



저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

2010년 8월

석사학위논문

Traffic Control Aware Routing for Vehicular Sensor Networks

소

신

2010년 8월
석사학위논문

Traffic Control Aware Routing for Vehicular Sensor Networks

조선대학교 대학원

컴퓨터공학 전공

소 신

차량 센서 네트워크를 위한 교통 관제 인지 기반 라우팅

Traffic Control Aware Routing for Vehicular
Sensor Networks

2010년 8월 25일

조선대학교 대학원

컴퓨터공학 전공

소 신

차량 센서 네트워크를 위한 교통 관제 인지 기반 라우팅

지도교수 정 일 용

이 논문을 컴퓨터공학 석사학위신청 논문으로 제출함.


2010년 4월


조선대학교 대학원


컴퓨터공학 전공

소 신

소 신의 석사학위 논문을 인준함.

위원장 조선대학교 교수 朴相晩 

위 원 조선대학교 교수 姜文秀 

위 원 조선대학교 교수 丁日鎔 

2010년 5월

조선대학교 대학원

Contents

Contents	i
List of Figures	iii
List of Tables	iv
ABSTRACT	v
1. Introduction	1
2. Background and Related Work	4
2.1 Background of VSNs	4
2.2 Geographic Forwarding Routing	5
3. System Model	8
3.1 Network Model	8
3.2 Mobility Model	9
4. Traffic Control Aware Routing	12
4.1 Vertices Selection	12
4.2 Packet Forwarding Between Two Vertices	17
5. Performance Evaluation	20
5.1 Simulation Environment	20
5.2 Simulation Results and Discussion	23

5.2.1 Packet Delivery Ratio	23
5.2.2 Average End-to-End Delay	25
5.2.3 Normalized Routing Overhead	27
6. More Discussion	29
7. Conclusions	31
References	32

List of Figures

Figure 1. An application example of VSNs	2
Figure 2. An example of VSNs in the city environment	9
Figure 3. An example of a routing path in GSR	13
Figure 4. An example of traffic status	15
Figure 5. An example of greedy forwarding	19
Figure 6. An example of the distribution of P_{st}	22
Figure 7. Simulation results of packet delivery ratio	24
Figure 8. Simulation results of average end-to-end delay	26
Figure 9. Simulation results of normalized routing overhead	28
Figure 10. An example intersection with more than four entrances	30

List of Tables

Table 1. Parameters used in the simulation	22
--	----

초 록

차량 센서 네트워크를 위한 교통 관제 인지 기반 라우팅

소신

지도교수 : 정일용

조선대학교 대학원 컴퓨터공학과

최근 새로운 무선 센서 네트워크 패러다임으로서 차량 센서 네트워크(VSN)가 운전 편의성 및 교통 제어를 혁신할 수 있는 비전으로 부상하고 있다. 기존 센서 네트워크와 같이 VSN은 이벤트를 감지하고 전송, 처리하는 기능을 수행한다. 일반적으로 센서들은 택시나 버스 등과 같은 차량에 장착되어 있어 빠르게 움직인다. 따라서 기존의 무선 센서 네트워크용 라우팅 프로토콜이 부적합한 경우가 많아 더 많은 주의를 기울여 라우팅 프로토콜을 설계해야 한다. 본 논문에서는 차량 센서 네트워크를 위한 교통 관제 인지 기반 적응형 라우팅 프로토콜(PUT)을 제안한다. 제안한 프로토콜은 크게 두 모듈로 구성된다. 첫째는 패킷을 목적지로 전달하는 일련의 교차로들을 교통 관제 기반으로 선택하는 것이고, 둘째는 두 인접 교차로 사이에서 패킷을 전달하는 그리디 포워딩(greedy forwarding) 기법이다. 시뮬레이션 결과에 의하면, 제안한 라우팅 프로토콜은 패킷 전달률, 종단간 지연시간, 라우팅 오버헤드 측면에서 종래의 프로토콜보다 성능이 우수하다.

ABSTRACT

Traffic Control Aware Routing for Vehicular Sensor Networks

Xin Su

Advisor : Prof. Il-Yong Chung, Ph.D.

Department of Computer Engineering

Graduate School of Chosun University

Recently, *vehicular sensor networks (VSNs)* have emerged as a new wireless sensor network paradigm that is envisioned to revolutionize driving experiences and traffic control systems. Like conventional sensor networks, they can sense events and process sensed data. In general, sensors are moving fast because they are equipped on vehicles such as taxis and buses. Thus, the design of a routing protocol needs more attention when the routing protocols developed for conventional *wireless sensor networks (WSNs)* become unfeasible. Many existing routing protocols for VSNs have good performance on data routing in a city environment. In this thesis, I propose an adaptive routing protocol associated with urban traffic control mechanism for VSNs, which is called PUT. It considers two modules of (i) the traffic control aware selection of vertices through which a packet is passed toward its destination and (ii) the greedy forwarding strategy by which a packet is forwarded between two adjacent vertices. The simulation results show that the proposed PUT outperforms conventional protocols in terms of packet delivery ratio, end-to-end delay and routing overhead.

1. Introduction

The vehicular ad hoc networks (VANETs) have been obtaining commercial relevance because of recent advances in inter-vehicle communications and decreasing costs of related equipments. This situation stimulates a brand new group of visionary services for vehicles, i.e., from entertainment applications to advertising information, and from driver safety to opportunistic transient connectivity to the fixed Internet infrastructure [1]-[4]. Particularly, a VSN [5], a sort of VANETs, is emerging as anew infrastructure for monitoring the physical world, especially in urban areas where a high concentration of vehicles equipped with onboard sensors is expected. VSNs inherently provide a perfect way to collect dynamic interest of information, and sense various physical quantities with very low cost and high accuracy. Fig. 1 is an application example of VSNs. The attached vehicular sensors capture the urban physical quantities, e.g. urban temperature, and then forward this information to the nearest base station. Then, the base station can send the information to the application server by using one or more wired networks.

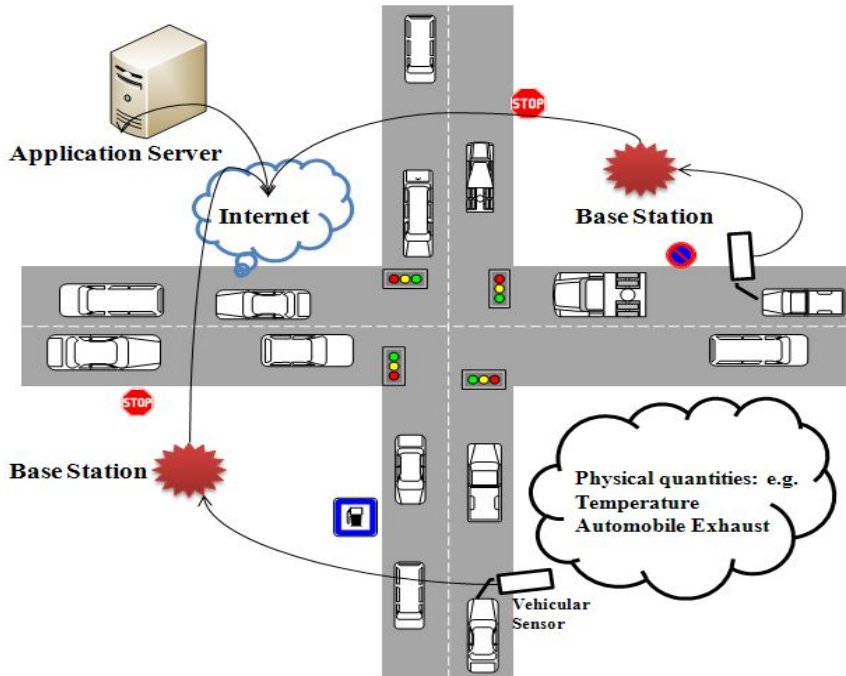


Fig. 1: An application example of VSNs.

