Computer Networks 81 (2015) 116–135

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Large-scale mobile phenomena monitoring with energy-efficiency in wireless sensor networks ☆

Soochang Park [a], Seung-Woo Hong [b], Euisin Lee [c,*], Sang-Ha Kim [d], Noel Crespi [a]

[a] Wireless Networks and Multimedia Services Department, Telecom SudParis, 9 Rue Charles Fourier, Evry Cedex 91011, France
[b] Network OS Research Team, Electronics and Telecommunications Research Institute, 218 Gajeong-ro, Yuseong-gu, Daejeon 305-700, Republic of Korea
[c] School of Information & Communication Engineering, Chungbuk National University, 52 Naesudongro, Seowongu, Cheongju, Chungbuk 362-763, Republic of Korea
[d] Computer Science and Engineering Department, Chungnam National University, 220 Gung-dong, Yuseong-gu, Daejeon 305764, Republic of Korea

ARTICLE INFO

Article history:
Received 15 July 2013
Received in revised form 31 January 2015
Accepted 4 February 2015
Available online 11 February 2015

Keywords:
Continuous objects
Monitoring
Sensing
Energy efficiency

ABSTRACT

In intelligent sensing systems with wireless sensor nodes, energy efficiency is one of the most important research issues. In this paper, we focus on energy efficiency for monitoring a large-scale object such as gas and chemical material diffusion and spread of radioactive contamination and wild fire. For monitoring of a large-scale object, a great number of sensor nodes might be participated in object detection and tracking. Thus, general functions of such huge quantities of sensor nodes like sensing and message exchanging could be sources of energy exhaustion and shorten network lifetime. Therefore, we firstly adopt the sleep/wakeup state switching to restrict active sensor nodes for object tracking. That is, since an object dynamically alter its own shape by wind or geographical condition, we support that only the sensor nodes around the current boundary of the changeable object actively function while the others are on the sleep mode. In addition, we also propose that active nodes are steadily held as a small set of sensor nodes collaborated for detecting and tracking of the current boundary. A variety of computational simulations proves that our proposal is able to provide high energy efficiency as well as to trace accurate boundary shapes.

1. Introduction

Advances in wireless communication and micro-sensor technologies enable to deploy large-scale intelligent sensing systems. Intelligent sensing systems have supported a wide variety of monitoring applications from industrial complexes to smart cities. Event detection and tracking to trace the roaming path of a target object are one of the most typical monitoring applications. Since sensor nodes usually rely on their limited and unattended battery, it is highly important to achieve energy efficiency and thus prolong system lifetime [1–3]. For energy efficiency, sleep/wakeup state switching of sensor nodes is widely used in object tracking studies. It makes only a small set of sensor nodes participate in tracking procedures while the other sensor nodes remain in the sleep mode for energy saving until an object approaches to them. In fact, up to date there are numerous energy-efficient object tracking researches [2–20] using their unique sleep state switching algorithms, and almost researches mainly aim to track small individual objects such as human beings, animals, vehicles, and so on.

☆ This research was supported by the IT R&D program of MSIP (Ministry of Science, ICT and Future Planning)/IITP (Institute for Information & Communications Technology Promotion). [10043380, Development of The High Availability Network Operating System for Supporting Non-Stop Active Routing]. This research was supported by the National Research Foundation of Korea (NRF-2012R1A1A2044460).

* Corresponding author.

http://dx.doi.org/10.1016/j.comnet.2015.02.002
1389-1286/

Recently, the research on monitoring for large-scale objects, denoted by continuous objects, such as gas, wildfire, mud flows, bio-chemical material, oil spill, radioactive contamination, etc., is of increasing concern due to dire calamities happening lately like Japan's nuclear disaster and Greek forest fires. Previous studies on continuous object tracking [21–27] focus on monitoring the current boundary of a large-scale object rather than detecting the entire diffused area of it. Even though they merely track the current boundary, a large number of sensor nodes may join in sensing and communicating to determine the current boundary essentially. Accordingly, the studies have proposed their own cost reduction mechanisms to control communicating based on static/dynamic clustering or representative selection. In other words, they do not adopt the state switching paradigm for high energy efficiency in the case of individual object monitoring.

In order to achieve high energy efficiency and prolong the lifetime of energy-constrained sensor systems, sleep/wakeup state switching by which only small sets of sensor nodes on boundaries are activated should be taken into account. Since a continuous object diffuses widely and changes its shape dynamically, previous state switching methods proposed for an individual object considered as a movable point cannot be applied. Accordingly, a novel stat switching method should be designed for continuous object. In addition, although the novel state switching method is adopted to sensor systems, it should be able to provide adequate boundary tracking accuracy as well as be based on light control signaling. This means that we also try to beat previous sensing and communicating mechanisms to detect boundaries.

Therefore, we propose a energy-efficient and accurate monitoring scheme for a continuous object in this paper. Fig. 1 shows the operation and architecture of the proposed scheme. We first introduce a well-balanced boundary node selection algorithm for low control overhead and high accuracy. This is because detection of a boundary of a continuous object, i.e., the current boundary at this moment, would be the basis to determine the next boundary, i.e., the current boundary at the next time. Then, we are able to start explaining our sleep/wakeup state switching algorithm, named a boundary zone based state switching that offer successive activation of a small set of sensor nodes only around boundaries of a continuous object along its diffusion.

For providing high boundary detection accuracy and light control signaling, our boundary detection algorithm focuses on selection of boundary nodes only closest to an actual boundary of a continuous object. Since previous boundary detection schemes merely rely on communication range to select boundary nodes, they select all the sensor nodes within communication range from a boundary as boundary nodes. It means that a boundary shape drawn from too many boundary nodes selected irregularly might not be correct and node density could influence the boundary shape. On the other hand, our scheme evenly chooses the closest sensor nodes to the current boundary by location-aware boundary node selection and explicit representative arrangement among boundary nodes. Due to this

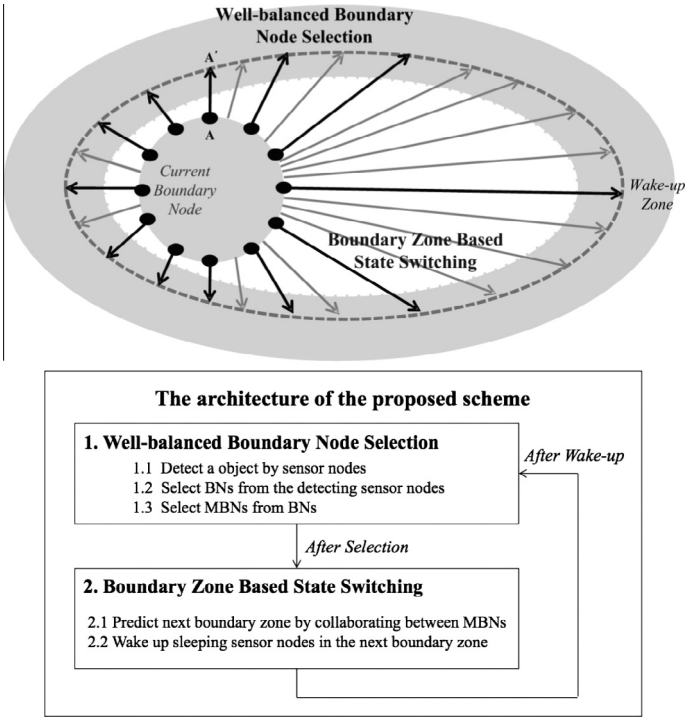


Fig. 1. The operation and architecture of the proposed energy-efficient and accurate monitoring scheme.

sophisticated selection of boundary nodes, boundary detection accuracy is higher than previous works even though a smaller set of sensor nodes is used for boundary detection.

We address a boundary zone based state switching scheme. The switching scheme predicts a future boundary line of a continuous object by measuring diffusing speed and direction of the current boundary line of it. The measurement is performed in boundary zones for representative boundary nodes. Using the predicted boundary line, the switching scheme activates new set of sleeping sensor nodes that will participate in tracking the predicted boundary line at future time. Thus, in this scheme boundary zone allocation and control are the most important part since a continuous object dynamically spread out.

The rest of this paper is organized as follows. Section 2 offers comparative literature reviews regarding energy consumption of divers sensor devices and object tracking applications. Section 3 and Section 4 explain our boundary node detection algorithm and predictive tracking scheme with state switching including problem analysis of previous works, respectively. In Section 5 and 6, we examine performances of our proposals by computational simulations. Section 7 conclude the paper.

2. Reviews of the relevant literature

In this section, we review characteristics of object tracking applications with comparing between individual objects and continuous objects. In addition, we provide energy consumption values along state switching regarding sensor devices that have been developed and used for intelligent sensing systems.

