JOURNAL OF KOREA MULTIMEDIA SOCIETY, VOL. 10, NO. 12, DECEMBER 2007(pp. 1733-1740)

A Wireless Location-Based Routing Algorithm in Multi-User Game Environments

Jong Min Lee†, Seong Woo Kim††, Jung Hwa Lee†††

ABSTRACT

In this paper, we propose a wireless location-based routing algorithm which uses the location information of its neighbor nodes and a destination node. At first, the proposed routing algorithm forwards a packet to the X direction by selecting a closest node to its destination as a next hop in terms of the X coordinate until the packet reaches closely to the packet's destination. Then the packet is forwarded to the Y direction by selecting a closest node to its destination in terms of the Y coordinate. We use a back off mechanism in case that a next hop cannot be found using the proposed routing algorithm, which resolves loops while forwarding. The experimental results show that the proposed routing algorithm performs well like the existing routing algorithms Ad hoc On-demand Distance Vector and Greedy Perimeter Stateless Routing. It is expected to use the proposed routing algorithm in the digital battlefield of military environments and survival games of commercial environments.

Keywords: Game, Wireless Network, Routing, Location Information

1. INTRODUCTION

As the military and commercial needs for mobile information systems increases [1], research topics on wireless communication have been studied widely. In military environments such as digital battlefield, wireless communication is very important to communicate with each other elements such as soldiers, airplanes, satellites etc. This kind of wireless technology can be adopted to multi-user game environments such as survival games. By adopting the technology used in the digital battlefield to connect people in the same team, participants in a survival game can understand their situation better and play collaboratively.

Participants in the survival game can be considered as mobile nodes. Thus, this kind of network can be regarded as an ad hoc network. An ad hoc network is different from the existing wired network in that a mobile node should perform a router function as well as a host function [2,3]. Routings for ad hoc networks can be classified into proactive routing and on-demand routing [4,5]. Proactive routings manage the routing path to destination nodes in the form of a table, which is also called a table-driven routing [5]. On-demand routings manage only routing information about necessary destination nodes, which results in small amount of resources. Routing algorithms for ad hoc networks can also be categorized into hierarchical routing and flat routing in terms of network configuration [6].

In this paper, we propose a location-based rout-

※ Corresponding Author : Jong Min Lee, Address : (614-714) Dept. of Computer Software Engineering, Dong-Eui University, 995 Eomgwangno, Busanjin-Gu, Busan, Korea, TEL : +82-51-890-1736, FAX : +82-51-890-2629, E-mail : jongmin@deu.ac.kr
Receipt date : May. 31, 2007, Approval date : Oct. 8, 2007
† Dept. of Computer Software Engineering, Dong-Eui University
†† Dept. of Computer Software Engineering, Dong-Eui University
(E-mail : libero@deu.ac.kr)
††† Dept. of Computer Software Engineering, Dong-Eui University
(E-mail : junghwa@deu.ac.kr)
※ This work was supported by Dong-Eui University Grant.(2004AA163)

ing algorithm in ad hoc networks using the location information of nodes. A node forwards a packet to the X direction first until the packet reaches near a destination and then it forwards the packet to the Y direction to the destination. By comparing the location information of a packet to be forwarded to a certain destination with its location information, a node selects a next hop for the packet. We use a back off mechanism in case that a next hop cannot be found using the proposed routing algorithm.

The rest of this paper is organized as follows. Section 2 discusses some related work. In Section 3, we describe the proposed routing algorithm and its implementation issues. Performance evaluation of the proposed routing algorithm is described in Section 4. Finally, the conclusion is given in Section 5.

2. RELATED WORK

In AODV [7] and DSR [8], it is required to broadcast a RREQ message in order to set up a routing path from a source node to a destination node, which increases the network traffic and the routing delay. As RREQ traverses to a destination, the address of each intermediate node is added to the RREQ header in DSR. A destination node receiving RREQ can find the routing sequence in RREQ. On the contrary, the forward path setup and the reverse path setup are performed in each intermediate node when AODV is used, which differs from the routing path setup in DSR.