VSNs represent a novel and challenging deployment scenario, which are considerably different from conventional WSNs [6]-[7], thus requiring innovative solutions on routing. In fact, vehicular sensors are not affected by strict energy constraints and storage capabilities because they can be equipped with powerful processing units and wireless transmitters in a vehicle. Consequently, energy dissipation and data storage space are not considered often as design issues of routing protocols in VSNs. The major routing issue considered in VSNs is the rapidly changing network topology. This is because wireless communication is unreliable in vehicle-to-vehicle (V2V) communication due to multi-path fading, shadowing, and Doppler effects caused by the high mobility of vehicles. Such effects make routing

protocols quite complicated.

Conventional routing protocols for VSNs, such as GSR [15], VVR [17] and GyTAR [20], have fairly good performance. However, they did not take the condition of traffic control mechanism into consideration, which is one of the most important constraints affecting routing performance. In this paper, I aim to design a routing protocol based on geographic forwarding for VSNs, which is associated with the traffic control mechanism. The vertices selection and the greedy forwarding between neighboring vertices are based on the current traffic situation. Several assumptions have been made in the paper as follows:

- Vehicles participating in a VSN can know their own position, speed, moving direction and acceleration/deceleration correctly by using the electronic control unit and navigation system.
- Vehicles are equipped with identical pre-loaded urban digital maps.
- Vehicular sensors have plentiful space for storage and power supply. The energy dissipation and storage usage are not taken into account in this paper.
- Vehicular sensors sense and recognize the physical quantities correctly.

The remainder of this thesis is organized as follows. Section 2 overviews the background and routing strategies based on geographic forwarding for urban wireless vehicular networks. The system model of network and mobility is outlined in Section 3. In Section 4, the proposed PUT is presented. Section 5 discusses the performance evaluation of PUT. The simulation environment and results are presented in detail. More discussion will be presented in Section 6. Finally, the thesis is concluded in Section 7.

2. Background and Related Work

2.1 Background of VSNs

The idea of embedding sensors in vehicles is very novel. An example project is the Massachusetts Institute of Technology's CarTel [8], [9] that deals with the MobEyes-addressed [5] challenging issues of car-based sensing and distributed opportunistic search of sensed data. In CarTel, users submit their queries about sensed data on a portal that is hosted on the wired Internet. Then, an intermittently connected database is in charge of dispatching "queries" to vehicles and of receiving replies when vehicles move near open access points to the Internet. Other interesting research projects have focused on providing mobile Internet access to vehicles. For instance, InternetCar aims at providing vehicles with seamless Internet connectivity by envisioning various applications such as a traffic information dissemination service where "raw" sensed data from vehicles are collected in a central server, and traffic information is distributed to the drivers [10], [11]. Unlike CarTel and InternetCar, MobEyes exploits mobile collector agents instead of relying on the wired Internet infrastructure thus improving robustness. In addition, note that the FleetNet project, which aims at developing an inter-vehicle communication platform for vehicular applications, recognized the potential for services that distribute location-tagged information (e.g., traffic jam warning) by collecting and processing data from cars in a distributed fashion [12], [13]. MobEyes support proactive urban monitoring by exploiting vehicle mobility to opportunistically diffuse summaries about sensed data. It can harvest summaries and build a low cost distributed index with reasonable completeness, good scalability and limited overhead.

2.2 Geographic Forwarding Routing

The existing routing protocols for VSNs are basically based on the so-called geographic forwarding. The geographic position of nodes is necessary to forward the packet in a greedy way to the neighbor which is geographically closest to the destination. If the node does not find a neighbor closer to the destination than itself within its radio range, the greedy algorithm may fail.

Urban Multi-hop Broadcast Protocol (UMB) [14] is designed to overcome interference, packet collision, and hidden node problem during message dissemination in multi-hop forwarding. The sender nodes try to select the furthest node in the direction to assign the duty of forwarding and acknowledging the packet without any a prior topology information. At the intersection, repeaters [14] are installed to forward the packet to all road segments.

Geographic Source Routing (GSR) [15] uses reactive location service (RLS) to know the current position of the desired communication partner. When the querying node requires position information of neighboring nodes, it floods the "position request" containing its ID to the network in a reactive way. When the corresponding node receives the request, it sends "position reply" to the querying node. With the position information of neighbor nodes, the sender node computes a sequence of junctions, through which a packet has to traverse to reach its destination using a city map. Note that the sequence of junctions can be either contained in the packet header or computed by each forwarding node. Forwarding a packet to successive junctions is done on the basis of greedy forwarding and using the Dijkstra's shortest path algorithm [16], and the distance from source to destination can be calculated based on the city map. When a route break occurs, GSR uses the recovery strategy "fall back on greedy mode" to bypass the particular node.

Virtual Vertex Routing (VVR) [17] uses the line information (i.e. roads, rails and

courses) of each vehicle, which is provided by a navigation system or digital road map equipped in vehicles. It forwards packets in greedy way and solves the so-called routing hole problem. If the node density is high, routing holes occur rarely and geographic routing is effective [18]. However, it is claimed [17] that the node density is highly dependent on the layout of lines. So, high node density does not help to solve the routing hole problem if all the vehicles lie on a specific line. VVR represents the network as a graph and uses the concept of virtual vertex (i.e. the adjacent crossing point of two vertices). The intermediate nodes in the proximity of vertex perform routing towards the destination using the Floyd algorithm [19]. To tackle the routing hole problem, VVR greedy routing (VVR-GR) and VVR face routing (VVR-FR) were proposed as well [17]. VVR-GR reduces the recovery time of routing holes and VVR-FR can guarantee the delivery of packets.

Improved Greedy Traffic Aware Routing Protocol (GyTAR) [20] is an improved greedy traffic aware, intersection-based geographic routing protocol which uses real-time traffic density information and movement prediction to route packets. It consists of two modules of (i) selection of junctions through which a packet must pass to reach its destination and (ii) an improved greedy forwarding mechanism between two junctions [20]. When a vehicle receives a packet, it computes its next junction with the highest score by considering traffic density and curve-metric distance to the destination. The junction with the highest score is geographically closest to the destination vehicle and has the highest vehicular traffic. Between two adjacent junctions, the packets are forwarded through the vehicles on between the successive junctions by using improved greedy forwarding. Each vehicle maintains a table containing position, velocity and direction of each neighboring vehicle, and the table is updated by periodically exchanging HELLO messages among vehicles. Using the information in the table, forwarding vehicles select their next hop neighbor which is closest to the

destination junction.

Greedy Perimeter Coordinator Routing (GPCR) [21] utilizes the fact that the vehicular sensors at a junction in the street follow a nature planar graph. Thus, a restricted greedy algorithm can be followed as long as the sensors are on a street. Junctions are the only places where actual routing decisions are taken. Therefore, packets should always be forwarded to a node on a junction rather than being forwarded across the junction.

The above-mentioned routing protocols are not designed with the awareness of the urban traffic control mechanism. Moreover, the vehicle acceleration and deceleration are not considered a condition for choosing forwarding neighbors. This motivated our work.

3. System Model

In this section, a network model is presented first followed by a mobility model.

3.1 Network Model

In this paper, it is assumed that all vehicles communicate with each other by using IEEE 802.11 standard. In city environments, high-rise buildings are the radio obstacles. In Fig. 2, vehicle B is within the communication range of vehicle A. Vehicle A forwards a packet to vehicle B, but vehicle B cannot receive the packet from vehicle A because of radio obstacles. In such an area, while greedy forwarding is used to forward a packet to its neighbor, source node (node and vehicle are used interchangeably) chooses a neighbor which is closest to the destination node within its communication range but the transmitted packet may be lost due to radio obstacles. Consequently, the packet is retransmitted, and consumes unnecessary channel bandwidth. So, in urban environments, the prediction of the future network state is important, and prediction-based routing should be a promising approach. Alternatively, prediction of vehicles near the vertex (intersection) plays a vital role in VSNs within the urban scenario.

