Implementation of Controlled Sink Mobility Strategies with a Gradient Field in Wireless Sensor Networks

Shinya Toyonaga*, Yuki Fujita*, Daichi Kominami† and Masayuki Murata*
*Graduate School of Information Science and Technology, Osaka University, Osaka, Japan
Email:{s-toyonaga, y-fujita, murata}@ist.osaka-u.ac.jp
†Graduate School of Economics, Osaka University, Osaka, Japan
Email:d-kominami@econ.osaka-u.ac.jp

***Abstract*—Reduced energy consumption and extension of network lifetime are important challenges for wireless sensor networks, due to the energy-constraint properties of its elements. To solve these problems, many studies have been conducted on incorporating mobile sinks and controlled mobility into wireless sensor networks. When a mobile sink relays data from a sensor network to an operator, some sensors can save energy by reducing the number of transmitted packets and the communication range. Another important advantage is that sparse and disconnected networks are better handled when a mobile node relays packets between networks. In existing methods, however, all or some nodes must know their own location through the use of devices such as Global Positioning System (GPS) receivers. Yet it entails a high cost when each node is equipped with a GPS receiver. Moreover, GPS-based localization solutions cannot provide reliable location estimate in indoor environments, and in the presence of obstacles. We propose a method for controlling the mobility of a mobile sink so that it moves toward a targeted sensor node without GPS. In the proposed method, a sensor node selected as a target node broadcasts a control message, and each node records a hop count from the target node so as to construct a gradient field. A mobile sink can approach the target node using the gradient field. We evaluate our method by computer simulation and experiments using a cleaning robot that implements our method. We show that a mobile node with our proposed method can reach a target node in about 6 min.**

***Index Terms*—mobile sink; controlled mobility; sensor network.**

I. Introduction

The main challenges for Wireless Sensor Networks (WSNs) include reduced energy consumption and lifetime extension. To solve these problems, many controlled mobility protocols have been studied [1-6]. A mobile node that moves to collect information in WSNs is called a mobile sink. In controlled mobility, an operator deploys mobile sinks, whose mobility is dynamically controlled by received information from inside or outside of networks.

When a mobile sink relays data from sensor networks to an operator, some sensors can save energy from the reduced number of transmitted packets and communication range. Another important advantage is that sparse, disconnected networks can be better handled, since a mobile node can relay packets from one network to another. However, in existing methods, it is important to point out that all or some nodes need to know their own location information using devices such as GPS receivers. Yet, it entails a high cost when each node is equipped with a GPS receiver. Moreover, GPS-based localization solutions cannot provide reliable location estimate in indoor environments, and in the presence of obstacles.

In this paper, we propose a method to control the mobility of a mobile sink so that it moves toward a target sensor node by using a gradient field. The concept of a gradient field has been applied in gradient-based routing [7]. In gradient-based routing, all nodes have a gradient, which is a value presenting the direction through which the sink can be reached. Such a gradient field can be set up according to different information, such as hop count, energy consumption, or physical distance. When each node forwards a data along the gradient field, the data can reach the sink. In this paper, we use a gradient field for guiding a mobile sink to a location where a specific node is deployed. For example, a device such as a Personal Digital Assistant (PDA) floods a message, and PDAs that receive the message record the hop count from the flooding PDA. When such a hop count is assigned to PDAs, they can guide a mobile sink, such as a rescue robot, to the position where a person is requesting help.

In the proposed method, a sensor node selected as a target node floods a control message, and each node records a hop count from the target node. Then, a gradient field towards the target node is set up. When a mobile sink moves along the gradient field, like a data flow in gradient-based routing, it can reach the target node. For that, a mobile sink intercepts hop count information exchanged in the network. It then approaches the sensor node with smaller hop count and finally reaches the target node.