Greedy Perimeter Stateless Routing (GPSR) [9] uses greedy packet forwarding, which chooses a locally optimal next hop from the viewpoint of the distance. If greedy packet forwarding cannot be used, perimeter forwarding is used. Two main assumptions are used in GPSR, which is the same assumptions used in this paper. First, it assumes that each node knows location information of its own and neighbor nodes. Second, it assumes that the location information of a destination is also known. GPSR shows good scalability on densely deployed wireless networks.

Location-Aided Routing (LAR) algorithm uses two zone concepts such as an expected zone, which is the possible location of a destination node after Δt, and a request zone, within which messages to find a route to the destination node should be forwarded, to reduce the message traffic which is needed for a route discovery [10]. The Geographical Adaptive Fidelity (GAF) algorithm has been devised to reduce energy consumption in ad hoc networks by using location information and virtual grids to determine node equivalence [11].

3. SYSTEM DESCRIPTION

3.1 Terminology

The location information of a node and a neighbor set is described as follows:

- A triple (i, x_i, y_i): two-dimensional coordinates (x_i, y_i) of node i
- A set of neighbor nodes of node i, N_i: a set of triples (j, x_j, y_j) which can be seen in the node j's HELLO beacon

Partial neighbor sets are defined as follows, which are used to find candidates for finding a next hop.

- $N_i^{+X} = \{(j,x_j,y_j) | x_j \geq x_i\}$ for all $(j,x_j,y_j) \in N_i$
- $N_i^{-X} = \{(j,x_j,y_j) | x_j < x_i\}$ for all $(j,x_j,y_j) \in N_i$
- $N_i^{+Y} = \{(j,x_j,y_j) | y_j \geq y_i\}$ for all $(j,x_j,y_j) \in N_i$
- $N_i^{-Y} = \{(j,x_j,y_j) | y_j < y_i\}$ for all $(j,x_j,y_j) \in N_i$

3.2 Basic Concept

Basically, we assume that a node knows its location information and the coordination of a destination node. A node can find its location information, for example, using the GPS information. A node only uses its local information and the location information of a destination node. A node can find its neighbors' location information by exchanging HELLO messages periodically. The location information of a destination node can be found with the broadcast of its location information when the node participates in a certain ad hoc network.

Fig. 1 shows the basic concept of the proposed algorithm. If a node receives a packet to a certain node, it compares its X coordinate with that of a destination node and then it selects the closest node as a next hop in terms of its X coordinate. The packet is forwarded to the X direction until it reaches a certain node, which is in the shaded area of Fig. 1, close to the destination in terms of the X coordinate. Then it is forwarded to the Y direction until it reaches to the destination by selecting the closest node as a next hop in terms of its Y coordinate. In Fig. 1(a) and (b), the packet is forwarded to its X direction. In Fig. 1(c) and (d), the packet is forwarded to its Y direction to reach its destination.

The proposed routing algorithm differs from GPSR in that it does not select the geographically closest node as a next hop. It selects a node closest to either X direction or Y direction by considering the network as a 2-D mesh. The routing direction only changes from the X direction to the Y direction as soon as the packet arrives at a certain area which is close to the destination. If it is impossible to find a next hop for the destination, the back off mechanism is used, which will be described later.

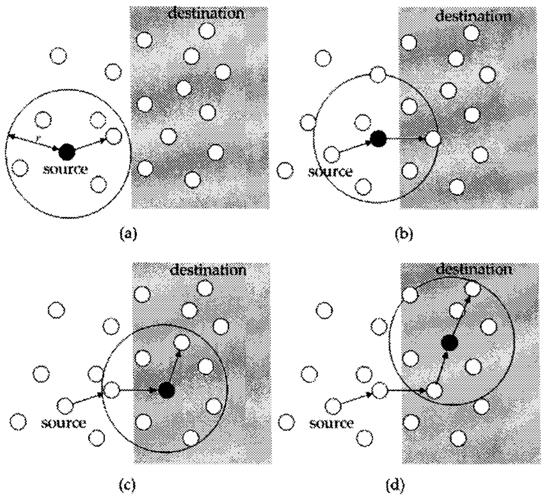

Fig. 1. An example of packet forwarding from a source to a destination.

