2) Vertex Change: VVR forwards packets vertex-by-vertex. Once a packet arrives at a vertex, its intermediate destination is updated. Note that there is no designated physical node corresponding to the vertex. One of the nodes that are in the proximity of the geographical location of the vertex will serve as the vertex virtually. The pseudo code of this procedure is presented in Algorithm 2. When the packet reaches a node in the proximity of $VVR.nextVtx$ (line 2), the forwarding node (virtual vertex) changes $VVR.nextVtx$ to the next vertex of the shortest path toward $VVR.dstVtx$ (line 4). Note that the shortest path has been already calculated by the Floyd algorithm. After arriving in the proximity of the last vertex ($VVR.dstVtx$), the packet is forwarded to D (line 13).

We define the proximity of a vertex as the area within the circle whose center is the vertex and whose radius is the half of the radio range. The rationale behind the proximity size is as follows. In the proximity of the vertex, the forwarding node should be able to transmit the packet to its next hop which is located on the different edge. By limiting the proximity to the half of the radio range, the distance between any two nodes in the proximity is less than or equal to the radio range.

Let us illustrate the forwarding of VVR. In Fig. 1, a routing hole occurs at S using existing geographic routing protocols. However, VVR forwards the packet successfully without a routing hole. In VVR, S sets the $VVR.srcVtx$ to v_3, $VVR.nextVtx$ to v_3, and $VVR.dstVtx$ to v_1, and greedily forwards to $VVR.nextVtx$ (v_3). When a in the proximity of v_3 receives a packet, it changes $VVR.nextVtx$ to v_1, and performs greedy forwarding to $VVR.nextVtx$ (v_1). The packet reaches c in the proximity of $VVR.dstVtx$ (v_1) and then c changes the VVR.nextVtx to $null$. After then, the

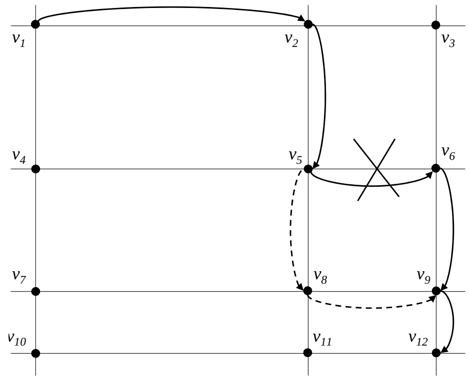


Fig. 4. 'X' represents the disconnect of the edge. Nodes are not shown for the simplicity of explanation of VVR-GR and VVR-FR.

packet is greedily forwarded to D (the destination node).

C. VVR Advanced Mechanisms

The success of the packet delivery through edges is not guaranteed because if the distance between the adjacent nodes on the edge is longer than the radio range, the edge is actually disconnected. If the edge on the shortest path is disconnected, a routing hole occurs. To tackle this routing hole, we propose two schemes, VVR-Greedy Routing (VVR-GR), VVR-Face Routing (VVR-FR). Both schemes can be used together.

1) Edge Connectivity: To know the connectivity of all edges requires the location of all nodes in the network. Using this global information is not appropriate for large scale distributed systems due to the considerable overhead. Therefore, nodes on an edge maintain the minimum connectivity information, i.e., connectivity to the edge's two endpoints (two vertices). To do so, a node sends modified HELLO messages periodically and maintains a *Connectivity Table* whose record has two fields - <vertex v, next hop list l> where v is reached through one of nodes in l. When a node sends a HELLO it checks connectivity of two vertices on its edge from the *Connectivity Table*. If connected, it adds the vertices in a HELLO message. A node in the proximity of a vertex adds the vertex in a HELLO message. When a node receives a HELLO, it updates the *Connectivity Table*. The next hop list l contains all the neighbors in the radio range toward the vertex. When a node looses its connectivity to its neighbor node, the neighbor is removed from the next hop list in all records of the *Connectivity Table*.