2.1. Object tracking applications

Object tracking is to follow a moving course or variation of a mobile object, and it is one of principal applications for intelligent sensing systems in which a network of wireless sensors is assigned the task of tracking a particular object. The network employs object tracking techniques to continuously report the current position of the object, i.e., coordinates data, to a sink node or to a central base station.

Target objects, which user is interested in, can be divided into two classes: *individual objects* and *continuous objects*. Individual objects have usually small sizes comparing with the large area of sensor nodes deployed, e.g., tanks, animals, soldiers and so on. Continuous objects originate at one source spot but consecutively and widely spread out on sensor fields, e.g., radioactive contamination, wildfire, oil spill, poison gas, etc. They have irregular boundary and grow dynamically. Although continuous objects might be specific mobile phenomena in object tracking area, their tracking are very important as well as common because most of them are occurred at emergency situations in major facilities. Table 1 shows difference between an individual object and a continuous object.

Table 1
Individual objects versus continuous objects.

Item	Individual object	Continuous object
Display	Dots	Irregular Polygons
Object Size	Very Small	Very Large
Territory	Narrow Area	Wide Area
Sensing	Some Sensors	Large # of Sensors
Effects	Very Restrictive	Very Sensitive
Movement	Relatively Uniform	Dynamic

2.1.1. Schemes for individual object tracking

Individual object tracking might be one of very common applications in wireless sensor networks. However, significant feature of the network is the low-power consumption requirement. Therefore, many methods have been proposed to solve the problem. Object tracking have two critical operations, monitoring and reporting. Hence, we can divide the methods into *communication cost control* schemes for reducing the number of reporting and *sleep-wakeup mode switching* schemes for reducing the energy consumption in monitoring operation. Thus, we explains the communication cost control schemes and the sleep-wakeup mode switching schemes.

2.1.1.1. Communication cost control schemes. In tree-based target tracking, nodes in a network may be organized in a hierarchical tree or represented as a graph in which vertices represent sensor nodes and edges are links between nodes that can directly communicate with each other. Examples of tree-based methods include Scalable Tracking Using Networked Sensors (STUN) [7], Deviation-Avoidance Tree (DAT) [8,9] and Dynamic Convoy-Tree-based Collaboration (DCTC) [10,11]. In this case a tree structure is maintained across the network. The tree is rooted at the node that is closest to the target. Thus as the target moves some nodes get added to the tree and some get deleted. This scheme reduces the overhead in terms of energy and information flow, as the information flows from the root to the end or periphery of the network through a particular route, as the information flows is controlled so energy consumption automatically gets controlled [12].

Prediction-based methods are based on the fact that the movement of a tracked object is sometimes predictable. Using the predicted direction and velocity of moving target, tracking algorithm can estimate a next location of the object. According to the prediction, a sensor node transmits only subset of location information or may not send any data. Examples of prediction-based mechanisms are PREdiction-based MONitoring (PREMON) [17–19] and Dual Prediction-based Reporting mechanism (DPR) [17].

2.1.1.2. Sleep/wakeup mode switching schemes. While communication cost schemes contribute in term of reporting for energy-efficient tracking, sleep/wakeup mode switching schemes are conducive to effective monitoring. Most sensor nodes can sleep and wake up for energy saving, if they are necessary. Through this mechanism, only subset of all sensor nodes activates to maximize energy-utilization. However, if all of sensor nodes are in sleep mode, target object can not detected at right time. Therefore, only minimal number of sensor nodes should activate, or sensor nodes, where the object will arrive, should be awaken. We define the former method as reactive scheme, and the latter method as proactive scheme.

Most of schemes using the sleep/wakeup mode switching strategy employ cluster-based methods for awakening per cluster group. In such cluster based architecture, there are several sensor nodes and for a certain group of nodes, they are assigned a cluster leader or cluster head [12]. Examples of cluster-based methods include [13–16]. The network is formed with powerful cluster head nodes and low-end sensor nodes. The cluster head node which has the strongest sensed signal of target becomes the cluster head, and organizes nearby sensor nodes to form a cluster. The sensor nodes in the cluster transmit the sensed data to the cluster head, and then the cluster head sends the digested data to the base station after data aggregation. In order to prevent missing the target object, all cluster head nodes need to monitor continuously. Since only the cluster head nodes can be the cluster head, object tracking becomes very difficult in some areas where the cluster head nodes are sparsely distributed. And after a cluster is formed, there is no rotation of the cluster head, so the higher speed the target object travels at, the higher energy the cluster head would consume. So in this scheme, the energy consumption among the cluster head nodes is unbalanced, and thus reduces the network lifetime [15].

2.1.2. Schemes for continuous object tracking

Strategy for tracking a continuous object might be different with tracking an individual object, because they have each other dissimilar features. A continuous object has large scale and change irregularly its shape, and so users are interested in its current shape as well as location. Boundary information is enough to know its shape and location. Therefore, continuous object tracking focus on tracking the boundary. In order to detect the continuously moving boundary of continuous objects, it usually required a large number of detection message exchanges between sensor nodes. Therefore, most of studies concentrate on reducing the communication cost through the report message aggregation based on cluster structure and boundary approximation with reducing the number of boundary nodes. On the other hand, recently, schemes applying a sleep/wakeup mode switching to continuous object tracking are proposed. Thus, we introduce the communication cost control schemes, and then sleep/wakeup mode switching schemes for energy-efficient continuous object tracking.

2.1.2.1. Communication cost control schemes. Communication cost control schemes focus on reducing the number of reports to transmit to sinks. We divide schemes to reduce the communication cost into cluster based schemes and representative selection schemes. Cluster based schemes rely on the clustering the boundary nodes. Representative based schemes try to reduce the number of reported boundary nodes without clustering.

DCS (Dynamic Cluster Structure) [21] proposes a dynamic cluster based algorithm that tracks the movement of continuous objects by monitoring the boundary of those objects. In the DCS, when a sensor detects the emergence of any phenomena at current time, it immediately broadcasts a query message to its neighbors to ask for the neighbors' readings and the neighbors reply by sending their current readings to the sensor. If the sensor receives at least one different detection status from any neighbors, the sensor becomes a Boundary Node (BN). After the boundary node selection, cluster formation process takes place among the BNs. To reduce the report message, cluster head aggregates the report message in its vicinity. In each cluster, boundary information are aggregated. Although the DCS approach provides substantial energy savings, it still requires a significant volume of communications since the boundary sensors are identified by requiring each sensor which detects the emergence of the object to communicate with all of its one-hop neighbors to determine whether or not they too detect the same object [22].

The static clustering approach for continuous objects detection and tracking is proposed in a Continuous Object Detection and tracking Algorithm (CODA) [22]. CODA proactively establishes static clusters of all sensor nodes on a whole sensor field and then a sensor node in a static cluster is selected as the head node of the cluster. When a continuous object appears, the cluster head gathers data about the boundary information from the boundary nodes in the area of its own cluster and reports the data to a sink. So, the sink is able to detect the entire boundary of the continuous object by using all data from all clusters.

Two-tier Grid based Continuous Object Detection and tracking (TG-COD) [23] takes into account a two-tier grid structure in order to achieve flexible and reliable detection and tracking. Firstly, for flexibility, the scheme proactively constructs a coarse-grained grid structure and then, once a continuous object appears, fine-grained grid structures are reactively established within coarse-grained grid cells only around the continuous object. To quickly deal with diffusion of the continuous object the scheme prepares the fine-grained grid structures into next coarse-grained grid cells toward diffusion direction of the continuous object. Additionally, in order to achieve reducing cluster organizing cost, TG-COD relies on a grid structure that is simply established by location information of a reference point and a grid cell size value on the assumption that sensor nodes are aware of their own locations. Thus, by the grid structure TG-COD could offer rapid structure termination and re-establishment according to movement or alteration of continuous objects [23]. Nevertheless, measurement of diffusing direction and speed is not defined.

Regarding the representative selection approach, there have been a series of continuous object tracking studies which are initiated from [24,25]. Their claim is that cluster-based scheme is not suitable for tracking the boundary of a continuous object because of its inefficient cluster formation overhead. We simply call such a series of studies as BN-array based algorithm as they use BN-array to detect the boundary of a continuous object.

In [24,25], COBOM, an energy-efficient algorithm for boundary detection and monitoring is proposed. Although more BNs are selected than DCS [21], the number of nodes that actually report to the sink are Representative Nodes (RNs), and RNs is much less than DCS. When a sensor's current reading is different from the one previously observed, the sensor broadcasts its reading and ID. A node that receives the reading and ID stores the received reading into its array (called BN-array) and if the node finds that

any different reading exists in the BN-array, then the sensor becomes a BN. Among those boudary nodes (BNs), a few Representative Node (RNs) will be selected. The more number of different detection status a BN has in its array, the more likely it becomes a RN that eventually reports all the gathered detection result to the sink. This algorithm is energy-efficient in a way such that first, only a few representative nodes will be chosen and second, by using the BN-array, the report message size would not increase considerably since each message contains all of its neighbors' detection status as bits only rather than keeping the neighbors' IDs also, which requires only few bits while the precision of boundary estimation might be guaranteed.