In our method, a mobile sink does not use any GPS receivers, but rather a Received Signal Strength Indication (RSSI) of messages exchanged in the sensor network. Because a gradient field is insufficient for a mobile sink to determine the direction to the node, mobile sinks measure the RSSI of a message from a sensor node, and searches for directions in which the RSSI is larger. Specifically, if a mobile sink goes straight and the RSSI increases, it continues to go straight because it approaches the sensor node. When going straight decreases the RSSI, the mobile sink measures RSSI while turning, and heads in the direction in which the RSSI is largest. By repeating these processes, a mobile sink can get closer to

TABLE I
DESCRIPTION OF CONTROL MESSAGES

Information	Description
node_id	Sender's node ID
hop_count	Hop count from sender to target
net_id	Sender's network ID

the sensor node.

The main advantage of using a gradient field is that a mobile sink needs not know the global information of the whole network. Instead, it only needs to know the local gradient information. Therefore, all nodes do not have to communicate over a long distance nor to send much information about a network.

We evaluate our proposed method by computer simulation and experiment using a cleaning robot. Our simulation and experiment results show that a mobile sink can be guided to the target node by using a gradient field. Since the main scope of our method in this paper is the mobility control of the mobile sink, our simulation and experiment focus on the capability of the mobile sink to be guided toward the target node by using a gradient field.

The rest of this paper is organized as follows: Section II presents our proposed method of controlled mobility. In Section III, we validate our method by computer simulation. Section IV discusses implementation using a cleaning robot, and we evaluate our method using the cleaning robot and sensors in Section V. Finally, Section VI presents our conclusions.

II. Controlled mobility with a gradient field

In our protocol, each node has a hop count from a target node, which is regarded as a gradient field. To assign a hop count from the target node to each node and update it, all sensor nodes send a control message to their neighbor nodes. A target node assigns its hop count to zero and generates a control message containing hop count that is initially set to one and advanced in increments by the forwarding nodes. The other node broadcasts the control message and assigns its hop count to the hop count contained in it. If a node receives more than one control message, the node assigns the hop count to the smallest hop count and broadcasts the message containing the smallest hop count. Table I shows the information included in the control message. Each node is assumed to have a unique identifier (node ID) to break ties, which can be easily removed by allowing each node to choose a random number in a large enough interval. Network ID is an identifier of the network which is comprised of a node connected to the target node. Mobile sinks intercept this control message to control their own mobility.

When a mobile sink receives a control message, it starts to control its own mobility along the gradient of the hop count field. First, it checks whether the network ID of the control message is the same as one it recorded. When the network ID is different from the recorded one, it ignores the control message. Otherwise, a mobile sink checks the hop count of the control message, and compares it to the hop count which it has recorded as the smallest one. If the received hop count is smaller than the currently recorded one, the mobile sink records the sender's node ID and its hop count, then tries to approach the sender. After the mobile sink receives data from a target node, it needs to restore its mobile algorithm because it may deliver data to a base station or move to the next network in which the other target node is located. When a mobile sink receives the data from a target node, it therefore records the network ID as the node that it has been arrived at and ignores control messages containing the same network ID as the recorded one.

However, a mobile sink cannot determine the direction of the sensor node because of feature of non-directional radio. We thus propose a controlled mobility using an RSSI. Our proposed method is applicable when mobile sinks are equipped with a non-directional antenna, but it can determine the direction more precisely when equipped with a directional antenna. Figure 1 shows the algorithm, which describes the operation of the controlled mobility. In that algorithm, h_{min} is a hop count, node_{min} is a node ID, and RSSI_{min} is an RSSI, when a mobile sink receives a control message containing a hop count that is the smallest it has received. net_{id} is the network ID of the target node that a mobile sink needs to approach to currently.