2) VVR-Greedy Routing(VVR-GR): When a packet arrives at a node in the proximity of a vertex, the node selects a new vertex to the destination. If the edge to the new vertex is disconnected, the forwarding node appends the current vertex to $VVR.holeList$. Then, it changes $VVR.nextVtx$ to the vertex (v_{new}) among the connected neighbor vertex such that the shortest path between v_{new} and $VVR.dstVtx$ does not include any vertex in $VVR.holeList$ and the cost of the path, $dist(VVR.nextVtx, v_{new})$ + $floyd\text{-}path\text{-}dist(v_{new}, VVR.dstVtx)$, is minimum. A simple example is shown in Fig. 4. In this scenario, $VVR.srcVtx$ and $VVR.dstVtx$ are v_1 and v_{12}, respectively. Between the two vertices, the shortest path is determined as $v_1 \rightarrow v_2 \rightarrow v_5 \rightarrow v_6 \rightarrow v_9 \rightarrow v_{12}$. In the proximity of v_5, the edge between v_5 and v_6 is disconnected. The node in the proximity of v_5 appends v_5 to $VVR.holeList$ and then checks the candidate next vertex among the connected neighbor vertices (v_2, v_4, and v_8). The shortest path between v_2 and v_{12} includes v_5 which is in $VVR.holeList$. Thus, v_2 cannot be selected as the $VVR.nextVtx$. When we compare v_4 and v_8, $dist(v_5, v_8) + floyd\text{-}path\text{-}dist(v_8, v_{12})$ is smaller than $dist(v_5, v_4) + floyd\text{-}path\text{-}dist(v_4, v_{12})$. Therefore, $VVR.nextVtx$ is set to v_8.

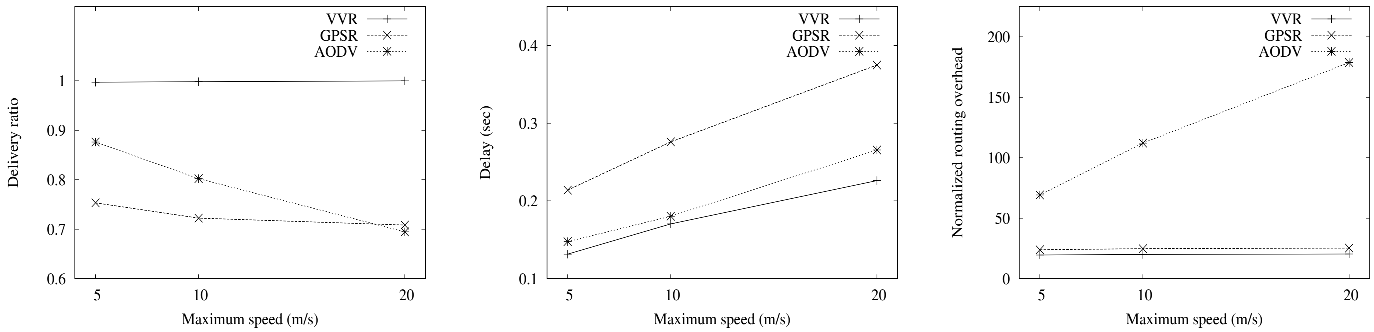
3) VVR-Face Routing(VVR-FR): The overhead of VVR-GR is trivial because it can be done by $floyd\text{-}table$ lookup and needs no extra control packets. However, VVR-GR does not always guarantee the delivery of packets. For example, if in the proximity of v_5 in Fig. 4, the edges to v_4, v_6 and v_8 are disconnected, VVR-GR fails although many paths exist between v_1 and v_{12}. To address this problem, we borrow the idea of face routing [1] [2] [3].

Face routing chooses its next hop using the well-known right hand rule (an intermediate node always chooses first edge in counter clockwise) for traversing a graph and it works on planarized graphs which have no crossing edges. GPSR [1] [2] performs face routing hop by hop when a routing hole occurs. Note that in GPSR vertices are nodes and edges are the wireless links between neighbor nodes. GPSR should keep the planarized graph in a distributed manner; this is not a trivial overhead because edges are frequently changed due to mobility of nodes.

Contrary to GPSR, VVR-FR performs face routing vertex by vertex. Note that graph G is a planarized graph already, which means VVR-FR has no overhead to maintain the planarized graph. If VVR-GR fails, the forwarding node seeks the next vertex among the connected neighbors by the right-hand rule and sets $VVR.nextVtx$ to the chosen vertex and $VVR.frVtx$ to the current vertex. Then, it greedily forwards to $VVR.nextVtx$. This procedure is repeated only until the packet escapes from the local minimum, i.e., the packet reaches the vertex closer to $VVR.dstVtx$ than $VVR.frVtx$. From then on the routing returns to VVR from VVR-FR. With the help of face routing, VVR-FR can guarantee the delivery of packets if there are paths.