DEMOCO [26] focuses on reducing the number of BNs that eventually results in lowering the number of RNs to reduce traffic as well as communication between sensor nodes. DEMOCO has improved [24,25] by just considering nodes in "IN" range, ignoring those in "OUT" range which theoretically reduce half of selecting BNs and RNs so as to achieve energy saving as well as prolong network lifetime. EUCOW [27] is only focusing on monitoring unsmoothed a continuous object and introduce the "genEvent algorithm" to theoretically generate the expected event.

2.2. Energy consumption of sensor devices

A wireless sensor node, being a microelectronic device, might be typically tiny and low-cost. This may need to fit into a matchbox-sized module [31–33]. In some extreme cases, an entire sensor node could have lighter weight than 100 g, cheaper than US$ 1, and dissipate less than 100 μW [34]. However, despite of constraint of such size and cost, sensor nodes must consume extremely low power.

The sensor node can only be equipped with a limited power source (<0.5 Ah, 1.2 V). In some application scenarios, replacement of power resource might be impossible. Therefore, sensor node lifetime shows a strong dependence on battery lifetime. Additionally, in a multihop ad hoc sensor network, each node plays the dual role of data originator and data router. The malfunctioning of a few nodes can cause significant topological changes and might require rerouting of packets and reorganization of the network. Hence, power conservation and power management take on additional importance. Namely, energy efficiency is the most critical factor to prolong the lifetime of intelligent sensing systems.

To save energy, most sensor nodes support a *sleep-wakeup mode switching function*. In other words, they can power down (put to sleep) or power up (wake up) if necessary. Table 2 shows examples of sensor nodes that support a sleep-wakeup mode switching. Rockwell's WINS node [35], which represents a high-end sensor node and is equipped with a powerful StrongARM SA-1100 processor from the Intel, a radio module from the Conexant Systems, and several sensors including acoustic and seismic ones. Transmission (Tr.) power of a WINS node is from 771.1 mW to 1080.5 mW according to several radio mode. Receive (Rc.) power of a WINS node is 751.6 mW. When a Micro Control Unit (MCU) of the WINS node is slept, power consumption is 64.0 mW. MEDUSA-II [36] is an experimental sensor node developed at the Networked and Embedded Systems Lab, UCLA. The MEDUSA node, designed to be ultra-low power, is a low-end sensor node similar to the COTS Motes developed as part of the SmartDust project [37]. It is equipped with an AVR microcontroller from ATMEL, a low-end RFM radio module, and a few sensors. A MEDUSA-II node's Transmission (Tr.) power is from 19.24 mW to 27.46 mW according to radio modes, mode schemes and data rates. Receive (Rc.) power of it is 22.20 mW. When a MCU of the MEDUSA-II is sleep, and sensor as well as radio are off state, power consumption is 0.02 mW. Telos [38] is an ultra low power wireless sensor module developed by UC Berkeley, using a Texas Instruments MSP430 microcontroller, Chipcon IEEE 802.15.4-compliant radio, and USB. When a Telos node use O-QPSK as modulation type, Transmission (Tr.) power and Receive (Rc.) power of it is each 35 mW and 38 mW. When a MCU of a Telos node is in sleep state, its power consumption is 0.015 mW. IRIS [39] is the product from Crossbow Technology, which operates TinyOS. The IRIS node uses AT86RF230 radio, which is 2.4 GHz radio transceiver compatible with IEEE 802.15.4–2003 standard. An IRIS node has 75mW and 63mW as its Transmission (Tr.) power and Receive (Rc.) power, respectively. When a MCU of an IRIS node is in sleep state, its power consumption is 0.036mW.

3. Well-balanced boundary node selection

This section addresses problems of previous works in terms of continuous object tracking, and then explains our algorithm for high boundary detection accuracy and low control overhead, named well-balanced boundary node selection.

3.1. Historical perspectives: continuous object tracking

As a continuous object diffuses in the wide region unlike a single object, it is not suitable to collect sensing data from every sensor nodes in the entire diffused area,

Table 2
Power consumption of sensor devices.

States	Sensor devices			
	WINS	MEDUSA-II (mW)	Telos (mW)	IRIS (mW)
Tr.	771.1–1080.5	19.24–27.46	35	75
Rc.	751.6	22.20	38	63
Active	727.5	22.06	3	31
Sleep	64.0	0.02	0.015	0.036

and in many case of application scenarios, just monitoring the boundary of continuous objects is often sufficient [21–27]. For example, in the forest fire monitoring, we may be interested in the location and expanding direction of major boundaries of fires. In order to detect a boundary of a continuous object, every sensor node sets some threshold for event detection. If the sensing value of a node is larger than the threshold, the node checks status of its neighbor nodes. If some nodes sense lower values yet, it becomes a Boundary Node (BN). If all neighbor nodes were over already, it means the node is located inside a continuous object. All the BNs would report sensing data with their locations as boundary information to a sink. The straightforward approach to data collection is to let all BNs periodically report their sensing to a sink, and the sink can construct the view of whole boundary line based on reported data. Herein is still such a problem that if the boundary is long or the node density is high, too many BNs would detect it and then send a large quantity of data. Such quantities of data will consume large amounts of network energy and may also cause unnecessary traffic and packet loss.

To solve this problem, there could be two solutions. The first one is to use clustering and message aggregation for reducing the report messages [21–23]. If we divide the entire boundary detection area into some small portions of regions, called clusters, one of BNs, called a cluster head, in each cluster can aggregate the boundary readings in its vicinity. Then only the cluster header reports aggregated data to a sink. As a result of message aggregation, the number of reports could be reduced in proportion to the provisioned cluster size. This approach takes effect on energy saving by reducing long transmission communication cost. However, we cannot disregard the overhead of cluster formation. For instance, if a cluster is formed dynamically at the time of each boundary detection period like [21], additional local messages should be sent to decide a cluster zone and a cluster header in the predefined size of the region. This signaling could be another reason of energy exhaustion. If static cluster formation [22,23] can be used rather than dynamic clustering, it could reduce the clustering overhead by pre-formatting clusters at the initial deployment time. But static clustering has the scalability drawback when network topology changes caused by supplement or elimination of sensor nodes. Namely, hidden signaling overhead for maintaining proactive clusters could be higher due to network dynamics.

The second solution is to reduce the number of BNs. For tracking a large-scale continuous object, Boundary Nodes (BNs) that refer to sensor nodes locating on the current boundary of an object periodically report their own location to a sink, and the sink can construct a view of whole boundary line by combining reported locations. That is, a large quantity of data may be generated; thus, it causes high energy consumption. Hence, studies [24–27] mainly focus on reducing the number of reporting nodes toward a sink through selecting Representative Nodes (RNs) among BNs.

However, since the previous studies elect BNs by merely the communication range, the density of deployed sensor nodes and the number of BNs directly influence the number of RNs as well as RNs could be irregularly allocated, as shown in Fig. 2(a). In other words, node density and BN selection methods is deeply related to performances of a continuous object tracking application requested by energy-efficiency and boundary accuracy.

In this section, we propose a novel BN selection algorithm that makes a small set of BNs independent of node density and provides well-balanced RN selection from the small set. Unlike previous studies, the proposed algorithm chooses only the closest sensor nodes to the current boundary of an object as BNs. For this, we newly design a Neighbor Descriptor Table (NDP) that includes neighbor location information and event detection status. From the sophisticatedly selected BNs, the algorithm tries to choose only one RN per each radio range regardless of node density as shown in Fig. 2 (b).

3.2. Network model and definitions

In this subsection, we will describe the proposed boundary detection algorithm in detail. First of all, we give a general assumption of sensing systems that each sensor node knows its own location by possibly using the global positioning system (GPS) [28] or other techniques such as triangulation [29] or location services [30].

To clarify the algorithm descriptions, we have defined the required components that are used by boundary and master node selection process.