When a mobile sink receives a control message, it checks whether the sender is a target node (lines 1 and 2). If so, the mobile sink communicates with the target node and restores its own mobile algorithm (lines 3–6). Otherwise, the mobile sink checks the network ID included in the control message whether it is the same as net_{id}. If the network ID is different from net_{id}, a mobile sink ignores the control message (lines 8 and 9). Otherwise, the mobile sink checks the hop count included in the control message. If the received hop count is smaller than h_{min}, the mobile sink updates h_{min}, node_{min}, and RSSI_{min}, because it needs to approach the sender (lines 11–14). The mobile sink then searches for the direction to the sender (line 15). The operation of the searching direction to the node is described in Figure 2. If the received hop count is larger than h_{min}, the mobile sink ignores the control message (lines 16–18). If the received hop count is the same as h_{min}, the mobile sink checks the node ID included in the control message. If the node ID of the sender is different from node_{min}, the mobile sink ignores the control message and goes straight (lines 19 and 20). If the node ID of the sender is the same as node_{min}, the mobile sink checks the RSSI included in the control message. If the received RSSI is larger than RSSI_{min}, the mobile sink continues to go straight because it is approaching the sender (lines 21–23). Otherwise, the mobile sink searches for the direction to the sender, because it is getting farther from the sender (line 25).

In the following section, we use computer simulation to show that the proposed method can realize controlled mobility.

III. Verification of controlled mobility by computer simulation

We evaluate our method by computer simulation to show that the mobility of a mobile sink can be controlled. The monitoring area is 120×80 m. Seven sensor nodes and one

```
Algorithm 1 A mobile sink approaches a target node
 1: receive message $m$
 2: if the mobile sink receives $m$ from target node then
 3:     the mobile sink receives data from target node
 4:     the mobile sink records $m$.net_id
 5:     the mobile sink moves according to the usual mobile
        algorithm
 6:     return
 7: end if
 8: if net$_{id} \neq m$.net_id then
 9:     return
10: end if
11: if h$_{min} > m$.hop then
12:     h$_{min} \Leftarrow m$.hop
13:     node$_{min} \Leftarrow m$.node_id
14:     RSSI$_{min} \Leftarrow m$.RSSI
15:     search direction
16: else if h$_{min} < m$.hop then
17:     Go straight
18: else
19:     if node$_{min} \neq m$.node_id then
20:         Go straight
21:     else if RSSI$_{min} < m$.RSSI then
22:         RSSI$_{min} \Leftarrow m$.RSSI
23:         Go straight
24:     else
25:         search direction
26:     end if
27: end if
```

Fig. 1. Approach algorithm toward a target node

```
Algorithm 2 A mobile sink searches for the direction to the sender
 1: The mobile sink moves on the circumference centering
    on the current position (L_c) and records the position
    (L_max) where the mobile sink receives the message with
    the maximum RSSI.
 2: The mobile sink returns to (L_c).
 3: The mobile sink goes straight in the direction toward
    position (L_max).
```

Fig. 2. Direction search algorithm toward the sender

mobile sink are deployed at random places, and one of seven sensor nodes behaves as a target node. The model of radio attenuation is the free space model [8] and we assumed that no noise exists.

Our method is implemented in the OMNeT++ 4.1 [9] network simulator. We evaluate T_{move}, which is the time required for the mobile sink to approach the target node and to receive a control message from the target node. The default mobility algorithm of the mobile sink is the random way point model [10], which switches to the mobility algorithm shown in Section II, when it receives a control message from a sensor node. A mobile sink is deployed at random place in initial settings. The mobility based on the random way point model is shown in following.

1) A mobile sink selects the random place in the monitoring area.
2) It goes straight towards the selected place at constant velocity.

TABLE II
SIMULATION CONFIGURATION

Parameter	Value
Control message send time	1 s
Communication range	50 m
Data packet size	128 byte
Bandwidth	250 kbps
Mobile sink velocity	1 m/s
Mobile sink angular velocity	$\frac{\pi}{6}$ rad/s

TABLE III
SIMULATION RESULTS

	T_{move}
with controlled mobility	138.78 ± 67.17 s
without controlled mobility	308.30 ± 102.58 s

3) When it reaches the selected place, its mobility process returns to 1).