3.3 Routing Algorithm

Fig. 2 shows the basic routing algorithm proposed in this paper. Step 1 shows the termination condition, which means the successful packet delivery to a destination node if the condition is satisfied.

Step 2 shows how to forward a packet to the X direction. If the X coordinate of the current node is quite close to that of the destination node as in the condition in Step 2.1, there should be a change in the routing direction to the Y direction, which is shown in Step 3. And also there should be a

```
Input:
    * Destination location information (d, x_d, y_d)
Output:
    * A next hop h
Variables:
    * const int X_MOVE_FORBIDDEN = 0x01;
    * const int Y_MOVE_FORBIDDEN = 0x02;
    * int dir_flag = 0x00;

Step 1. [TERMINATION CONDITION]
1.1 if (d == i), stop;

Step 2. [X-MOVE]
2.1 if ((|x_i − x_d| < ε) go to Step 3;
2.2 else if ((x_i < x_d) ∧ (N_i^{+X} ≠ ∅)) {
        h = j such that max{x_j|(j, x_j, y_j, ···) ∈ N_i^{+X}};
        return;
    }
2.3 else if ((x_i > x_d) ∧ (N_i^{−X} ≠ ∅)) {
        h = j such that min{x_j|(j, x_j, y_j, ···) ∈ N_i^{−X}};
        return;
    }
2.4 else if (((x_i < x_d) ∧ (N_i^{+X} == ∅)) ∨ ((x_i > x_d) ∧ (N_i^{−X} == ∅))) {
        dir_flag ^= X_MOVE_FORBIDDEN;
        if (dir_flag & Y_MOVE_FORBIDDEN) go to Step 4;
        else
            go to Step 3;
    }

Step 3. [Y-MOVE]
3.1 if ((y_i < y_d) ∧ (N_i^{+Y} ≠ ∅)) {
        h = j such that max{y_j|(j, x_j, y_j, ···) ∈ N_i^{+Y}};
        return;
    }
3.2 else if ((y_i > y_d) ∧ (N_i^{−Y} ≠ ∅)) {
        h = j such that min{y_j|(j, x_j, y_j, ···) ∈ N_i^{−Y}};
        return;
    }
3.3 else if (((y_i < y_d) ∧ (N_i^{+Y} == ∅)) ∨ ((y_i > y_d) ∧ (N_i^{−Y} == ∅))) {
        dir_flag ^= Y_MOVE_FORBIDDEN;
        if (dir_flag & X_MOVE_FORBIDDEN) go to Step 4;
        else
            go to Step 2;
    }

Step 4. [PACKET DROP]
4.1 drop the packet;
    stop;
```

Fig. 2. Basic routing algorithm.

change in the routing direction if anode cannot find a next hop in Step 2. In this case, a next hop for the Y direction will be investigated as though the packet still meets the condition of Step 2.

In Step 3, a next hop is investigated for the Y direction until it is delivered to the destination, which means that the packet is near to the destination. Step 3.3 shows the packet drop condition. If a node cannot find a next hop for either X or Y direction, then the incoming packet should be removed from the network, which can cause a lot of packet removal. This kind of packet removal can be alleviated by considering a back off mechanism, which will be discussed in the later section.

Fig. 1 shows the packet delivery in case there is always a next hop candidate to its forwarding direction. Fig. 3 shows how a packet is delivered to its destination in case there is no next hop candidate to its forwarding direction. A packet is forwarded according to its normal forwarding direction in Fig. 3(a) and (c). In Fig. 3(b) and (d), partial neighbor sets N^{+X} and N^{+Y} are empty and thus next hop candidates are searched according to Step 3.1 and Step 2.2, respectively.