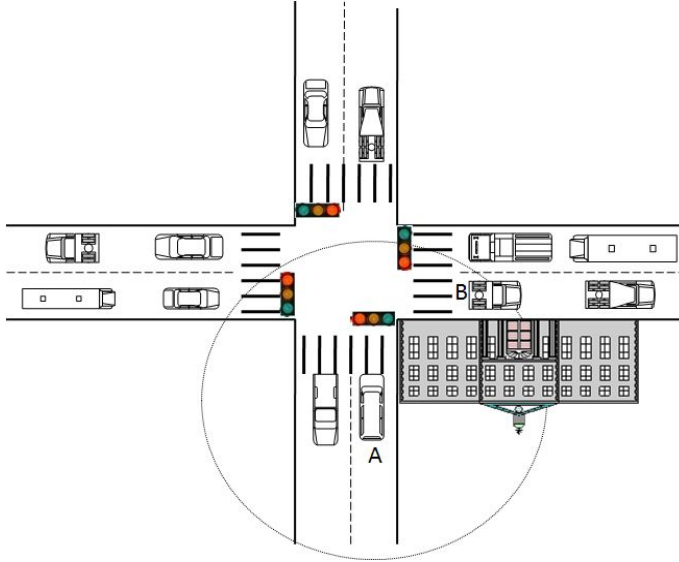


Fig. 2: An example of VSNs in the city environment.

3.2 Mobility Model

In VSNs, the mobility pattern of vehicles influences the route discovery, maintenance, reconstruction, and accuracy. I illustrate the three key factors of street layout, obstacles, and traffic control mechanism that affect the mobility of vehicles as follows:

Street layouts: Streets force vehicles to confine their movement to well-defined paths. This constrained movement pattern determines the spatial distribution of vehicles and their connectivity. Streets can have either single or multiple lanes and can allow either one-way or two-way traffic. The movement of every vehicle is influenced by the movement pattern of its surrounding vehicles. For example, a vehicle would try to maintain a minimum distance from the one in front of it. It may increase or decrease its speed, or may change to another lane.

Obstacles: Obstacles, such as buildings, determine the number of intersections in the area,

which in turn determines the frequency of vehicle stops. They also determine whether vehicles at the neighboring intersections can sense each other's radio transmissions. Larger obstacles make the network more sensitive to clustering and degrade performance.

Traffic control mechanisms: The most common traffic control mechanisms are traffic lights and stop signs. These mechanisms result in the formation of clusters and queues of vehicles and subsequent reduction of their average speed. In this paper, I assume every street intersection has a traffic light. If vehicles following each other move to an intersection with a red light, the vehicles form a queue at the intersection. Each vehicle waits for at least the required time once it gets to the head of the intersection after other vehicles ahead in the queue leave. The traffic light gives the vehicles a probability, denoted as P_{inter} , to stop at the intersection when the vehicles reach it with an empty queue. With the probability $1 - P_{inter}$, the vehicles can directly cross the intersection without stopping. In PUT, for every street intersection, I use a unique P_{inter} for vehicles stopping at intersections with an empty queue because traffic lights are altered periodically after the implementation of the system. At the same time, I set a stop sign in the middle of the each street segment. If a vehicle moves to a stop sign with an empty queued vehicles line, it stops at the stop sign with a probability P_{st} . With the probability $1 - P_{st}$, the vehicle can pass by the stop sign immediately. The value of P_{st} for different street segments, however, varies because it is determined by the roadside buildings, such as schools, hospitals and restaurants. Obviously, a vehicle moving on a street with more roadside objects has a higher value of P_{st} . I assume that the vehicles which move on the same street segment can share a unique P_{st} . However, the value of P_{st} varies between street segments (An example of the distribution of P_{st} will be given in Section 5). Furthermore, if a vehicle decides to wait in an empty queue, the amount of waiting time is randomly chosen between 0 and T seconds. Any vehicle that arrives later at a non-empty queue will have to wait for the remaining wait time of the previous vehicle plus one second. The additional one second simulates the startup delay between queued vehicles. Whenever the traffic light or stop sign turns green,

the vehicles begin to cross the signal at intervals of one second, until the queue is empty. The next vehicle that arrives at the head of an empty queue again makes a decision on whether to stop with a probability P_{inter} or P_{st} and so on.

In the system, Manhattan mobility [22] is used for vehicles which move in a grid road topology mainly proposed for movement in an urban area, where the streets are organized in a regular grid. In this mobility model, the mobile vehicles move in horizontal or vertical directions on an urban map. The Manhattan model employs a probability approach in the selection of vehicles movements. At each intersection, the probability of taking a left turn is 0.5 and a right turn is 0.25 in each case. Thus, 0.25 is used for vehicles moving straight ahead.

4. Traffic Control Aware Routing

Most conventional V2V routing protocols [14, 15, 17, 20, 21] do not consider the factors affecting the vehicles' mobility. However, the mobility models determine the location of nodes in the topology at any given time interval and they strongly affect network connectivity and throughput. In this paper, I aim to design a new V2V routing protocol for VSNs associated with urban traffic control mechanism, which inevitably impact on vehicles' mobility. The proposed PUT is divided into two phases: (i) vertices selection and (ii) packet forwarding between two adjacent vertices. They are detailed as follows.

4.1 Vertices Selection

As a strategy to deal with the high mobility of nodes on one hand and with the specific topological structure of a city on the other hand, I have chosen a position-based routing protocol, which is supported by a digital map of the city. The presence of a digital map is a valid assumption when vehicles are equipped with on-board navigation systems. Thus, each vehicle is aware of its geographic position, and knows the position of neighbors by sensing beacon messages that are periodically exchanged by vehicles and roadside infrastructure. I also assume that every vehicle is aware of the current traffic status. This information can be provided either through a simple distributed mechanism for on-road traffic estimation realized by all vehicles or by traffic sensors installed beside the streets.

In GSR [15], the packet sending node can compute a path to the destination by using

the navigation system. This path can be abstracted as a directed graph $P(V,E)$ where V is the set of vertices and E is the set of edges. The sequence of vertices can be put into the packet header, and forwarding the packet between two successive vertices is done on the basis of greedy forwarding. The path between source and destination in GSR is determined by the Dijkstra shortest path calculation based on the street map. In Fig. 3, upon sensing any event on the road, the sender, vehicle A, communicates with the nearest base station (sink) in an ad hoc manner among local vehicles. An example of the shortest path determined by the Dijkstra algorithm is: vehicle $A-V1-V2-V4-V6-BS$.

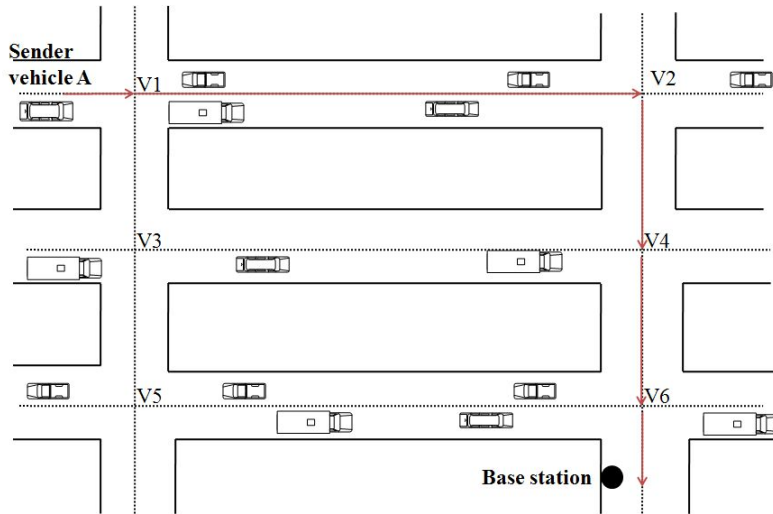
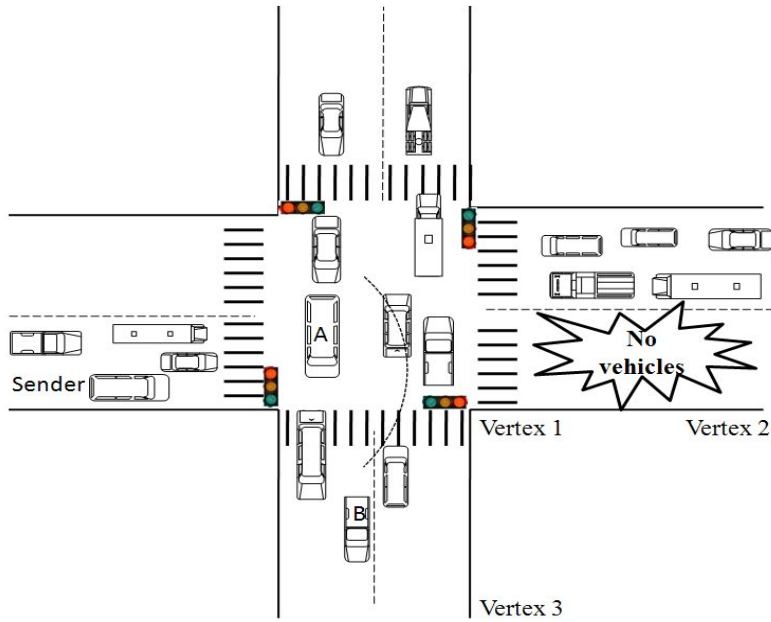