- *Nu*: Let u be a node. *Nu* represents neighbor nodes of u. the neighbor nodes *Nu* are located within communication range r. If a node u and one of *Nu* are in different detection status, there exists a boundary between them.
- *NuD[Nu-ID, l, b, t]*: It represents the descriptor of a *Nu* which is a neighbor node of node u, *Nu-ID* is unique *ID* of *Nu*, s is current detection status, l is location of *Nu*, b means boundary node flag, and t is timestamp of event detection.
- *NDT[ordered-set(NuD, d)]*: It is a neighbor descriptor table. Each node u should maintain the *NDT*, which is the basis for boundary node selection. d is the distance between node u and *Nu*.
- *evt-msg(ID, s)*: Any node u who detects the emergence or disappearance of a object should report the event message. The message includes the *ID* of v, which is one of *Nu* and current detection status s.
- *bn-cond(R1, R2)*: It denotes the condition for a boundary node, which has two rules *R1* and *R2*. A node u becomes a boundary node if it is satisfied with these two rule as blow;
  - *R1*: Status s of node $u \neq$ status s of received *evt-msg(ID, s)*
  - *R2*: The distance $\overline{vu}$ is shorter than any other distances $\overline{vNu}$
- *BN*: a Boundary Node (BN) denotes a sensor node which detects a continuous object and is selected through bn-cond(R1, R2). Only BNs could participate in the process for being selected as Master Boundary Nodes (MBNs).
- *MBN*: a Master Boundary Node (MBN) denotes a BN which is selected from BNs by signaling among them. A MBN sends reporting data about the detected

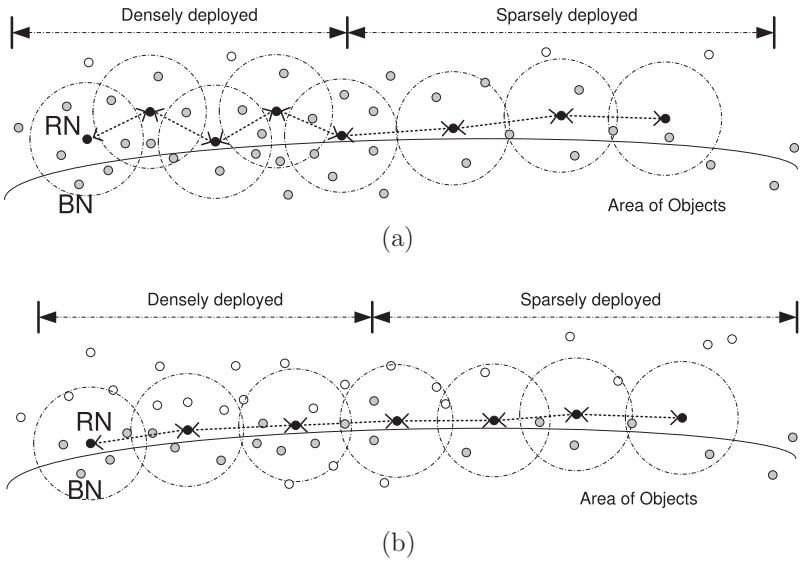


Fig. 2. RN selections: (a) irregular selection of RNs and (b) well-balanced and distributed selection of RNs.

continuous object to a sink node and thus is same as a Representative Node (RN) of previous studies.

- *SBN*: a Slave Boundary Node (SBN) denotes a BN which is also selected among BNs and assists the MBN selection process for selecting another MBN by signaling with MBNs.

In this algorithm, there are four main process functions; initial processing, boundary election processing, master boundary node biding processing and slave boundary node election processing. Major functionality of the initial processing is to create the *NDT[ordered-set(NuD, d)]* by querying to all neighbors from each other. When a sensor node receives the *evt-msg(ID, s)* from detected neighbors, it immediately begins the boundary election processing, in which each sensor node independently decides whether it could become boundary node or not. If any node becomes the boundary node, it starts to bid in order to become the MBN, and the winner will become MBN. The MBN begins to select consecutive SBNs and neighbor MBNs in sequence. Consequently, all MBNs reports sensing data to sink node. More details will be described in subsection.

3.3. Boundary node selection process

The boundary node selection process mainly depends on the *NDT[set(NuD, d)]*. When a sensor node is deployed initially, the sensor node does not have any information about its neighbors. Then, we start from the initial construction process for *NDT[set(NuD, d)]*. The sequence of boundary node selection will be described step by step.

1. *busy-sensing() step*: As a sensor node u is deployed in sensor field, it first checks a initialized flag, If the flag is clear, then it broadcasts a *inform-neighbor-msg(ID, l, b, t)*, which includes its *ID*, location coordinates *l*, boundary node flag *b*, and timestamp *t*.
2. *tde-update() step*: If a sensor node u receives the *inform-neighbor-msg(ID, l, b, t)* from any *Nu*, it calculates *l*, which is the distance between itself and *Nu*, and it updates its *NDT[set(NuD, d)]*. The distance can easily be calculated by Euclidean distance.
3. *evt-msg(ID, s) broadcast and listening step*: A sensor node u should report the *evt-msg(ID, s)* with its *ID* and detection stats *s*, when the detection status of previous time *t-1* and the present time *t* is different.
4. *BNs election with bn-cond(R1, R2) step*: If a sensor node u received the *evt-msg(ID, s)* from neighor node v, it first updates *NuD* of v and checks the *bn-cond(R1, R2)* weather it can become BN. If the condition is not satisfied, it ignores the event message. Otherwise the sensor node u becomes BN and it counts the number of receiving *evt-msg* during the given time, which will be usefully used during the master node election bid time.

Currently, only sensor nodes located in the nearest to continuous objects boundaries become the BNs. Fig. 3 shows the sequence charts of proposed boundary node election process.

3.4. Master boundary node selection process

Fig. 4 shows the sequence charts of the proposed MBN and SBN selection nodes process.

In order to elect a MBN, when the boundary node election process is done, every elected boundary node begins the MBN election bid to become a MBN. A MBN is firstly elected based on the received event messages, and the explicit MBN and SBN election process is propagated to left and right boundary nodes. Initial selection for MBNs depends on the number of *evt-msg(ID, s)*, and the process sequence for master boundary node selection is presents as blow:

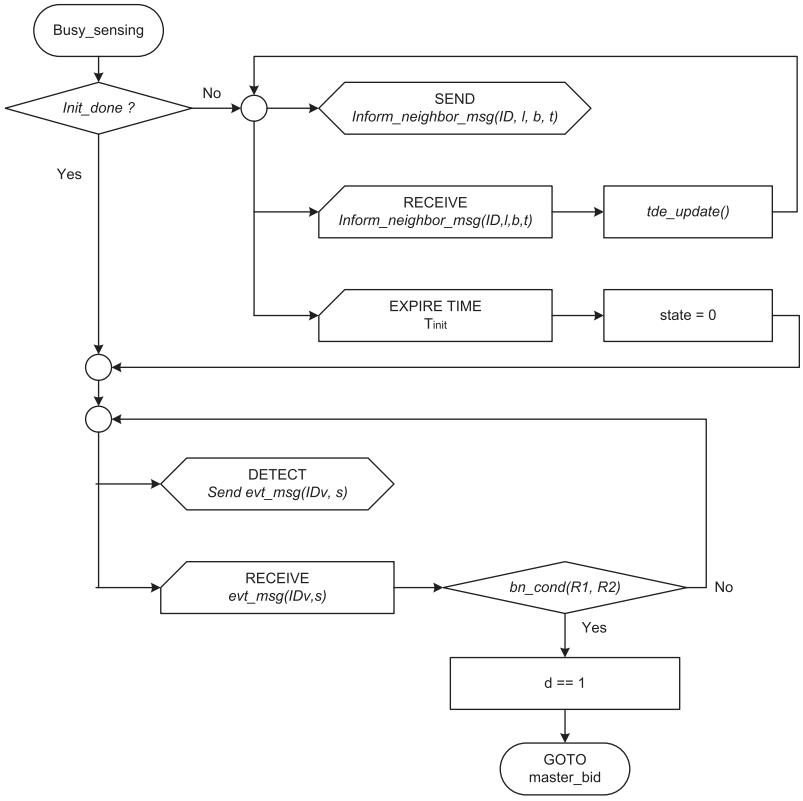


Fig. 3. BN election process.

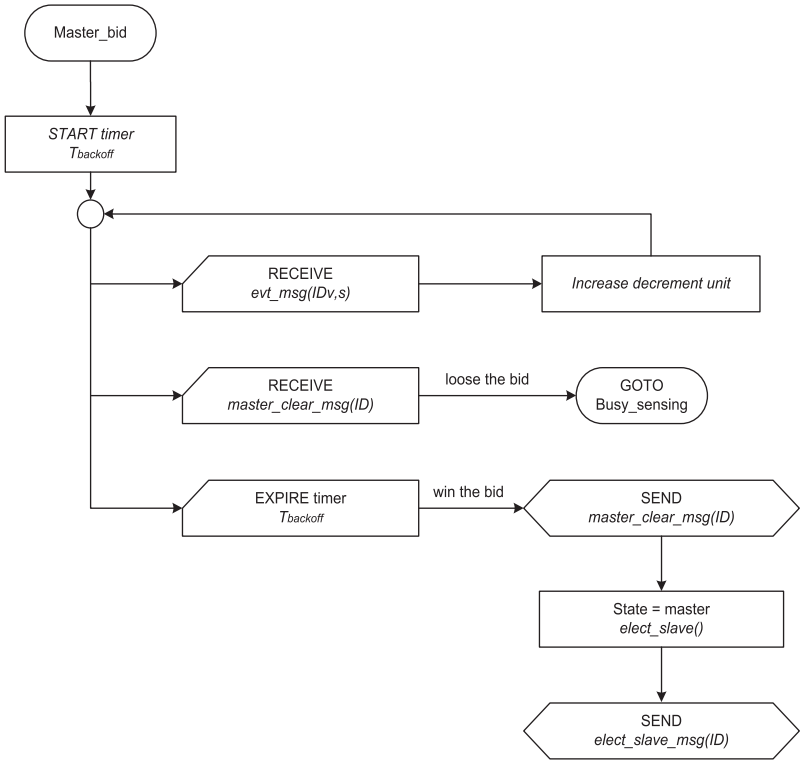


Fig. 4. MBN and SBN selection process.