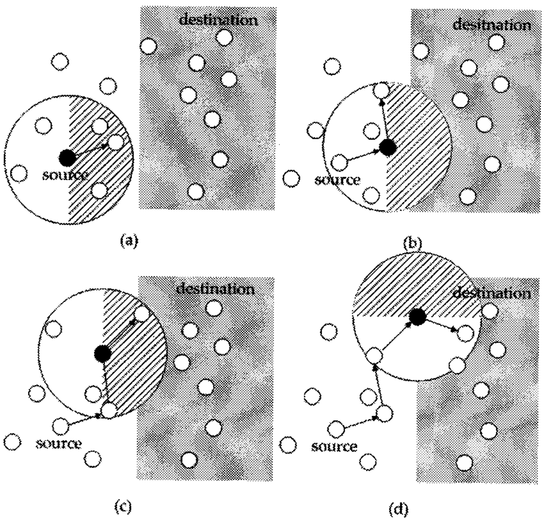

Fig. 3. Routing delivery in case there is no next hop candidate to its forwarding direction.

3.4 Implementation Issues

We consider some implementation issues when it is not easy to find a proper routing path due to the movement and the fault of a node.

3.4.1 Back Off Mechanism

Fig. 4 shows the situation that packet forwarding can be blocked. For simplicity, we assume that nodes are arranged in 2-D grid manner and that every node can only connect to its adjacent nodes because of the limited radio power. A packet is assumed to be delivered from node 0 to node 15. According to the basic routing algorithm, the packet is forwarded through node 1 and node 2. In node 2, the packet is delivered to the Y-axis direction according to Step 2.4 and Step 3.1. But there does not exist a neighbor node which can be used for a next hop candidate around node 6 to reach the destination. If only the basic routing algorithm is used, the packet cannot be delivered to node 15 and must be dropped. However, this kind of blocking can be avoided using the back off mechanism.

Now we describe how to determine whether the received packet cause a loop. Let $r(s,d,h_p,h_n,r_d) \in R$ be a route entry for a route cache R, where s, d, h_p and h_n represent a source, a destination, a pre-

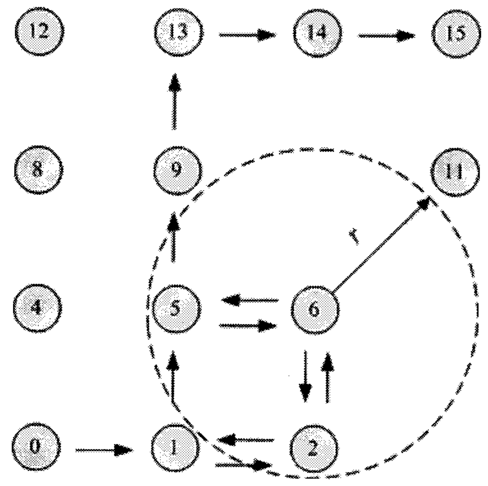

Fig. 4 An example of back off.

vious hop, and a next hop, respectively. r_d is an element of a routing direction set $\{+X,-X,+Y,-Y\}$. Let p_c be the current previous hop for a packet. There can be two kinds of loops: a direct loop and an indirect loop. If there exists an h_n such that

$$\{r(s,d,h_p,h_n,r_d)\in R \mid h_n = p_c\} \quad (1)$$

this means that the packet which is just forwarded to its neighbor $h_n(=p_c)$ is returned. We call this kind of a loop a *direct loop*, of which the hop distance is two. If a direct loop is found, it can be removed by searching a new next hop candidate in its neighbor nodes without the recent previous hop p_c. If a next hop candidate cannot be found using the basic routing algorithm, the packet should be backed off to h_p.

An *indirect loop*, of which the hop distance is greater than two, can be found in node i if there exists an h_n such that

$$\{r(s,d,h_p,h_n,r_d)\in R \mid (h_n = h_p)\wedge(((d_x<i_x)\wedge(r_d=+X)) \vee((d_x>i_x)\wedge(r_d=-X))\vee((d_y<i_y)\wedge(r_d=+Y)) \vee((d_y>i_y)\wedge(r_d=-Y)))\} \quad (2)$$

where (i_x,i_y) and (d_x,d_y) represent the location of node i and a destination node d, respectively. Eq. 2 indicates that it finds that a route entry which was inserted by the back off mechanism previously when a packet is forwarded to node i. Handling an indirect loop is similar to that of the direct loop except that the neighbor node ignored to find a new next hop candidate is the node h_n, not the previous hop p_c. If a new next hop candidate cannot be found using the basic routing algorithm, the packet should be backed off to the previous hop p_c.