When a mobile sink goes straight and hits against a boundary of a monitoring area, it goes straight to the reflection direction. When a mobile sink searches direction to the node, it moves in a circle. When it moves in a circle and hits against a boundary of a monitoring area, it turns around in reverse. The details of the simulation configuration are summarized in Table II.

To compare the influence of our method, we evaluate T_{move} when the mobile sink moves only according to the random way point model. Table III shows simulation results. The number of trials is 50, and the confidence interval is 95%.

T_{move} is smaller when our method is implemented, so controlled mobility using a gradient field can be realized. The confidence interval when not using our method is larger due to the randomness of the random way point model, but the ratio between the confidence interval and the average T_{move} is larger under our method. This is because the number of circular movements when searching for a direction varies considerably according to the location where the mobile sink first receives a control message.

IV. Implementation of Controlled Mobility Using a Gradient Field

In this section, we describe how to implement controlled mobility in a mobile sink. We present an outline, and then details regarding configuration and system implementation.

A. Outline

For method implementation, as a mobile sink we use a patrolling robot such as an automatic cleaning robot. When the cleaning robot receives a control message from a sensor node while cleaning, it is guided to a target node along the gradient field. In our experimentation, the robot restores its mobility for cleaning after receiving a control message from the target node.

B. Configuration

In our experiment, we use a Roomba 790 (iRobot Corp.), an automatic cleaning robot with a publicly documented serial interface [11]. For sensor nodes we use the IRIS Mote XM2100 (Crossbow Technology) [12]. IRIS wireless modules

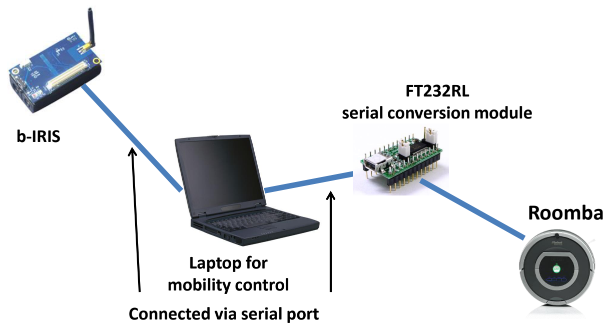
b-IRIS
FT232RL serial conversion module
Roomba
Laptop for mobility control
Connected via serial port

Fig. 3. Components of the mobile sink

TABLE IV
ROOMBA MOBILE PHASES

Phase	Description
FIND_NEWIRIS	Find an s-IRIS with smaller hop count
APPROACH_IRIS	Approach the s-IRIS with recorded node ID
REACH_TARGET	Reach target node
CLEAN	Do not control mobility and move based on default mobile strategy

are widely used for WSN development, are IEEE 802.15.4 compliant, and can be programmed using C#, Java, and other languages. We implement two types of IRIS with different functions. One is an s-IRIS (sensing IRIS), which generates sensing data and constructs the gradient field. The other is a b-IRIS (base station IRIS), which intercepts control messages and RSSI, and forwards them to the Roomba's mobility controller. We prepare a laptop computer for controlling the mobility of Roomba and attach it on the Roomba. The laptop computer decides the strategy of the mobility based on received information.

We used devices connected as shown in Figure 3. The b-IRIS was connected with a laptop computer via a serial port, and the laptop was connected with the Roomba via a serial conversion module (FT232RL). The laptop sends commands to the Roomba via the FT232RL, which serializes the command and forwards the results.

C. Implementation

Our proposed controlled mobility consists of three phases. The b-IRIS selects the next mobile phase based on information received from an s-IRIS. The b-IRIS then sends the next mobile phase and RSSI of the received control message to the Roomba's mobility controller, which selects an appropriate command according to the mobile phase and RSSI, and sends the command to the Roomba. Table IV shows these phases. In the following, we describe the processes and implementation of the b-IRIS and mobility controller.