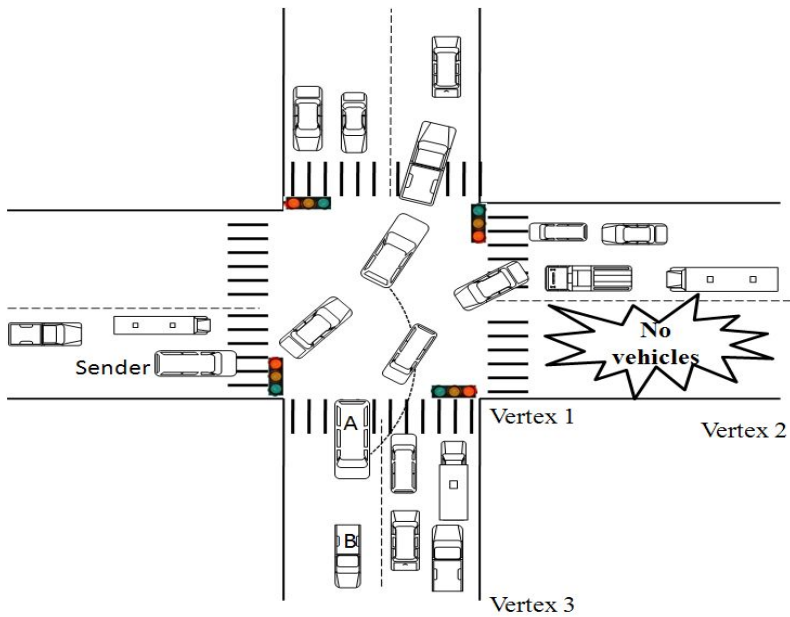
Fig. 3: An example of routing path in GSR.

PUT, similar to conventional position-based source routing protocols, adopts the anchor-based routing approach with street awareness. So, data packets will be routed between vehicles according to the street map topology. However, unlike the conventional V2V routing protocols, where the sender statically computes a sequence of vertices the packet has to traverse in order to reach the destination, intermediate vertices in PUT are chosen dynamically one by one, considering both distance to destination and the current traffic status. Involved intermediate vertices are determined by each data forwarding vehicle associated with the periodically updated traffic information.

Fig. 4 exhibits the problem of conventional routing protocols without taking traffic control mechanism into consideration. According to Fig. 3, the calculated shortest routing path is $A-V1-V2-V4-V6-BS$. However, as illustrated in Fig. 4, if the sender stops and clusters with its neighbors at the street intersection or moves in a different direction from the computed shortest packet forwarding path, there may be no vehicles which can be the next router to forward the packet along edge $E(V1, V2)$. In this case, by considering the distance between the source and destination, a substitute routing path is $E(V1, V3)$, where the packet can be forwarded to vehicle A, and then forwarded to vehicle B.



(a) Stop at intersection with red traffic light



(b) Turn left with green traffic light

Fig. 4: An example of traffic status.

The prediction of a sequence of vertices in PUT is done as follows:

Notations:

S : Sender.

D : Destination.

$dis(S, D)$: Distance between S and D .

N : Next forwarding node of current sender.

Tab : Computed routing table with the shortest routing path.

i : Current vertex.

j : Next vertex of i along with the shortest routing path computed by the sender.

$V_{(1,2,3,...n...k)}$: The set of other neighbor vertices of i except j .

Note here, if we use the two-street intersections, $k = 2$.

Algorithm 1: Vertex selection

1: **if** a packet is relayed from the previous upstream node to i

2: **if** there is no N within the transmission range of S in the direction of j

3: **if** there is V_n can minimize $dis(S,D)$ **then**
 update Tab and forward the packet to V_n with the minimum $dis(S,D)$ according to the greedy forwarding mechanism.

4: **else**
 carry and forward the packet to j

5: **else**
 forward the packet to j according to the greedy forwarding mechanism

6: **else**
 forward the packet to i according to the greedy forwarding mechanism

7: **end if**

In Fig. 3, let us suppose that the sender wishes to forward data to the nearest base station. It can identify a sequence of vertices between itself and the nearest base station with the help of a city map data provided by the navigation system. There are three identified routes to get to the destination; they are $A-V1-V2-V4-V6-BS$, $A-V1-V3-V4-V6-BS$ and $A-V1-V3-V5-V6-BS$. I assume that the sender prefers the shortest path among these three routes. However, a routing hole problem occurs at $V1$ as illustrated in Fig. 4. The sender A identifies this situation and then reselects the third path to forward the packet. Additionally, if the packet is relayed at $V5$, and there is only one path $E(V5, V6)$ can minimize $dis(S, D)$. In this case, I use the strategy of "carry and forward" [23] to send the packet close to the destination when there is no forwarding vehicle on the calculated shortest path.

4.2 Packet Forwarding Between Two Vertices

Once the sequence of valid vertices between the source vehicle and base station is determined, the improved greedy strategy is used to forward packets between the two involved vertices. Each vehicle maintains a neighbor table in which the position, velocity, acceleration/deceleration and direction of each neighbor vehicle are recorded. This table is updated through hello messages exchanged periodically by all vehicles. Thus, when a packet is received, the forwarding vehicle computes the new predicted position of each neighbor using the recorded information and then selects the next hop neighbor. I explain the proposed greedy routing strategy based on Fig. 5. In Fig. 5(a), when vehicle A is moving in the same direction as the sender with a higher speed than vehicle B , vehicle A will receive the forwarded packet since at time2 illustrated in Fig. 5(b), it is the closest vehicle to the next vertex. Without using this prediction, the forwarding vehicle would choose vehicle C as leading the routing loops. In this paper, as I implement the stop signs beside the streets as illustrated in Fig. 4(c) and (d), the greedy routing prediction will become more complicated. In Fig. 4(c), if vehicle A has a higher moving speed it is supposed to

receive the packet from the sender. However, if vehicle A has to stop at a stop sign with probability of P_{st} as illustrated in Fig. 4(d), vehicle B then, is better than vehicle A to be the forwarding router, and will receive the packet from the sender. In this case, I cannot determine the forwarding vehicle by the vehicle's direction and speed. I also need to consider the vehicle's position and acceleration/deceleration. It is obvious vehicle A has the highest deceleration, which is not an ideal next forwarding vehicle even if it moves faster than vehicle B. This is because vehicle A will stop somewhere in a short time, due to the environmental constraints. In contrast, vehicle B has already passed the stop sign, and can move at a stable speed, without deceleration. Consequently, vehicle B is the ideal forwarding neighbor for the sender vehicle. In this situation, there is a high risk that a packet will get stuck in a local optimum, where the forwarding vehicle might be the closest to the next vertex. Hence, a recovery strategy is required. The repair strategy of PUT is based on a "carry and forward" scheme [23], where the forwarding vehicle of the packet in recovery mode will carry another vehicle, closer to the destination.