1. *start-timer(r) and increase-dec-rate(C) step*: Elected BNs start timer, and starts master bid process. Let current time as T_c, initial timer as T_b, report threshold as T_r, then $T_b < T_r - T_c$, and let decrement rate as R_d, the number of *evt-msg(ID, s)* as C, then $T_b = T_b - R \times C$.
2. *master-clear-msg(ID) step*: If the T_b of the node u is expired, u becomes a MBN, and it broadcasts the *master-clear-msg(ID)* with its own *ID*.
3. *elect-slave() step*: The MBN u starts the recursive MBNs selection process by selecting the left SBN l and right SBN r. Let the location of u as (x_u, y_u), and the location of u's neighbor BN_i as (x_i, y_i), then u firstly finds any node s among BN_i, which has the maximum euclidian distance to the MBN u as blow:

$$\overline{ui} = max(\sqrt{(x_s - x_u)^2 + (y_s - y_u)^2}). \quad (1)$$

If the angle $\theta_{\overline{us}}$ between u and s is in $-\pi/2 < \theta_{\overline{us}} < \pi/2$, then s becomes the right SBN and the right SBN r vice versa. l can be found in remaining BN_i that is located in $\pi/2 < \theta_{\overline{ul}} < -\pi/2$. The angle $\theta_{\overline{ui}}$ between u and BN_i can get by:

$$\theta_{\overline{ui}} = cos^{-1} \frac{x_u - x_i}{\sqrt{(x_u - x_i)^2 + (y_u - y_i)^2}}. \quad (2)$$

4. Boundary zone based state switching

Basic motivation of our state switching algorithm is to only activates the sensor nodes near the current boundary of a continuous object while the other sensor nodes are far away from the boundary in the sleep mode for energy saving. For this selective wakeup approach, a set of sensor nodes should be activated to detect and report the current boundary at a time $T1$. At the next time $T2$, the previous set of sensor nodes would go back to the sleep mode, and another set of sensor nodes which located around the next boundary, i.e., the current boundary diffused until the next time, needs to be activated. Of course, before the previous set of sensor nodes go to the sleep mode, they should estimate the next boundary line and send a wakeup signal to sensor nodes around the estimated line. Therefore, there is always one set of activated sensor nodes located in vicinity of the boundary line at each certain time and the set of sensor nodes fulfills boundary monitoring. We define this region as *boundary zone*, denoted by *BZ(t)*.

4.1. Conceptual perspectives: selective state switching

The key problem of our state switching algorithm is how *BZ(t)* estimates *BZ(t + Δt)*; Δt means a certain period for object tracing procedures, e.g., Boundary Node (BN) decision and Master Boundary Node (MBN) selection. A boundary line is recognized by linked MBNs, namely, the current boundary line *bl(t)* is the concatenated line of all MBNs $\{mbn_1(t), mbn_2(t), mbn_3(t), \ldots, \text{mbn}_n(t)\}$, where n is the number of total MBNs in *BZ(t)*. If $mbn_i(t)$ can possibly estimate a location of a virtual MBN at the next time $t + \Delta t$, denoted by $vbn_i(t + \Delta t)$, the virtual boundary line *vbl(t + Δt)* can be estimated using all the $vbn_i(t + \Delta t)$, and then we can define a set of sensor nodes located around the *vbl(t + Δt)* as *BZ(t + Δt)*.

In order to realize this concept, each $mbn_i(t)$ measures diffusing direction and speed of the boundary line at time t and estimates a location of its corresponding $vbn_i(t + \Delta t)$. Then, $mbn_i(t)$ and $mbn_{i+1}(t)$ collaboratively decide a certain zone between $vbn_i(t + \Delta t)$ and $vbn_{i+1}(t + \Delta t)$.

Fig. 5(a) presents an example of diffusing zones, in which the current MBNs, A and B, estimate two *vbn(t + 1)* of A and *vbn(t + 1)* of B using their own local diffusing direction and velocity respectively, where t is the current time and periodic time is 1. After that, A and B estimate a certain range of the diffusing zone based on estimated two *vbn(t + 1)* of A and *vbn(t + 1)* of B. Since B has another adjacent MBN C, it should also cooperates with C to estimates the diffusing zone by using each of estimated virtual MBNs. If the diffusing zones are estimated by all current $mbn_i(t)$ in *BZ(t)*, all of those nested diffusing zone eventually become the *BZ(t + 1)*.

In case of Fig. 5(a), we present the ideal case of diffusing boundary line between MBNs A, B and C, which expand same direction and same speed at all of MBNs. But in most cases of real world, diffusing boundary line generally takes the shape of irregular line rather than flat line as shown in Fig. 5(b). Because characteristics of continuous objects expand to all direction and diffuse with different diffusing speed to each direction, every MBN surely has different diffusing direction and speeds in its boundary location. The local diffusing measurement and estimation method for diffusing zone will be described in the next subsections.

4.2. Network model and definitions

We consider the network model with following assumptions in order to support our state switching algorithm.

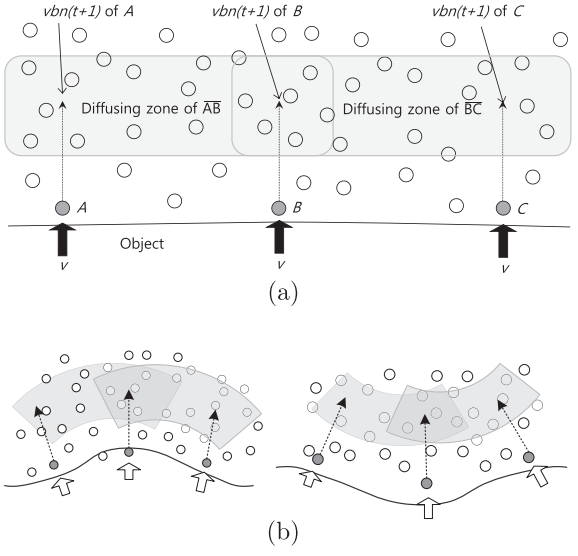


Fig. 5. *BZ(t + Δt)* estimation: (a) diffusing zone of *AB* and *BC* and (b) diffusing zone of *AB* and *BC*.

– All sensor nodes could have wakeup capability in radio communication ranges using the ultra-low power MAC [40] or specialized hardware to serve as a wakeup paging receiver such as PicoRadio [41].
– All sensor nodes might use synchronized time, and the clock-drift among the sensors is small enough to be negligible. Several well-known reference broadcast synchronization [42] and delay measurement time synchronization [43] can meet this requirement.

Based on such assumptions, we define five features of states for a sensor node as follows:

- *Sleep-White*: Sensor nodes stay in this state when they are initially deployed in a sensor filed or they have not been notified about coming target object. Sensors keep the sleep mode, and various initial detection schemes such as sentry-based mechanisms can be used to initially detect a target object.
- *Active-White*: Sensor nodes switch into this state when they are activated by the previous boundary zone at the previous time (the current time – T). In general, nodes in this state can sleep more $T - X$ time, where X is protocol processing time. But, it can be awake earlier in order to recover error.
- *Active-Test*: Sensor nodes in the white or the black active state get into this state when the process for recognizing the target object is expired. They perform normal signaling protocol, which consists of two steps; First, boundary detection, and second, estimation of the diffusing zone and wakeup nodes in the zone.
- *Sleep-Black*: After a sensor node in the active-test performs signaling tasks, it can change to this state in case of when the target is detected by all neighbors. The case means that the sensor node is currently within the tracked object, then, it keeps in the sleep state until the boundary is close to sensing range.
- *Active-Black*: Sensor nodes in sleep-black switch into this state when they are activated from previous activated sensors. The operation is similar to active-white.