In the above failure scenario of VVR-GR, the forwarding node in the proximity of v_5 performs VVR-FR; sets $VVR.nextVtx$ to v_2 which is the vertex of the first edge in counter clockwise, and $VVR.frVtx$ to v_5. $VVR.nextVtx$ is updated to v_3 and then v_6 sequentially by VVR-FR. When the packet reaches v_6 the routing returns to VVR from VVR-FR since the packet escapes from the local minimum. Then VVR forwards the packet to v_{12} through v_9.

4) Inconsistency of Connectivity Information: The inconsistency of connectivity information may occur due to the delay of propagation. To mitigate this inconsistency, we use triggered HELLO, i.e., if a new vertex is added or an old vertex is removed in the *Connectivity Table*, a HELLO message is immediately sent. In spite of using a triggered HELLO, inconsistency still occurs because a HELLO message is not



(a) Delivery ratio (b) Delay (c) Normalized routing overhead

Fig. 5. Performance comparisons between VVR, GPSR, and AODV with respect to mobility of nodes.

reliable due to the wireless channel error. Therefore, in the middle of packet relay over an edge, the packet can be stuck. To handle this problem, the VVR header contains the previous vertex ($VVR.prevVtx$) which is set to the current vertex when $VVR.nextVtx$ is changed. When a packet is stuck at a node in the middle of an edge, the node forwards it to $VVR.prevVtx$. In the proximity of VVR.prevVtx, VVR-GR or VVR-FR is performed.

IV. Simulation Results

In this section, we evaluate the performance of VVR, which includes both basic and advanced mechanisms, using NS-2 [16]. We compare the performance of VVR with GPSR [1] [2] and AODV [5]. GPSR and AODV are the representative geographic and reactive routing protocols, respectively. We exclude the proactive routing protocols for comparison due to large delay of routing table convergence [14], which is not suitable for high mobility environments in VANETs.

A. Simulation Setup

IEEE 802.11 [15] is used for the network interface. The bandwidth is 1 Mbps and the radio range is 250 m. In a 3000 m x 1500 m area, two horizontal lines and four vertical lines are placed at random. In addition, the boundaries of the area are also used as lines. Totally, there are four horizontal lines and six vertical lines in the area. 420 nodes are distributed on the lines and move along the lines. The speed and the direction of nodes are selected randomly at each vertex. The speed is chosen between zero and the maximum speed (5, 10, 20 m/s). For performance evaluation, 20 pairs of sources and destinations are selected randomly and the sources send 1000 bytes packets over UDP with the interval of one second. The interval of HELLO messages is set to one second in both VVR and GPSR. The simulations are performed five times for 300 seconds.

We use three metrics: packet success delivery ratio, end-to-end packet delay, and normalized routing overhead. The performance metrics are averaged over five runs.

B. Packet Success Delivery Ratio

Fig. 5(a) shows packet delivery ratio with respect to the maximum speed. Only packets for which a path exist to the destination are included for the comparison. The delivery ratio of GPSR is lower than AODV in the low and medium mobility (5, 10 m/s). The reason is as follows. Routing holes occur frequently when nodes are placed only on lines as described in Section II. If routing holes occur, GPSR's face routing is used, which needs a planarized graph. When nodes have mobility, the planarized graph may become inconsistent with the current network status. This inconsistency may cause routing loops. Furthermore, GPSR's face routing make paths longer, which causes drop of packets due to TTL expiration. Due to reactiveness of AODV, AODV rediscoveries the path after routing fails when a link breakage occurs. Therefore, the delivery ratio of AODV decrease as the mobility is high. On the other hand, VVR updates neighbor's location with periodic HELLO messages and falls into routing holes rarely due to the knowledge of distribution of nodes. As a result, VVR outperforms others as shown in Fig. 5(a).

C. End-to-End Packet Delay

The end-to-end packet delay is affected by the delivery ratio, and some extreme long-delay packets may greatly increase the mean value. To better study the end-to-end packet delay, we examine *the lowest 90 % delivery delay*, which is the average delay of the lowest 90 % packets.

In general, geographic routing protocols have lower delay than reactive routing protocols. However, if routing holes occur frequently, the delay of geographic routing protocols can be lengthened because face routing make paths longer. Fig. 5(b) shows that the delay of AODV is larger than that of VVR and smaller than that of GPSR. This is because GPSR experiences routing holes much more frequently compared to VVR. Thus, GPSR is not appropriate for delay-sensitive applications in VANETs.