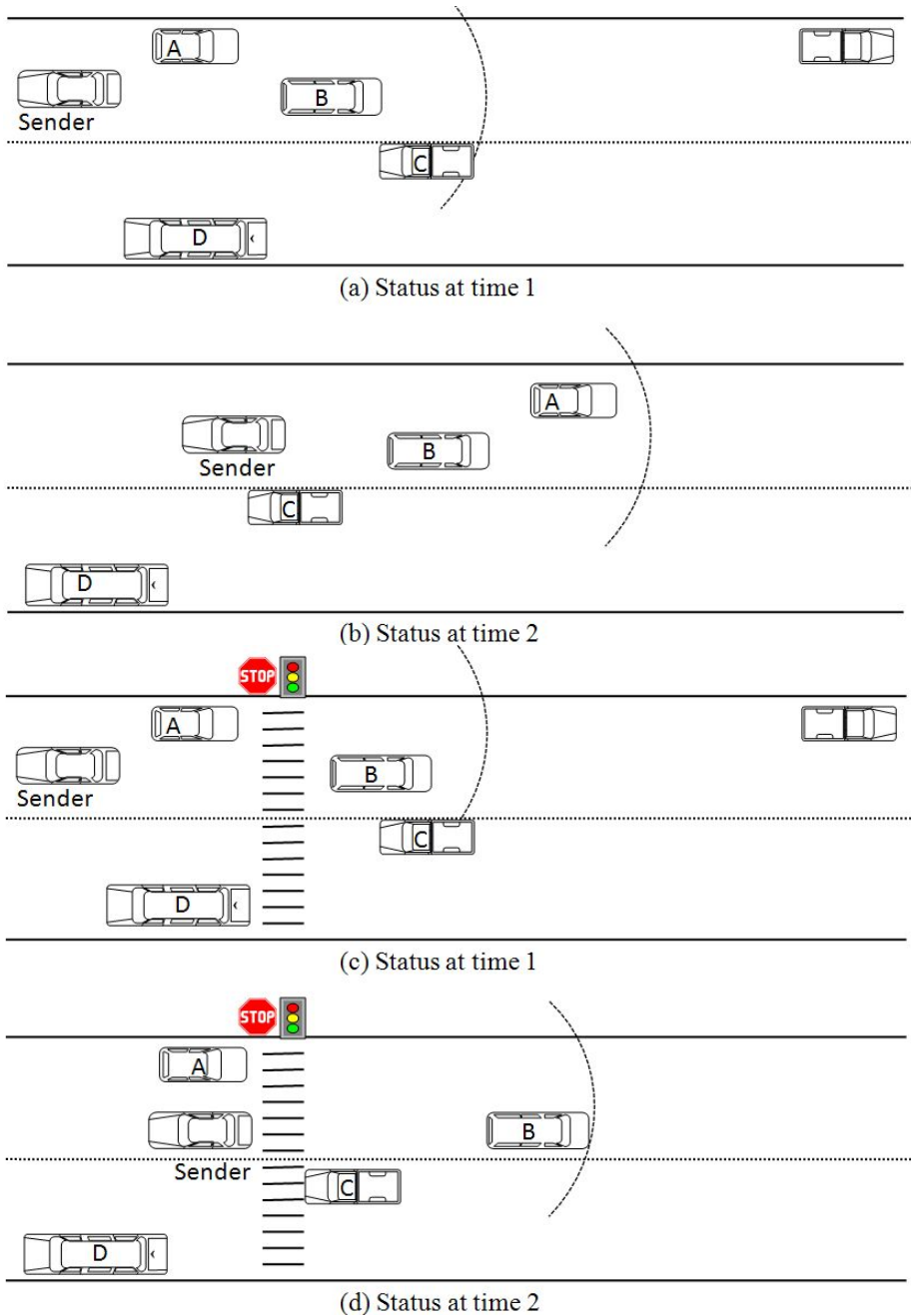


Fig. 5: An example of greedy forwarding.

5. Performance Evaluation

In this section, I evaluate the performance of our proposed routing protocol by using the *ns-2* simulator, which is a discrete event simulator developed by the University of California at Berkeley and VINT project. I used version ns-2.34 [24] based on the Monarch extensions to ns-2. The simulator models node mobility, allowing for experimentation with ad hoc routing protocols that must cope with frequently changing network topology. It implements the IEEE 802.11 Medium Access Control (MAC) protocol. The IEEE 802.11DCF is used with a channel capacity of 2 Mbps for vehicular sensors. I compare the performance of PUT with existing routing protocols GSR [15] and GyTAR [20]. They are representative reactive and geographic routing protocols, respectively, for V2V wireless networks.

5.1 Simulation Environment

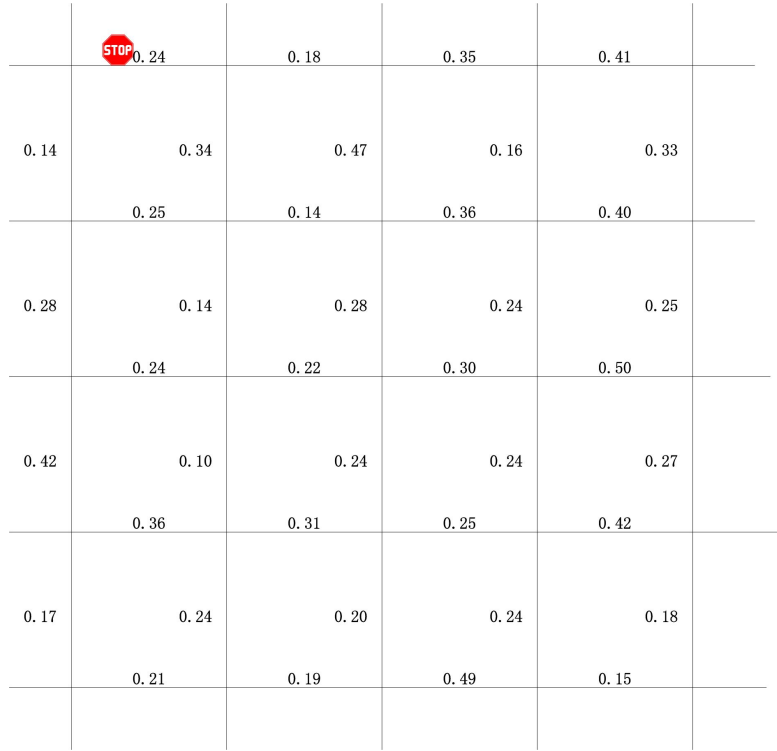
The experiment is based on a $2000\text{m} \times 2000\text{m}$ rectangular street area, which presents a grid layout. For the simulation, a $2000\text{m} \times 2000\text{m}$ area is chosen, consisting of 25 junctions or intersections and 10 two way roads. This street layout is derived and normalized into a realistic mobility trace from a Manhattan mobility model. The map data was then transformed into the data format used by the ns2, simulation tool. Vehicles with random start points and destinations were placed on the map. The model vehicles were assigned a maximal speed of 60km/h with accelerating/decelerating speeds of $-10 \sim 10\text{m/s}^2$. Each vehicle had radio propagation ranges of 250m. For the performance evaluation, 15 random connections were established using CBR traffic varying $1 \sim 16$ packet(s)/second with a packets size of 128 bytes. The value of the probability *Pinter* to stop the vehicles at street intersections when the vehicles reach an empty queue was set to 0.25. On the

other hand, the probability P_{st} that a vehicle would stop at a sign with an empty queue was randomly set in a range from 0.1 to 0.5. An example of distributed P_{st} of each stop sign is shown in Fig. 6. Take note that I varied the value of distributed P_{st} in different simulation runs. The maximum value T for waiting at intersections or stop signs is given as 10 seconds. All the key parameters of our simulation are summarized in Table.1. The simulation results are averaged over ten runs. Each simulation takes 900 seconds of simulation time.

The performance metrics used to evaluate the simulation results are as following:

- Packet delivery ratio: the ratio of originated data packets that are successfully delivered to their destinations to the original sent ones.
- Average end-to-end delay: the average time it takes for a packet to traverse the network from its source to destination.
- Routing overhead: the ratio of the total number of bytes of control packets to the total number of bytes of data packets delivered to the destinations during the entire simulation.

The routing protocols are compared under various data transmission rates and various vehicle densities. For the traffic generation in variable node densities, I set a constant packet sending rate i.e., 4 packets/second. On the other hand for traffic generation with variable packet sending rate I kept the number of nodes constant i.e., 200 nodes. Detailed analysis of the simulation results are given in the following.



]

Fig.6: An example of the distribution of Pst .