To clarify these descriptions, we have defined the required messages. Three types of wakeup messages are defined to activate sleeping nodes as blow:

- *wakeup-msg(v, p, l)*: This message has a *vbn(t + Δt)*'s location *v*, wakeup parameters *p*, which are explained in the next subsection, and a *mbn(t)*'s location *l*. When a sleeping node received this message checks *v*, if it is the closest node to *v*, it can be the *vbn* and activate itself.
- *wakeup-neighbors-msg(f)*: Parameter *f* represents a flag that identifies 'normal wakeup' or 'urgent wakeup'. A activated *vbn* should selectively awake its neighbors according to wakeup parameter *p* from *wakeup-msg(v, p, l)*. The *vbn* broadcasts 'normal wakeup' message to all 1-hop neighbors or unicasts it to chosen neighbors. On the other hand, when a continuous object initially appears, and a sentry node for the initial time detects it, the sentry node broadcasts 'urgent wakeup' message to 1-hop neighbors. If sensing status of the awaken node is true, it relays this message. If not, it stops flooding since it means that activation is reached the current boundary.

4.3. Local diffusing measurement

Basic model of our state switching algorithm is the switching-over boundary detection mission between current boundary zone *BZ(t)* and next boundary zone *BZ(t + Δt)*. This switching-over task is recursively performed every time period during tracking of continuous objects. As a result, only single *BZ(t)* exists in every sensing time. *BZ(t + Δt)* consists of a set of diffusing zones which are estimated by each pair of two adjacent MBNs in the *BZ(t)*. To estimate the diffusing zones, each pair of MBNs should measure the diffusing direction and velocity in their place.

Each of MBNs measures its location diffusing direction. Since one sensor node is geographically considered as single point when it is elected as a MBN in 1-hop ranges, it cannot decide which direction the continuous objects are coming from. In order to solve the problem, we can imagine the shape of a boundary line which connects all of MBNs. Though the real shape of continuous objects generally is irregular, the measured boundary line using sensor nodes actually takes the shape of polygon. The MBNs are represented as vertices and the connection line between adjacent two of MBNs are regarded as edges. Then, we can have a triangle that consists of three MBNs and its two edges have the same length(radio range $r \times 2$), because our boundary election algorithm chooses only one MBN in 1-hop radio range.

Fig. 6(a) shows the example of described triangle, *B* can measures its diffusing direction by forming a triangle with its two adjacent neighbors *A* and *B*. The distance between *B* and *A* is same to the distance between *B* and *C*. If we can estimates the middle point, denoted by *P*, of edge between *A* and *B*, the angle of diffusing direction of *B* can be estimated through two points *B* and *P*. Fig. 6(b) shows more detailed example to explain how to estimates diffusing angle of *A*. Since *D*, *A* and *B* have its own coordinates, we can estimates the coordinates of *P* using vector equation.

Let $\vec{a}$ be the vector between *D* and *A*, and let $\vec{b}$ be the vector between *D* and *B*, then, we can get the vector $\vec{p}$ by projecting the $\vec{a}$ to $\vec{b}$ with following equation:

$$Proj_{\vec{b}}\vec{a} = \left(\frac{\vec{a}\cdot\vec{b}}{|\vec{b}|^2}\right). \tag{3}$$

The coordinates of *P* can be estimates as:

$$(x_p, y_p) = \left[\frac{x_{i+1}\times x_i + y_{i+1}\times y_i}{(x_{i+1}-x_{i-1})^2 + (y_{i+1}-y_{i-1})^2}\right](x_i, y_i), \tag{4}$$

then, the diffusing direction θ_i of *A* is got by:

$$\theta_i = cos^{-1}\frac{x_i - x_p}{\sqrt{(x_i-x_p)^2 + (y_i-y_p)^2}}. \tag{5}$$

Suppose θ_n is the direction from a *A*'s neighbor *N* who detects the continuous object in *NDT[set(NuD, d)]*, *A* should check the following condition:

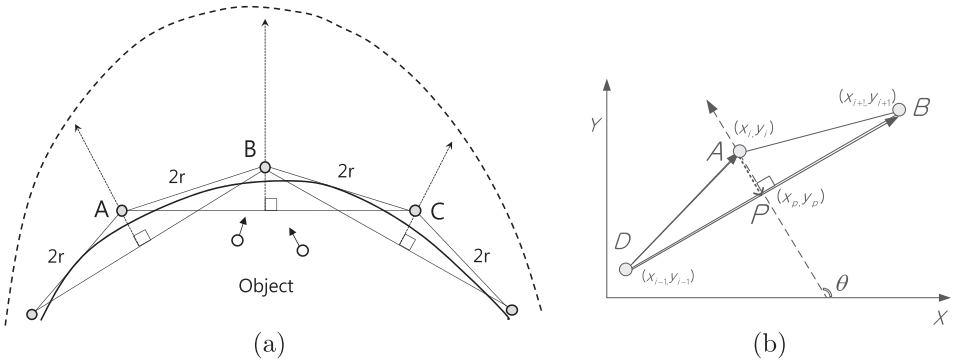


Fig. 6. Diffusing direction estimation.

$$if(\theta_i - \theta_n) > \frac{\pi}{2}, \tag{6}$$

$$\theta_i = \theta_i + \pi, \tag{7}$$

if *A*'s triangle forms a reserve triangle, θ_i points out the opposite side of diffusing direction, and every boundary node should check the Eq. (6).

In our scheme, a *BZ(t)* activates a *BZ(t + Δt)* in time *t*, and *BZ(t)* was activated from a set of MBNs in *BZ(t − Δt)*. When a MBN actives the next diffusing zone, it includes its location and detection time in wakeup message. Therefore, activated sensor nodes by wakeup messages can calculates the diffusing velocity using previous location, previous detection time, current location, and current detection time.

Suppose that a *mbn(t − Δt)* is the previous MBN who activates the diffusing zone *DZ(t)*, and a *mbn(t)* is the current MBN in *DZ(t)*. Also, let (x_{t-1}, y_{t-1}) be the location of *mbn(t − Δt)*, and let (x_t, y_t) be the location of *mbn(t)*. Let *t* be the current detection time and $t-1$ be the previous detection time. Then, the diffusing velocity of *bn(t)* can be calculated as blow:

$$v = \frac{\sqrt{(x_t - x_{t-1})^2 + (y_t - y_{t-1})^2}}{t_t - t_{t-1}}. \tag{8}$$

4.4. Predictive diffusing zone estimation

Based on given diffusing direction and velocity value, every *mbn(t)* can estimate the predictive location of *vbn(t + Δt)* for a next time *t*. Let (x_{i+1}, y_{i+1}) be the estimated location of *vbn(t + Δt)* and let (x_i, y_i) be the location of *mbn(t)*. We can simply get (x_{i+1}, y_{i+1}) as follows:

$$x_{i+1} = x_i + v_x t cos(\theta_x), \ and \tag{9}$$

$$y_{i+1} = x_i + v_x t sin(\theta_x). \tag{10}$$

After each of MBNs finishes estimation of the virtual location *vbn(t + Δt)*, it starts to estimate the diffusing zone. Fig. 7 shows the diffusing zone estimation process. Current MBN *M*1 and *M*2 constitute the current paired MBNs, whose pair refers to a boundary line element $\overline{M_1M_2}$. Each of those MBNs measures the local diffusing direction and velocity, and we can denote this local measured value as vector $\vec{x_i}$, which has direction and velocity element value as (v_i, θ_i). *M*1 estimates virtual location of future MBN *V*1 using its own $\vec{x_1}$, and *M*2 calculates location of *V*2 in the same way. Finally, two current paired MBNs can exchange virtual location information by sending query message to each other, and one of two MBNs (*M*1 in case of Fig. 7) has the knowledge about predictive boundary line element $\overline{M_1M_2}$, then *M*1 can estimate the diffusing zone based on $\overline{M_1M_2}$.