3.4.2 Route Recovery

If a node cannot see a HELLO beacon of a neighbor node for a long time, this means that the neighbor node either moves from its place or does not operate correctly due to the power failure or other reasons. This may cause the routing problem if the neighbor node is one of nodes in a routing path for a certain destination. In this case, a route recovery process should be taken for reliable packet forwarding. In previous routing algorithms such as DSR and AODV, the route error message (RERR) should be sent to a source node along the reverse path to remove related cache entries. And then the source node should initiate a new route discovery for a destination. To reduce the path set-up delay, AODV performs local repair by sending RREQ to a destination from a node where a line break takes place if the destination is not far away.

This kind of local route recovery is possible in the proposed routing algorithm. Let the current routing path from a source to a destination be as shown in Fig. 5(a). If node 2 has some problem, for example, in power or moves to a certain area far from node 1's radio coverage as shown in Fig. 5(b), node 1 cannot forward packets from a source to a destination. After some times later, node 1 cannot see the HELLO message from node 2, which will remove the neighbor information of node 2 from its neighbor cache and related cache entries in the route cache. Then, node 1 finds another next hop candidate from a source to a destination, which is shown in Fig. 5(c) and (d).

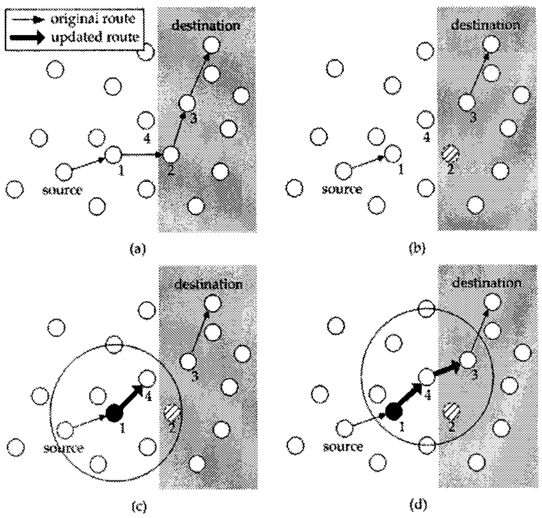

Fig. 5 An example of local route recovery.

4. PERFORMANCE EVALUATION

We have simulated the proposed routing algorithm using the network simulator *ns2* [12]. To see the performance of the proposed routing algorithm, we compared its performance with those of AODV [7] and GPSR [9] by varying the network traffic. We used the AODV implementation in *ns2* and the GPSR implementation in [13]. The assumptions used for the simulation is as follows. The network size is 1000m × 1000m. The number of nodes is one hundred. The nodes are static and are located randomly. The constant bit rate (CBR) traffic is used in the simulation, where a source node and a destination node are selected randomly and the packet generation rate is 512 bytes per second. The CBR traffic is generated using the CMU's traffic generator in *ns2*.

Fig. 6 shows the network throughput obtained by varying the number of maximum connections (mc). Here the network throughput means total number of bytes per second received in each destination node. One connection represents a source and a destination pair. Before time 200, the number of connections becomes its maximum connections. As the number of connections increases in time interval [0, 200], the network throughput also increases. After time 200, the traffic is generated with its maximum connections and the network throughput does not changes dramatically. The proposed routing algorithm shows the similar performance compared with AODV and GPSR in terms of the network throughput, which means that the proposed routing algorithm can handle the given traffic used in the simulation.