Table 1. Parameters used in the simulation

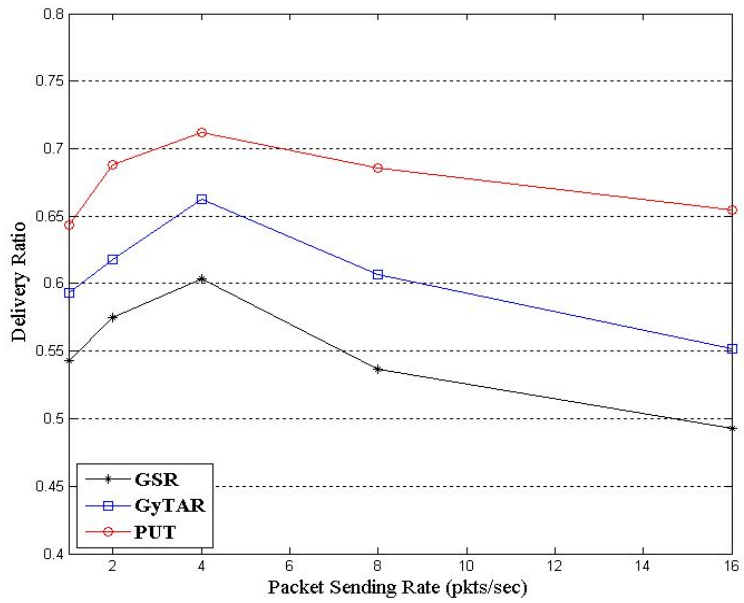
Simulation setup/Scenario		MAC/Routing	
Network area	2000m×2000m	Channel capacity	2 Mbps
<i>Pinter</i>	0.25	Packet sending rate	1-16 packets/second
<i>Pst</i>	0.1-0.5		
Vehicle speed	0-60 km/hour	Data packet size	128 bytes
Acceleration/ Deceleration	-10~10m/s ²		
Stop time (<i>T</i>)	10 seconds	MAC protocol	IEEE 802.11 DCF

5.2 Simulation Results and Discussion

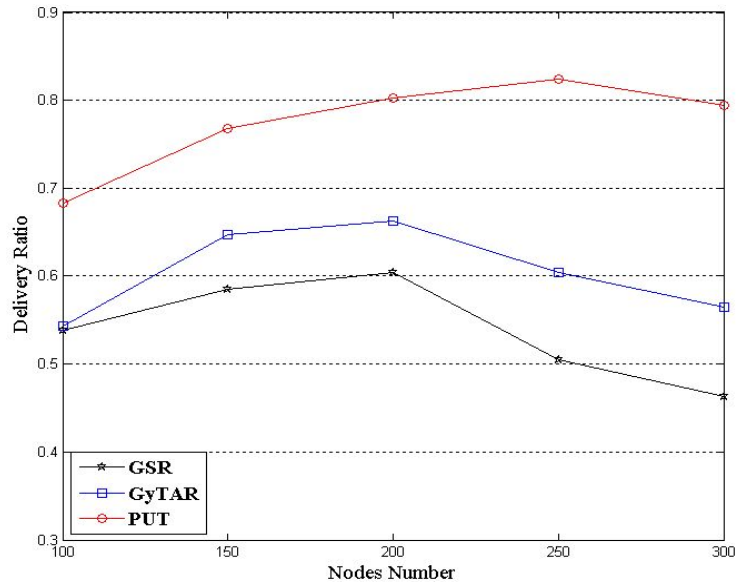
5.2.1 Packet Delivery Ratio

In this part, I compare the performance of PUT, GSR, and GyTAR in terms of packet delivery. For better performance, protocols should be tolerable to a small amount of packet loss. I will show how packet delivery is affected by the packet sending rate and the nodes' density.

In Fig. 7(a), GSR has the worst performance, i.e. less than 50% delivery ratio for 16pkt/sec. In case of GyTAR, delivery rates increase up to almost 56% for 16pkt/sec. Our proposed PUT achieves the highest packet delivery ratio across all packet sending rates observed. As many as 10% more packets are delivered by PUT than GyTAR. This is mainly because in PUT, the path is determined progressively following the current road traffic status. The data routing path is altered when routing holes occur due to traffic control mechanisms (while only the shortest path is used for route selection in GSR and the path with the most nodes is selected in GyTAR). A packet will move successively closer to the destination along streets which have good traffic situations providing good network connectivity. In Fig.7(b), all three protocols improve in reliability as the number of nodes increased. This is expected since more nodes increases the probability of connectivity, which in turn reduces the number of packets dropped due to local maximums. But, when the network density increases too much (>200) there is a decrease in the delivery ratios of GSR and GyTAR. This is because there is a high probability of vehicles being queued in front of stop signs and street intersections. Radio interference and collisions between nodes increase when many nodes are clustered together (PUT can improve the delivery ratio decrease threshold value up to 250). In this situation, I need a traffic status awareness routing protocol which selects the routing path based on the current traffic status. In general, PUT has a much higher delivery ratio than competitors because with local traffic awareness the packets can be routed successfully instead of being dropped.



(a) Varying packet sending rate



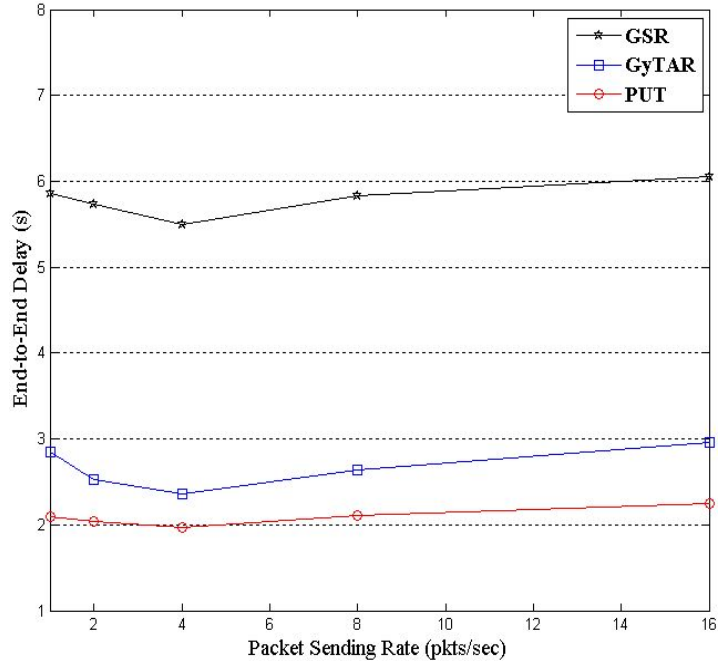
(b) Varying the number of nodes

Fig. 7: Packet delivery ratio.

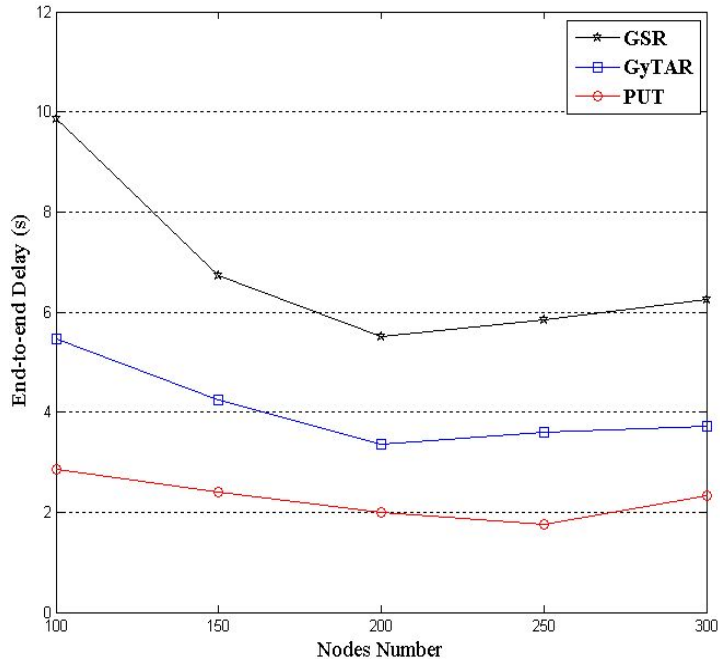
5.2.2 Average End-to-end Delay

In this section, I compare the performance of PUT with GSR and GyTAR in terms of end-to-end delay. As shown in Fig. 8, PUT achieves a much lower end-to-end delay than GSR and GyTAR in all tested configurations. This is mainly because in PUT, the number of hops involved to deliver packets is reduced due to the improved greedy strategy used to forward packets between two connected vertices, and also because PUT does not need to keep track of an end-to-end route before sending data packets from source to destination. More importantly, PUT not only considers the moving direction and speed of neighboring vehicles but also considers the position and acceleration/deceleration of them. This condition can help our protocol to choose a stable route for forwarding data to the destination.