1) B-IRIS processes : Figure 4 shows the state transition diagrams of the b-IRIS's process flow. The b-IRIS manages the mobile phase, as shown below, and informs the mobility controller of the next mobile phase. h_{min} is a hop count, node_{min} is a node ID, and RSSI_{min} is an RSSI, when the b-IRIS receives a control message containing the hop count that is the smallest so far received.

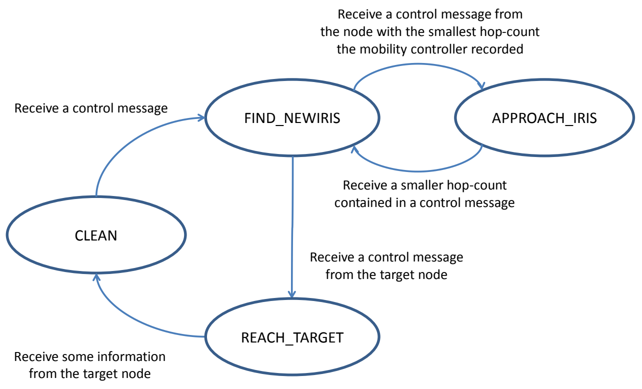
Receive a control message from the node with the smallest hop-count the mobility controller recorded
Receive a control message
FIND_NEWIRIS
APPROACH_IRIS
Receive a smaller hop-count contained in a control message
CLEAN
Receive a control message from the target node
REACH_TARGET
Receive some information from the target node

Fig. 4. State transition diagrams of the Roomba's mobile phases

a) CLEAN: When the b-IRIS receives a control message from an s-IRIS, it enters the FIND_NEWIRIS mobile phase.

b) FIND_NEWIRIS: When the b-IRIS receives a control message from a target node, it records the network ID to which the target node belongs, then enters the REACH_TARGET mobile phase. When the b-IRIS receives a control message from an s-IRIS, it informs the mobility controller of RSSI, and then the next mobile phase is APPROACH_IRIS.

c) APPROACH_IRIS: When the b-IRIS receives a control message from an s-IRIS, it compares the received hop count with h_{min}.

- If the received hop count is smaller than h_{min}, the b-IRIS updates h_{min} and node_{min}, then enters the FIND_NEWIRIS mobile phase.
- If the received hop count is larger than h_{min}, the b-IRIS does nothing.
- Otherwise, the b-IRIS checks the node ID of the sender. If the sender's node ID is the same as node_{min}, the b-IRIS informs the RSSI mobility controller. Otherwise, the b-IRIS records the node ID of the larger of the received RSSI and RSSI_{min}.

d) REACH_TARGET: When the b-IRIS receives a control message from a target node, it checks the received network ID. If the ID is already recorded, it ignores the received control message for a while. The next mobile phase is CLEAN.

2) Mobility controller processes: The mobility controller commands the Roomba based on the mobile phase and RSSI received from the b-IRIS.

a) CLEAN: When the mobility controller commands the Roomba to clean, the Roomba moves according to its original mobility algorithm.

b) FIND_NEWIRIS: The mobility controller discards and initializes all saved data.

c) APPROACH_IRIS: The Roomba approaches an s-IRIS. When the mobility controller searches for the direction of the s-IRIS, it commands the Roomba to rotate. The mobility controller searches for the direction with the largest RSSI, and goes straight in that direction. In order to exaggerate the difference in RSSI values, the b-IRIS is equipped with a pseudo-directional antenna in our experiments. The outline of the directional searching procedure is described below.

1) While the Roomba rotates, the mobility controller records the direction with the largest RSSI. When the mobility controller receives equally large RSSI from multiple directions, the mobility controller records the composition of those directions.
2) After the Roomba turns 360 degrees, the mobility controller commands the Roomba to turn to the recorded direction.
3) The mobility controller commands the Roomba to go straight. If the received RSSI tends to decrease while Roomba goes straight, the mobility controller processes 1).