In order to provide flexible sizes of wakeup zones, we define two parameters for wakeup zone; wakeup region width and wakeup region depth. In general, we may have an intuition that a big size of wakeup zones may be able to decrease the error rate caused by wrong prediction. However, it could awake unnecessary nodes additionally. Thus, those parameters have trade-off between energy efficiency and tracking accuracy by prediction error. Detailed property of each parameter is described as follow:

- *Wakeup region width*: In the case that a continuous object expands widely and the predictive boundary line is much longer than the current boundary line, we need to arrange additional predictive MBNs as shown in Fig. 7. Therefore, we can adjust the width between predictive MBNs. Fig. 7 shows the example of addition

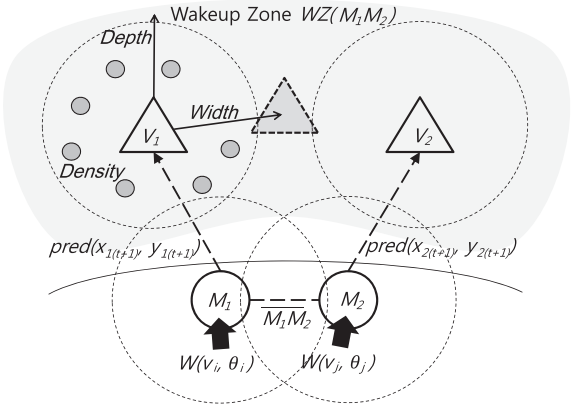


Fig. 7. Diffusing zone estimation.

estimation of *vbn(t + Δt)* when the wakeup region width is communication range *r*.*M*1 calculates additional numbers of vbn(t + Δt) by means of length-ratio between $\overline{M_1M_2}$ and $\overline{V_1V_2}$. In order words, it can estimate the additional vbn(t + Δt) with weighted sum vector of $\vec{x_1}$ and $\vec{x_2}$. The total number N of vbn(t + Δt) s in $\overline{V_1V_2}$ can be calculated as:

$$N = Ceil\left(\frac{\overline{V_1V_2}}{r}\right), \tag{11}$$

and the weighted measurement $\overrightarrow{x_1i}$ can be got by:

$$\overrightarrow{x_1i} = \frac{i}{N}\cdot\vec{x_1} + \frac{N-i}{N}\cdot\vec{x_2}. \tag{12}$$

The start point coordinate of $\overrightarrow{x_1i}$, which is located in the connected line $\overline{M_1M_2}$ between $M1$ and $M2$, can be estimated as:

$$(x_{1i}, y_{1i}) = \left(\frac{i}{N}(x_2 - x_1), \frac{i}{N}(y_2 - y_1)\right). \tag{13}$$

Thus, we can estimate the virtual location of $M1$'s additional vbn(t + Δt) using the estimated vectors and Eqs. (9) and (10).

- *Wakeup region depth*: We could control the thickness of the wakeup zone. To achieve this, predictive MBNs controls the strength of the wakeup signal. Thicker wakeup zone may make bigger wakeup zone. Thick wakeup zones may decrease error rates but could awake unnecessary nodes, i.e., trade-off between energy efficiency and tracking accuracy.

4.5. Error recovery

Although our prediction algorithm is fulfilled correctly, we may not make sure it provides actual results. It means that our prediction algorithm is able to meet prediction error. So, we need a recovery mechanism when activated boundary zones miss actual boundary lines of the targeted continuous object. We define the prediction error as the situation that a predictive MBN for a time $T1$ cannot detect the current boundary of the targeted continuous object at a time $T1$.

In order recover the prediction error, we define the wakeup-all message, denoted by *directial-wakeup-msg(f, l, d)*, which includes a flag *f* for forwarding or backwarding, a location of initiator *l*, and an activation duration *d*. When a time $T1$ passed, a predictive MBN checks its NDT, and if itself and all neighbors detected the object, it declares an *over diffused error* and broadcasts *directial-wakeup-msg(f, l, d)* to forward diffusing direction until activating nodes find the boundary. Note that *directial-wakeup-msg(f, l, d)* is flooded to only one direction, that is to say, if a sleeping node has received this message, then it calculates a direction θ between *l* and its location. And if θ is in the range of $0 < \theta < \pi$, it becomes an active node and rebroadcasts the message again. This forward message propagation is fulfilled in the case of the over diffused error that means the current boundary passed though predictive zones already. In contrast, the less diffuse error indicates the current boundary does not reach the activated zones yet, i.e., when a time $T1$ passed, a predictive MBN as well as all its neighbors did not detect the object. In this case, it sends *directial-wakeup-msg(f, l, d)* to backward direction until activating nodes meet the current boundary of the object.

Another considerable parameter is the activation duration *d*. Activating nodes by this emergency wakeup message do not need to keep the active state. After the recovery process is done, it can go back to sleep again. The parameter means the duration of the activation time for checking its object detection status to find the current boundary of the object.

5. Performance evaluation of balanced boundary detection

In this section, we evaluate the performance of our balanced boundary detection scheme based on simulation results. Computational simulations are implemented in the Qualnet v4.0 simulator [44] to evaluate and compare the performance of COBOM [24,25], the EUCOW [27] and the proposed scheme. Since reliability of report aggregation is another research area, our simulations do not concern possible data loss or contention between nodes and how to route data to the sink. The simulations will experientially assure that our scheme is usually more energy-efficient and accurate than COBOM and EUCOW due to less number of BNs and RNs generated. Though we use the name of MBN (Master Boundary Node) in our scheme, the concept of MBN and RN is similar. Then, in this performance comparison, we just use the name of RN (Representative Node).

5.1. Simulation environments

We simulate a wireless network with sensor nodes distributed on a $1000 * 1000$ m field. The node communication radius is set to 50 m. Numbers of nodes are made to vary from 500 to 1500 in order to change the average number of neighbors for each node (i.e. node density). Although all the sensor nodes can be distributed randomly or distributed with statistical distribution, in order to make sure the node density, we divide $1000 * 1000$ m sensor field into $100 * 100$ m grid cell. In the each of 100 grid cells, we can adjust density of a grid cell from 5 to 15 nodes. As describe in Section 3.2, in EUCOW and COBOM, they do not consider the node density while selecting RNs, then it is likely to be more RNs selected in dense area, but our scheme is not affected by the node density and selects RNs with regular distance basis. So, in order to analyze the impact of node density, we need to more concrete and straight method to vary the distribution with different node density. Therefore, we can measure the node density with the number of node in one grid cell.

5.2. Boundary accuracy

we use a metric named boundary accuracy to evaluate the accuracy of detecting continuous objects between the

proposed balanced boundary detection scheme and the proposed schemes. To measure the boundary accuracy metric, we set a simple ellipse shape of a continuous object at centered at coordinates (500, 500). The length of semi-major axis and semi-minor axis are 350 m and 200 m, respectively. We distrusted sensor nodes with the density of 5 in upper half of sensor filed and the density of 5 in lower half of sensor field. That means that we randomly distributes 10 sensor nodes in grid cells that located in both upper half of sensor field and vice versa. Then, the boundary accuracy metric means how similar the ellipse shape of the continuous object and the distribution of RNs in each scheme are as shown in Figs. 8–10.

Figs. 8–10 show the snapshot of selected RNs in each of the proposed scheme, EUCOW, and COBOM. It is quite obvious even at a glance that the proposed scheme better presents the shape of a continuous object. This result is mainly attributable to difference BN and RN selection rules. Since EUCOW and COBOM merely depend on communication range to select boundary nodes, they select all the sensor nodes within communication range from a continuous object as boundary nodes. Thus, they select a large number of boundary nodes, which result in degrading the quality of boundary accuracy as well as energy-efficiency.

In COBOM, the BN selection is based on BN-Array that only have 1 bit detection status of all neighbor nodes. If a sensor node receives an event detection message from a certain neighbor node who detects coming continuous objects, it first saves the detection status into BN-Array, and then, it scans the BN-Array whether any different detection status exists. If the node finds that any different status exists in BN-Array, it becomes a boundary node. The drawback of this selection rule is that thickness of estimated boundary is almost reached to $2 \times r$ (radio-range). Fig. 10 shows the snapshot of simulation result of COBOM.

Fig. 9 show the simulation result of EUCOW, we can see that the shape of aggregate RN is more accurate than the result of COBOM. In EUCOW, it uses similar detection rule with BN-Array, but in order to enhance the accuracy it compares its own current status and the status of event message. For example, if a sensor node's current detection status is false and it receives an event message from one of its neighbors, and if the status of the event message is true, then the sensor node becomes a BN since current status of the sensor node is different from the status of the event message. According to this BN selection rule, among the sensors which are adjacently located to the boundary of the objects, only the nodes that are out region or in the region will become BNs. Thus, thickness of estimated boundary is almost reached to $1 \times r$ (radio-range), which is half number of BNs compared to COBOM.

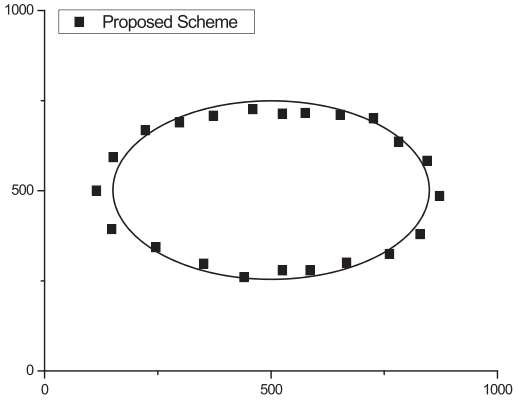


Fig. 8. RNs snapshot of the proposed scheme.

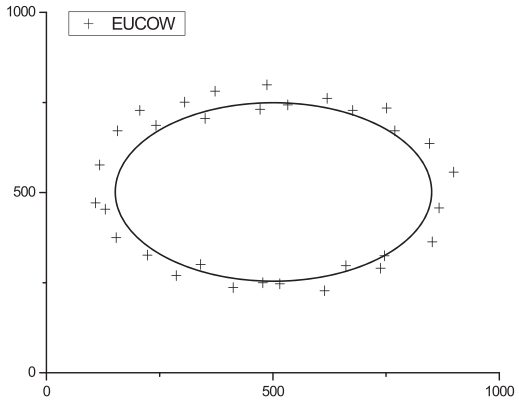


Fig. 9. RNs snapshot of EUCOW.