Delay in GSR is higher than GyTAR and PUT because packets whose deliveries were suspended are stored in the buffer for a longer time than in GyTAR and PUT's. GyTAR's delay is higher than PUT because GyTAR is not suited for more complicated traffic environments, and will select nodes with high moving speeds but low deceleration unlike PUT. Fig. 8(a) shows the results of varying packet sending rates. Up to packet sending rate 4, the three delay plots decrease slightly, but after that point they start slightly increasing. Fig. 8(b) illustrates the results of varying the node numbers. The plots display the opposite trend of delivery ratio. It first decreases as the number of nodes increases, and then (up to 200 for GSR and GyTAR, 250 for PUT) there is an increase thereafter.



(a) Varying packet sending rate

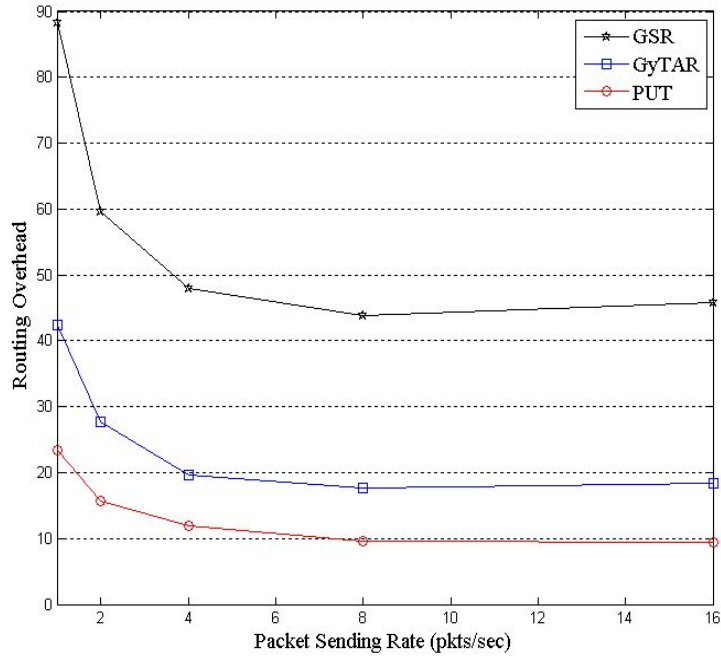


(b) Varying the number of nodes

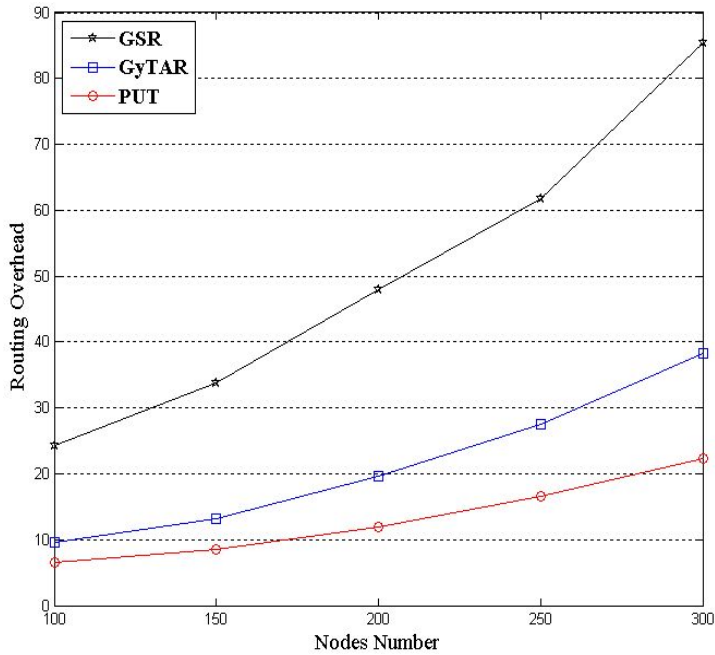
Fig. 8: Average end-to-end delay.

5.2.3 Normalized Routing Overhead

I evaluate the routing overhead of the three routing protocols as a function of data sending rate. In Fig. 9(a), it is observed that routing overhead decreases for all the protocols while increasing the packet sending rates up to 4 packets/sec. But, beyond 4 packets/sec routing overhead remains almost constant in all the routing protocols. This is expected since the number of control messages is constant for the same number of nodes (i.e. number of nodes is set to 200). As shown in Fig. 9(b), an increase in vehicle density leads to an increase in routing overhead since the rate of control messages depends on the number of nodes. In general, PUT outperforms the two competitors in all cases of varying data transmission rates and also with different vehicle densities. This is because in PUT, as in GyTAR, I have only three types of control messages, including Route request, Route reply, and Route error, which are used for route discovery and route maintenance. These control messages are updated accurately with the current traffic status. Therefore, in PUT, there is a low message re-transmitting rate that yields a low overhead plot. Although GSR uses only hello messages as control messages, it shows a higher routing overhead than PUT and GyTAR. This is because PUT and GyTAR do not need as many hello messages sent as GSR. This is due to the mechanism for a neighbor's position inference used in PUT and GyTAR. Hence, the frequency of hello messages needed by GSR is more than PUT and GyTAR.



(a) Varying packet sending rate



(b) Varying the number of nodes

Fig. 9: Normalized routing overhead.

6. More Discussion

Many V2V routing protocols have been evaluated with real city environments, where some street intersections have more than four entrances as illustrated in Fig. 10. In Fig. 10, sender vehicle have two ideal paths, i.e. segment B and segment C, to forward packets to the base station, and there are vehicles on these two paths maintaining network connectivity. In this case, vertices selection should combine another two conditions. One is the vehicle density [13] of each road segment, and the other is Pst . A road segment with a high density of vehicles and low Pst is a better selection as data routing path (among segments A and B). The vertices selection score S [13] can be modified in terms of curvometric distance (which is used to describe the distance measured when following the geometric shape of a road), traffic density, and the probability that vehicles will stop at a stop sign. The neighboring vertex which has the highest value of S will be selected as the next routing path. The calculation of S is done as follow.

Notations:

j : The next vertex

i : The current vertex

D_j : The curvometric distance from the candidate vertex j to destination

D_i : The curvometric distance from the candidate vertex i to destination

N : Total number of vehicles between j and i

n : Number of cells [13] between j and i

c : Ideal connectivity degree within a cell

r, s, t : Weighting factors for the curvometric distance, vehicular density, and vehicle stop probability, respectively ($r + s + t = 1$)

P_{st} : The probability to stop vehicles at the stop sign when it reach there with an empty queue between j and i .

Definition of Score

Score of j : $S(j) = r(1 - D_j/D_i) + s[\min(N \times c/n, 1)] + t(1 - P_{st})$

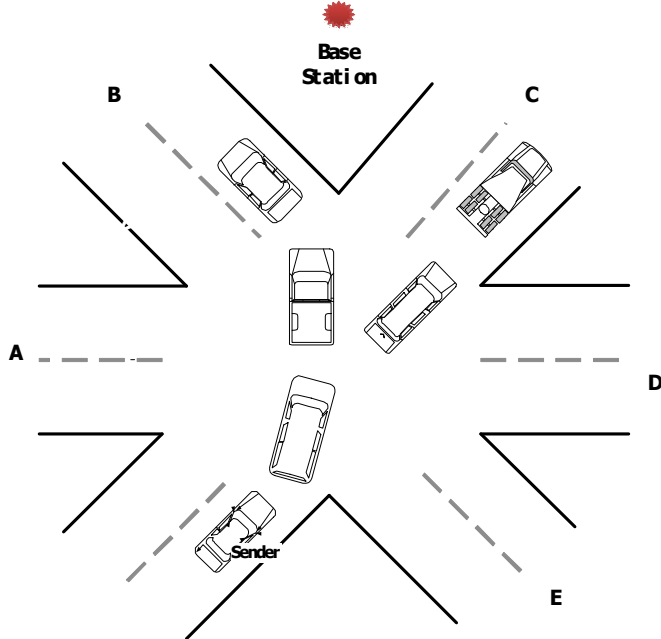


Fig. 10: An example intersection with more than four entrance.