Fig. 7 compares the number of forwards of the proposed algorithm with those of AODV and GPSR. The X axis represents a certain connection

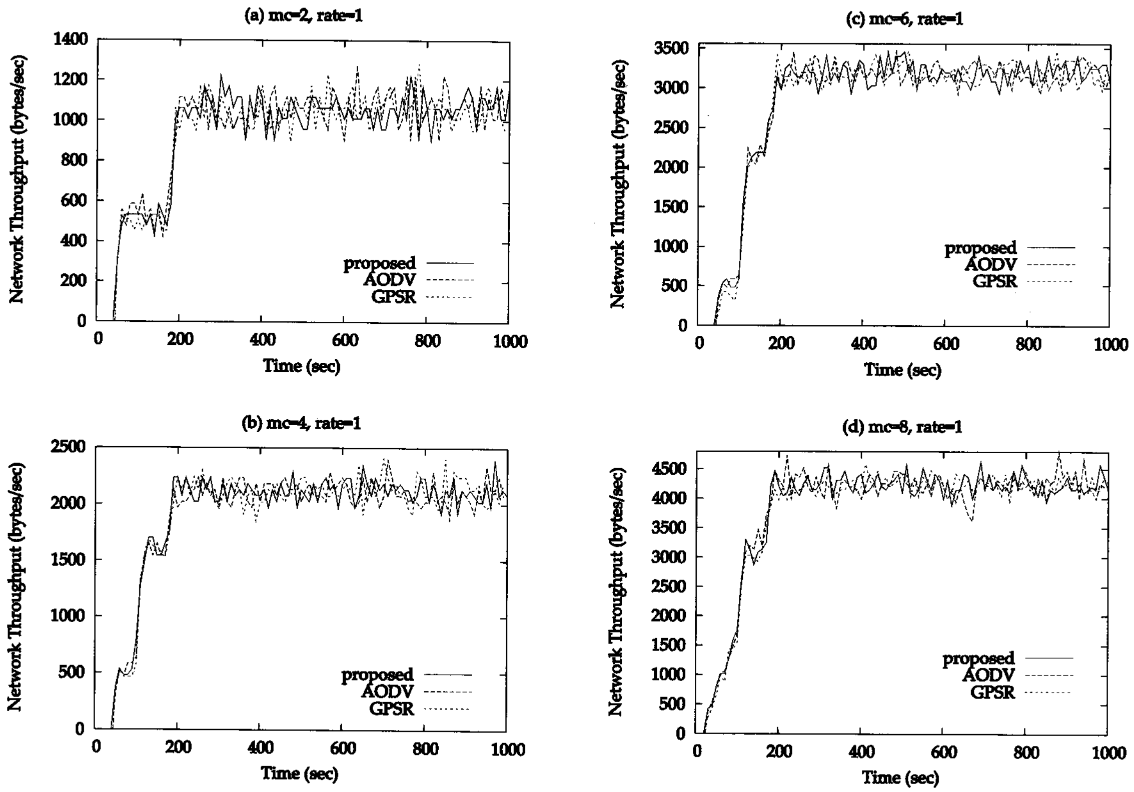

Fig. 6. Network throughput varying the number of source-destination pairs.

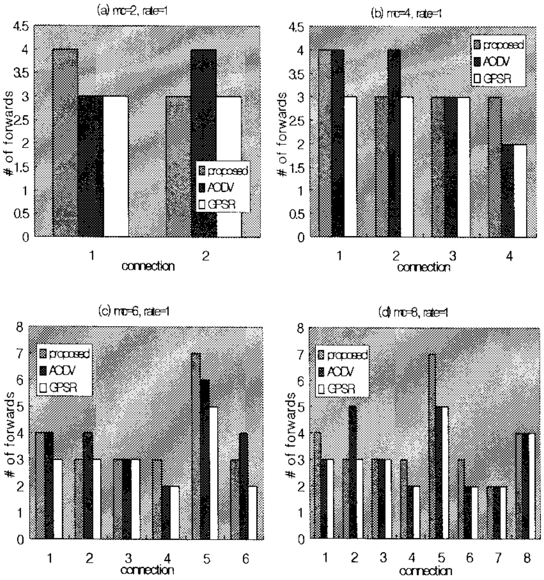

Fig. 7. Comparison of the number of forwards.

from a source to a destination. The proposed algorithm shows the similar performance compared with AODV. It shows an increase of at most two hops to reach a destination compared with GPSR, which results from the characteristic of GPSR to find a locally optimal next hop if possible. This experiment shows that the proposed routing algorithm does not make a large increase in the number of forwards compared with other routing algorithms.