In the following, we describe the directional searching procedure in detail. Each s-IRIS sends its hop count every second, and the angular velocity of Roomba is $\frac{2\pi}{N}$ rad/s when Roomba receives N messages and RSSI while it turns 360 degrees. The mobility controller numbers directions from 0 to $N-1$. This number is denoted by $RoombaDir$, and is initialized as 0 when the Roomba starts to turn. Then, every time the mobility controller receives RSSI, $RoombaDir$ is incremented through each direction. The mobility controller records the $RoombaDir$ with the largest RSSI as $TopDir$. If there are multiple directions with equally large RSSI, the mobility controller records the last two $RoombaDir$ values as $RD1$ and $RD2$, and the mobility controller records $\frac{RD1+RD2}{2}$ as $TopDir$. When the difference between $RD1$ and $RD2$ is $\frac{N}{2}$ or more, the mobility controller records $\frac{RD1+N+RD2}{2}$ mod N as $TopDir$ so that the Roomba does not go just in the opposite direction. The mobility controller uses an N s timer to decide when to finish searching directions. In an actual environment, the b-IRIS cannot always receive hop counts. Therefore, the mobility controller sets RSSI of the direction to 0 using the timer when the b-IRIS does not receive a hop count.

When the mobility controller decides $TopDir$, it needs to command Roomba to face the direction $TopDir$. When $TopDir$ is $\frac{N}{2}$ or more, the mobility controller commands the Roomba to turn for $(N - TopDir)$ seconds clockwise. Otherwise, the mobility controller commands the Roomba to turn for $(TopDir)$ seconds counterclockwise.

In an actual environment, the mobility controller may happen to go to the wrong direction because of noise or radio interference. Therefore, when Roomba goes on wrong direction, the mobility controller needs to revise the direction of movement of Roomba. In order to do so, the mobility controller determines the increase and decrease of RSSI tendency by the exponential moving average (EMA) of the RSSI. The EMA of the RSSI is calculated by equation (1). Every T seconds, the mobility controller calculates the EMA of RSSI and compares it with that of T seconds ago. E_n and R_n describe the EMA and RSSI respectively when the mobility controller observes the nth RSSI. α is a smoothing coefficient.

$$E_n = (1-\alpha)\cdot E_{n-1} + \alpha \cdot R_n \qquad (1)$$

In this experiment, E_1 is equal to R_1. The mobility controller compares E_n with E_{n-5}. It determines that RSSI is increasing when E_n is larger than E_{n-5}, and decreasing otherwise.

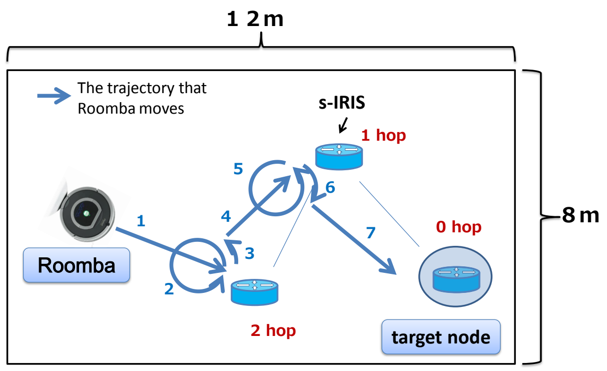

Fig. 5. The deployment of s-IRISs and the mobile sink

d) REACH_TARGET: The mobility controller commands the Roomba to stop for enough time to receive a control message from the target node.