D. Normalized Routing Overhead

The number of total control packets (e.g., HELLO, route discovery) sent by each node is normalized by the number of the successfully delivered data packets. Fig. 5(c) shows the normalized routing overhead with respect to the maximum speed. Since the route discovery of AODV normally causes network-wide flooding, AODV has large routing overhead. Especially, as node mobility increases, the routing overhead increases accordingly because link breakages cause route rediscoveries. Thus, AODV is not a good option for high mobility environments. On the other hand, geographic routing protocols send only periodical HELLO messages. Thus, routing overhead is constant and low irrespective of node mobility.

V. Conclusion

In this paper, we show that existing geographic routing protocols are not suitable for VANETs due to frequent routing holes. To tackle this problem, we propose Virtual Vertex Routing (VVR) using the information of roads, rails, or courses. This kind of information is provided by navigator systems embedded in vehicles. Using graph formulation, we introduce a new concept, the proximity of a vertex (or a virtual vertex). An intermediate node in this proximity performs routing toward the destination by Floyd algorithm. For routing holes, we propose two countermeasures: greedy routing (VVR-GR) and face routing (VVR-FR). The former can reduce the recovery time of routing holes and the latter can guarantee the packet delivery. Through extensive simulations, we show that VVR outperforms GPSR and AODV.

Acknowledgment

This work was supported in part by the Brain Korea 21 project of the Ministry of Education Korea, and in part by the Korea Science and Engineering Foundation (KOSEF) grant funded by the Korea government (MOST) (No. F01-2005-000-10040-0). The ICT at Seoul National University provides research facilities for this study.

References

[1] Brad Karp and H. T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” *Proc. ACM MobiCom ’00*, 2000.

[2] Brad Karp, “Geogrpahic Routing for Wireless Networks,” PhD thesis, Harvard University, 2000.

[3] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia, “Routing with Guaranteed Delivery in Ad Hoc Wireless Networks,” *Proc. ACM DialM ’99*, 1999.

[4] Fabian Kuhn, Roger Wattenhofer, and Aaron Zollinger, “Worst-Case Optimal and Average-Case Efficient Geometric Ad-Hoc Routing,” *Proc. ACM MobiHoc ’03*, 2003.

[5] C. Perkins, E. Belding-Royer, and S. Das, “Ad hoc On-demand Distance Vector (AODV) Routing,” *IETF Experiment RFC*, MANET working group, RFC 351, July 2003.

[6] D. Johnson and D. Maltz, “Dynamic Source Routing in Ad Hoc Wireless Netowrks,” In *Mobile Computing*, T. Imielinski and H. Korth, Eds, Ch. 5, pp. 153-181, Kluwer, 1996.

[7] Vincent D. Park and M. Scott Corson, “Temporally-Ordered Routing Algorithm (TORA),” *Proc. IEEE INFOCOM ’97*, 1997.

[8] C. Perkins and P. Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV),” *Proc. ACM SIGCOMM ’94*, 1994.

[9] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, “Optimized Link State Routing Protocol for Ad Hoc Networks,” *Proc. IEEE INMIC ’01*, 2001.

[10] X. Hong, K. Xu, and M. Gerla, “Scalable Routing Protocols for Mobile Ad Hoc Networks,” *IEEE Network*, vo. 16, no. 4, July-Aug 2002.

[11] J. Li, J. Jannotti, D. DeCouto, D. Karger, and D. Johnson, “A Scalable Location Service for Geographic Ad-hoc Routing,” *Proc. ACM MobiCom ’00*, 2000.

[12] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker, “GHT: A Geographic Hash Table for Data-Centric Storage in Sensornets,” *Proc. ACM WSNA ’02*, 2002.

[13] Robert W. Floyd, “ACM Algorithm 97: Shortes Path,” *Communications of the ACM*, vol.5, no. 6, June 1962.

[14] J. Broch, D. Maltz, D. Johnson, Y.-C. Hu, and J. Jetcheva, “A Performance Comparison of Multi-Hop Wireless Ad Hoc Network Routing Protocols,” *Proc. ACM MobiCom ’98*, 1998.

[15] “IEEE 802.11, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications,” IEEE 802.11 Standard, 1999.

[16] “The Network Simulator - NS-2,” http://www.isi.edu/nsnam/ns/, Online link.