5. CONCLUSION

In this paper, we have proposed a wireless location-based routing algorithm. The basic routing algorithm is to forward a packet to the X direction first until it reaches closely to a certain destination and then to the Y direction. While packets are forwarded to the given direction, there are some occasions when a next hop is not found due to either node movement or node failure. In this case, the proposed routing algorithm searches a next hop to the other direction for route discovery. If a next hop cannot be found nevertheless, the back off mechanism is used for route discovery. However, the back off mechanism may form a loop while forwarding a packet. Thus, we also devised a method how to find and break the loop. We have simulated the proposed routing algorithm using the network simulator *ns2.* The simulation results showed that the proposed algorithm works correctly using the back off mechanism and that the route recovery process works well.

REFERENCES

[1] B. M. Leiner, R. J. Ruth, and A. R. Sastry, "Goals and Challenges of the DARPA GloMo Program," *IEEE Personal Communications*, Vol.3, No.6, pp. 34-43, Dec. 1996.

[2] C.E. Perkins and P. Bhagwat, "Routing over Multi-hop Wireless Network of Mobile Computers," *in Mobile Computing (T. Imielinski and H. Korth, eds.)*, Ch.6, pp. 183-205, Kluwer Academic Publishers, Massachusetts, 1996.

[3] S. Corson and J. Macker, "Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations," RFC 2501, Jan. 1999.

[4] C.E. Perkins, "Ad Hoc Networking: An Introduction," *in Ad Hoc Networking (C.E. Perkins, ed.)*, Ch.1, pp. 1-28, Addison-Wesley, New Jersey, 2001.

[5] E.M. Royer and C.-K. Toh, "A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks," *IEEE Personal Communications*, Vol.16, No.2, pp. 46-55, April 1999.

[6] X. Hong, K. Xu, and M. Gerla, "Scalable Routing Protocols for Mobile Ad Hoc Networks," *IEEE Network*, Vol.16, No.4, pp. 11-21, 2002.

[7] C.E. Perkins, E. Belding-Royer and S. Das, "Ad hoc On-Demand Distance Vector (AODV) Routing," RFC 3561, July 2003.

[8] D.B. Johnson, D.A. Maltz, and Y. Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)," *IETF Internet Draft*, draft-ietf-namet-dsr-09.txt, Apr. 2003.

[9] B. Karp and H.T. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pp. 243-254, Aug. 2000.

[10] Y. Ko and N.H. Vaida, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks," *Proceedings of the 4th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 66-75, Oct. 1998.

[11] Y. Xu, J. Heidermann, and D. Estrin, "Geography-informed Energy Conservation for Ad Hoc Routing," *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, pp. 43-52, 2001.

[12] nsnam main page. http://nsnam.isi.edu/nsnam/index.php/Main_Page.

[13] Ke Liu's GPSR implementation, http://www.cs.binghamton.edu/~kliu/research/ns2code/index.html#gpsr.

Jong Min Lee

Rreceived the B.S. degree in computer engineering from Kyungpook National University, Korea, in 1992, and the M.S. and the Ph.D. degrees in computer science from Korea Advanced Institute of Science and Technology in 1994 and 2000, respectively. Since 2002 he has been an assistant professor of the Department of Computer Software Engineering, Dong-Eui University. From 2005 to 2006, he was a research associate at the University of California, Santa Cruz. His research interests include mobility management in wireless networks, routing in ad hoc networks and sensor networks.

Seong Woo Kim

Received his B.S., M.S., and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology, in 1991, 1993, and 1999 respectively. He worked at Electronics and Telecommunications Research Institute (ETRI) from 1999 to 2001. Since 2002 he has been an assistant professor of the Department of Computer Software Engineering, Dong-Eui University. His research interests include real-time operating system and sensor network.

Jung Hwa Lee

Received his B.S. and the Ph.D. degrees at the Department of Computer Science from Pusan National University, Korea, in 1995 and 2001, respectively. He is a professor at the Department of Computer Software Engineering, Dong-Eui University. His research interests include database and semantic web, etc.