V. Experimental Evaluation

In this section, we show that the Roomba's mobility can be controlled by our proposed method. To show that, we compare the time taken for the Roomba to approach a target node T_{move} when our method is implemented and when it is not. To approach the target node means that the b-IRIS receives a message with RSSI larger than a predefined threshold from the target node. IRIS operates under IEEE 802.15.4 in non-beacon mode, and can asynchronously communicate with other IRIS modules at all times. In this experiment, N is set to 14, α is set to 0.2 and T is set to 5.

A. Outline of experiments

Each IRIS is deployed in a room to construct a multi-hop network. To show the influence of controlled mobility, a single IRIS behaves as the target node. We use the topology shown in Figure 5. The target node IRIS is located at the network boundary. The Roomba starts to move from a specific location, and we measure T_{move}. To measure the efficiency of our algorithm, we compare T_{move} when the Roomba moves along the gradient field and T_{move} when the Roomba moves according to its default algorithm.

B. Experimental environment

The dimensions of the room used for the experiment was 12×8 m. Three IRIS modules were deployed in the room. We removed obstacles from the room to avoid impeding the Roomba's movement. The WSN bandwidth was 2.4 GHz, which overlapped with the bandwidth used for a wireless LAN. We used the 26th channel (2480 MHz), which investigation showed to have the least noise.

In the experiment, the Roomba starts to move after the gradient field is constructed in the WSN. The initial position of the Roomba is the same in each experiment, and experiments start by running a program implemented on the laptop. Figure 6 shows the process flow.

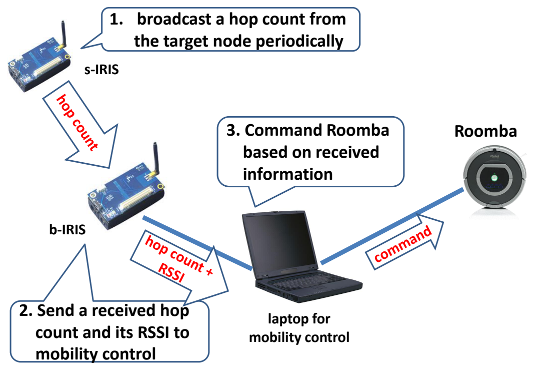
1. broadcast a hop count from the target node periodically
s-IRIS
hop count
3. Command Roomba based on received information
Roomba
b-IRIS
hop count + RSSI
command
2. Send a received hop count and its RSSI to mobility control
laptop for mobility control

Fig. 6. The process flow

TABLE V
EXPERIMENTAL RESULTS

	T_{move}
With controlled mobility	371.12 ± 93.3 s
Without controlled mobility	unsuccessful

C. Experimental result

We limited experiments to 10 min, on the assumption that if the Roomba cannot reach the target node in that time, it never will. Table V shows the results of the experiment. When the Roomba moves according its default algorithm, it cannot approach the target node at all. Using the proposed method, the Roomba reaches target nodes in about 6 min. In a real environment, the effects of shadowing or fading of radio wave are considerable. These effects dynamical change RSSI and therefore, lead the mobile sink into wrong directions. Nevertheless, the mechanism to detect the increase and decrease of the EMA of RSSI allows Roomba to revise the direction of its movement and to approach the target node.

In our simulation, because we adopt the random way point model as the default mobility of the mobile sink, the mobile sink without controlled mobility can reach the target node. However, because the Roomba moves to clean when it is not controlled, it does not necessarily reach the target node.

In terms of overhead of our method, all nodes need to exchange control messages in order to maintain the gradient field at least until a mobile sink reach the target node. However, when monitoring application is assumed and some representative nodes collect sensing information by using a gradient-based routing, additional overhead to control a mobile sink is negligible. In the rescue application, the most important demand is the probability of the guidance of a mobile sink to the target position rather than the overhead of communications.