7. Conclusions

In this thesis, the geographic prediction based routing protocol so-called PUT has been proposed for vehicular sensor networks in an urban environment. PUT considers the traffic control mechanism of traffic lights and stop signs. PUT performs the two key operations of the prediction of a sequence of vertices and the use of the predictive directional greedy routing to forward the data from a source vehicle to a destination through the sequence of vertices. The proposed routing protocol has been evaluated using the network simulator ns-2 (version 2.34) and compared with GSR and GyTAR in different conditions. The simulation results show that PUT outperforms GSR and GyTAR in terms of packet delivery ratio, end-to-end delay, and routing overhead. That is, PUT achieves higher performance and more reliable routing than GSR and GyTAR.

The proposed PUT selects vertices and intermediate vehicles in the city environment where sufficient vehicles are on the roads. As a future work, I will investigate the scenarios described in section 6, where some street intersections have more than four entrances to extend PUT.

References

- [1] A. Nandan, S. Das, G. Pau, M. Gerla, and M. Y. Sanadidi, "Co-operative downloading in vehicular ad-hoc wireless networks," Proc. of *IEEE WONS*, pp. 32-41. Jan. 2005.
- [2] A. Nandam, S. Tewari, S. Das, G. Pau, M. Gerla, and L. Kleinrock, "Ad-Torrent: delivering location cognizant advertisements to car network," Proc. of *IFIP WONS*, pp. 87-94, Jan. 2006.
- [3] Q. Xu, T. Mak, J. Ko, and R. Sengupta, "Vehicle-to-vehicle safety messaging in DSRC," Proc. of *ACM VANET*, pp. 19-28, Oct. 2004.
- [4] J. Ott and D. Kutscher, "A disconnection-tolerant transport for driven internet environments," Proc. of *IEEE INFOCOM*, pp. 1849-1862, Apr. 2005.
- [5] U. Lee, E. Magistretti, B. Zhou, M. Gerla, P. Bellavista, and A. Corradi, "Mobeyes: smart mobs for urban monitoring with a vehicular sensor network," *IEEE Wireless Communications*, Vol. 13, No. 5, pp. 52-57, 2006.
- [6] F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communication Magazine*, Vol. 40, Issue. 8, pp. 102-114, 2002.
- [7] J. M. Kahn. R. H. Katz and K. S. J. Pister, "Next Century Challenges: Mobile Networking for Smart Dust," Proc. of *IEEE International Conference on Mobile Computing and Networking*, pp. 270-278, Aug. 1999.
- [8] B. Hull, V. Bychkovsky, K. Chen, M. Goraczko, A. Miu, E Shin, H. Blakrishnan, and S. Madden, "Cartel: a distributed mobile sensor computing system," Proc. of the 4th *International Conference on Embedded Network Sensor Systems*, pp. 125-138, 2006.
- [9] MIT's CarTel Central, Available at <http://cartel.csail.mit.edu/>, 2010.
- [10] *InternetCAR Project*. [Online]. Available: <http://www.icar.wide.ad.jp>
- [11] T. Ernst, K. Mitsuya, and K. Uehara, "Network mobility from the InternetCAR

- perspective," *Proc. of AINA 2003 Interconnection Network*, vol. 4, no.3, pp. 329-343, Sept. 2003.
- [12] *FleetNet*, Available at <http://www.et2.tu-harburg.de>, 2010.
- [13] W. Enkelmann, "FleetNet-Applications for inter-vehicle communication," *Proc. of IEEE Intelligent Vehicles Symposium*, pp. 162-167. June 2003.
- [14] G. Korkmaz, E. Ekici, F. Özgüner, and Ü. Özgüner, "Urban multi-hop broadcast protocol for inter-vehicle communication systems," *Proc. of ACM Int. Workshop on Vehicular Ad Hoc Networks*, pp. 76–85, 2004.
- [15] C. Lochert, H. Hartenstein, J. Tian, D. Herrmann and M. Mauve, "A routing strategy for vehicular ad hoc networks in city environments," *Proc. of IEEE Intelligent Vehicles Symposium (IV2003)*, pp. 156-161, June 2003.
- [16] *Dijkstra's algorithm*. [Online]. Available: http://en.wikipedia.org/wiki/Dijkstra's_algorithm.
- [17] H. Lee, T. Kwon, and Y. Choi, "Virtual Vertex Routing for Course-Based Vehicular Ad Hoc Networks," *Proc. of IEEE Conf. on Wireless Communications and Networking*, pp. 4405-4410, Mar. 2007.
- [18] F. Kuhn, R. Wattenhofer, and Aaron Zollinger, "Worst-Case Optimal and Average-Case Efficient Geometric Ad Hoc Routing," *Proc. of 4th ACM Int. Symp. On Mobile Ad Hoc Networking and Computing*, pp. 267-278, 2003.
- [19] Robert W. Floyd, "ACM Algorithm 97: Shortest Path," *Comm. of the ACM*, vol.5, no.6, June 1962.
- [20] M. Jebri, S. Mohammed, R. Meraihi, and Y. G. Doudane "An Improved Vehicular Ad Hoc Routing Protocol for City Environments," *Proc. of IEEE Int. Conf. on Communications*, pp. 3972-3979, Jun. 2007.

- [21] C. Lochert, M. Mauve and H. Hartenstein, "Geographic routing in city scenarios," *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, vol. 9, no.1, pp. 69-72, Jan. 2005.
- [22] B. Divecha, A. Abraham, C. Grosan and S. Sanyal, "Impact of Node Mobility on MANET Routing Protocol Models." Available at <http://scholar.google.com>, 2010.
- [23] J. Li, J. Jannotti, D. De Couto, D. Karger, and R. Morris, "A scalable location service for geographic ad hoc routing", Proc. of *ACM/IEEE MOBICOM'2000*, pp. 120-130, 2000.
- [24] The Network Simulator – ns-2, Available at <http://www.isi.edu/nsnam/ns>, 2010.

저작물 이용 허락서

학 과	컴퓨터공학	학 번	20087778	과 정	석사
성 명	한글: 소 신 한문: 苏 新 영문: Xin Su				
주 소	광주광역시 동구 서석동 조선대학교 전자정보공과대학 정보통신실험실 810호				
연락처	010-2940-2207 E-MAIL: leosu8622@hotmail.com				
논문제목	한글 : 차량 센서 네트워크를 위한 교통 관제 인지 기반 라우팅 영문 : Traffic Control Aware Routing for Vehicular Sensor Networks				

본인이 저작한 위의 저작물에 대하여 다음과 같은 조건 아래 조선대학교가 저작물을 이용할 수 있도록 허락하고 동의합니다.

- 다 음 -

1. 저작물의 DB구축 및 인터넷을 포함한 정보통신망에의 공개를 위한 저작물의 복제, 기억장치에의 저장, 전송 등을 허락함
2. 위의 목적을 위하여 필요한 범위 내에서의 편집·형식상의 변경을 허락함.
다만, 저작물의 내용변경은 금지함.
3. 배포·전송된 저작물의 영리적 목적을 위한 복제, 저장, 전송 등은 금지함.
4. 저작물에 대한 이용기간은 5년으로 하고, 기간종료 3개월 이내에 별도의 의사표시가 없을 경우에는 저작물의 이용기간을 계속 연장함.
5. 해당 저작물의 저작권을 타인에게 양도하거나 또는 출판을 허락을 하였을 경우에는 1개월 이내에 대학에 이를 통보함.
6. 조선대학교는 저작물의 이용허락 이후 해당 저작물로 인하여 발생하는 타인에 의한 권리 침해에 대하여 일체의 법적 책임을 지지 않음
7. 소속대학의 협정기관에 저작물의 제공 및 인터넷 등 정보통신망을 이용한 저작물의 전송·출력을 허락함.

동의여부 : 동의(○) 반대()

2010년 8 월 25 일

저작자: 소 신 (서명 또는 인)

조선대학교 총장 귀하