VI. CONCLUSION AND FUTURE WORK

We proposed a method of controlled mobility using a gradient field. The simulation showed that the proposed method saves the time required for a mobile sink to approach a target node and receive a control message from the target node. We implemented the proposed method on a Roomba in consideration of radio fluctuation in a real environment. We demonstrated that mobile sink mobility can be controlled using a gradient field. The advantage of our proposed method is that any nodes need not to know their own location through the use of devices such as GPS receivers. Moreover, our proposed method can work in indoor environments.

In this paper, we only show the situation that a mobile sink approaches toward the target node gathering sensing data. Our proposed method, however, can be applied to many scenarios; for a patrolling robot or a rescue robot, our method can guide them to the person who requests help. In addition, when a gradient field is constructed according to residual energy, load balancing can be realized.

In future work, we plan to study the case multiple networks exist. In this paper, we use only one network and one target node. Therefore, we need to evaluate whether a mobile sink can approach all the target nodes by rotation when multiple target nodes exist. In this study, we assumed that only one mobile sink exists. We therefore need to consider the case of multiple mobile sinks.

REFERENCES

[1] A. Kansal, M. Rahimi, D. Estrin, W. Kaiser, G. Pottie, and M. Srivastava, "Controlled mobility for sustainable wireless sensor networks," in *Proceedings of the First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, Oct. 2004, pp. 1–6.
[2] S. Basagni, A. Carosi, E. Melachrinoudis, C. Petrioli, and Z. Wang, "Controlled sink mobility for prolonging wireless sensor networks lifetime," *Wireless Networks*, vol. 14, no. 6, Dec. 2008, pp. 831–858.
[3] R. Sugihara and R. K. Gupta, "Improving the data delivery latency in sensor networks with controlled mobility," in *Proceedings of the 4th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Jun. 2008, pp. 386–399.
[4] R. Sugihara and R. Gupta, "Optimizing energy-latency trade-off in sensor networks with controlled mobility," in *Proceedings of the 28th Annual Joint Conference on the IEEE Computer Communications (INFOCOM)*, Apr. 2009, pp. 2566–2570.
[5] F. Mourad, H. Chehade, H. Snoussi, F. Yalaoui, L. Amodeo, and C. Richard, "Controlled mobility sensor networks for target tracking using ant colony optimization," *IEEE Transactions on Mobile Computing (TMC)*, vol. 11, no. 8, Dec. 2012, pp. 1261–1273.
[6] R. Falcon, H. Liu, A. Nayak, and I. Stojmenovic, "Controlled straight mobility and energy-aware routing in robotic wireless sensor networks," in *Proceedings of the 2012 IEEE 8th International Conference on Distributed Computing in Sensor Systems (DCOSS)*, May 2012, pp. 150–157.
[7] J. Hao, Z. Yao, and B. Zhang, "A gradient-based multi-path routing protocol for low duty-cycled wireless sensor networks," in *Proceedings of the 2012 IEEE International Conference on Communications (ICC)*, Jun. 2012, pp. 233–237.
[8] R. Nagel and S. Eichler, "Efficient and realistic mobility and channel modeling for vanet scenarios using omnet++ and inet-framework," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*. ICST, Mar. 2008, pp. 89:1–89:8.
[9] A. Varga, "Omnet++," in *Modeling and Tools for Network Simulation*. Springer, 2010, pp. 35–59.
[10] C. Bettstetter, G. Resta, and P. Santi, "The node distribution of the random waypoint mobility model for wireless ad hoc networks," *IEEE Transactions on Mobile Computing (TMC)*, vol. 2, no. 3, Jul. 2003, pp. 257–269.
[11] "iRobot Roomba® 500 Open Interface (OI) Specification." [Online]. Available: http://www.robotikasklubs.lv/read_write/file/Piemers/iRobot_Roomba_500_Open_Interface_Spec.pdf [retrieved: June, 2013].
[12] "Crossbow MPR-MIB Users Manual." [Online]. Available: http://bullseye.xbow.com:81/Support/Support_pdf_files/MPR-MIB_Series_Users_Manual.pdf [retrieved: June, 2013].