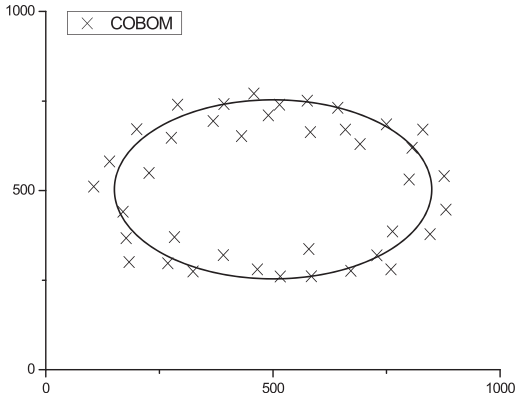


Fig. 10. RNs snapshot of COBOM.

In our boundary detection scheme, we extend the attribute of BN-Array and define NDT which has distance information of neighbor nodes, and we provide a method that guarantees only closest nodes to event sensor to be selected as boundary. Then, the thickness of estimated boundary is very thin and close to real boundary line. Furthermore, since we select RNs with regular distance basis among selected BNs, the selected RNs are located uniformly near the boundary line regardless of node density.

5.3. Impact of different node density

We compare the numbers of BNs and RNs selected in the proposed scheme, EUCOW, and COBOM. The numbers of BNs and RNs are critical performance metrics, because

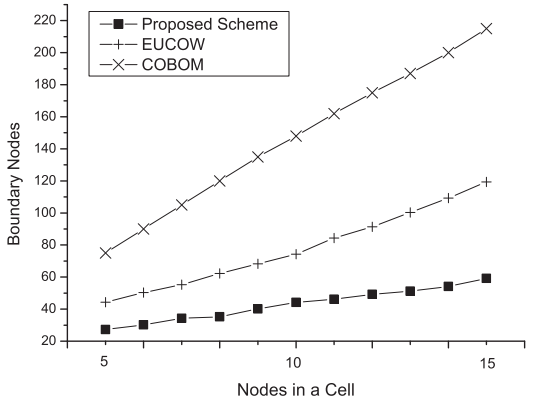


Fig. 11. Comparison of the number of BNs.

the numbers of those nodes directly is related to the numbers of report. For analyzing the effect on node density, we randomly distributed different numbers of sensor nodes ranging from 5 to 15 in each grid cell. The size of a continuous objects is same as previous experiment. As apparently shown in Fig. 11, increasing the nodes density makes more number of BNs in every detection schemes. In cases of COBOM and EUCOW, BNs in COBOM should cover both in and out regions whereas only the nodes that are either in or out region can become BNs in EUCOW. Thus, the number of BNs in EUCOW is much less compared to that of COBOM. But the experiment show that our scheme selects less number of BNs than EUCOW, since the number of closet nodes to boundary node is limited, although the density is higher.

Fig. 12 presents the number of RNs, similar to the case of the number of BNs. The node density influences to the number of RNs. Though a RN suppress its 1-hop neighbors during RN selection period in both of COBOM and EUCOW, the distance between RNs, denoted by d, is always in range $r < d < 2r$. However, our scheme almost guarantees $2r$ by cascading the selection rule regardless of the node density.

5.4. Impact of different object size

In this experiment, we evaluate an impact of diffusing of continuous objects. To make diffusing scenarios of a continuous object, we set simple its circle shape. The initial radius of a circle object is 50 m and starts from coordinates (500, 500) where is center of the simulated sensor field. The radius of the circle object increase 10 m at every second. Every sensor node starts the BN and RN selection at every 3 s sampling duration, the total simulation time is 30 s, and then total sampling number is 10 times. We do not take COBOM into consideration, since it almost makes double size of BNs and RNs as already shown in Figs. 11 and 12.

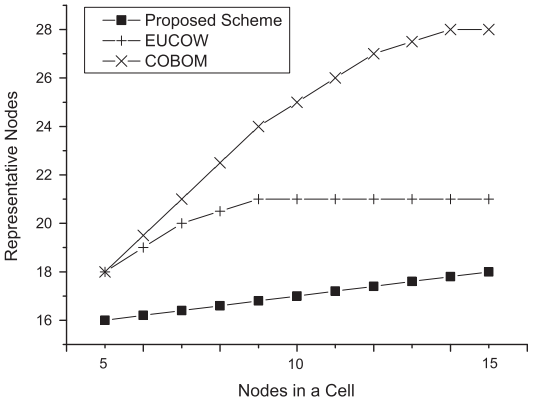


Fig. 12. Comparison of the number of RNs.

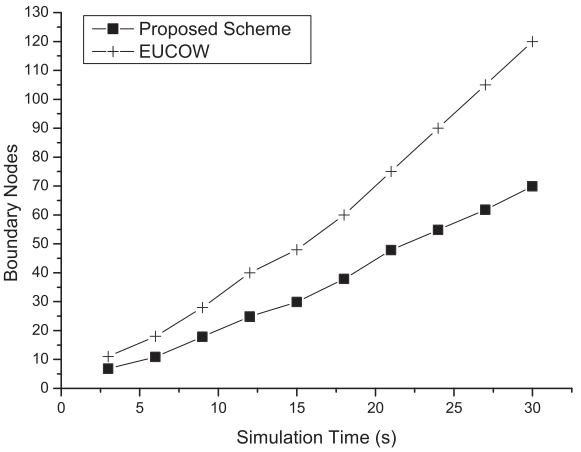


Fig. 13. Comparison of the number of BNs.

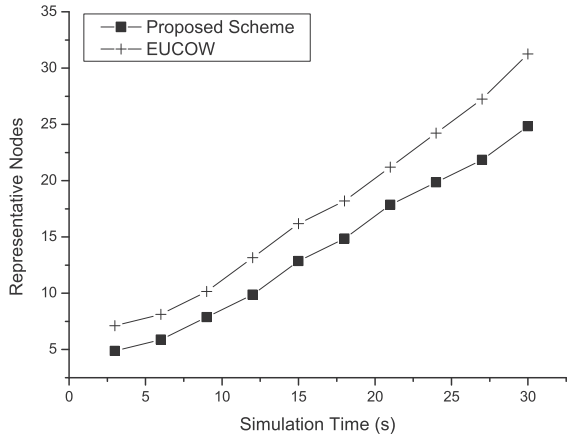


Fig. 14. Comparison of the number of BNs.

As intuitively expected, the number of BNs becomes larger as the object's radius increases. As the monitoring region that sensor nodes need to cover gets larger, it is natural that the number of BNs involved also increases in both EUCOW and our scheme, as shown it Fig. 13. However, less BNs in our scheme are selected and the gap of the number of BNs between two schemes keeps widening as the time goes on. This is because the boundary region is increasing highly as the radius of a circle object is longer. In the case of the number of RNs, both schemes also linearly increase, but our scheme always has less number of RNs as shown in Fig. 14.

6. Performance evaluation of boundary zone-based state switching

In this section, we present simulation results to evaluate the performance and effectiveness of our boundary zone-based state switching scheme. We first describe performance metrics used for evaluating simulation results and analyze the energy saving effect of scheme using the simulation results. And then, we will analyze the impact of two wakeup zone parameters: wakeup region width and wakeup region depth.

Two metrics are defined in order to evaluate the proposed scheme.

- *Total energy consumption*: Total energy consumption in both active and sleeping modes by the network during the simulation time.
- *Prediction accuracy*: we use this metric to evaluate the degradation of boundary detection caused by the error of our predictive diffusing zone estimation. The degradation of boundary detection means that our boundary zone-based state switching scheme does not select MBNs due to boundary detection fails caused by the predictive diffusing zone estimation error. As comparison basis of energy consumption, we use an all-active scheme which keeps all sensor nodes in active mode without state switching. We define the prediction accuracy metric as numerical equivalence between the total number of reported MBNs due to boundary detection successes denoted by $N_{success}$ and the total numbers of unreported MBNs due to boundary detection fails denoted by N_{fail}. Namely, the prediction accuracy is defined as $N_{success}/(N_{success}+N_{fail})$.

6.1. Simulation environments

The simulating network consists of 4000 sensor nodes that are deployed uniformly in a $2000 * 2000$ m square area. Properties of sensor nodes are same as previous experiment for evaluating boundary detection scheme. Based on the characteristics of IRIS mote hardware [39], active mode and sleep mode power consumption rates of sensor node are 31 mW and 36 μW, and transmitting and receiving power consumption rates are 75 mW and 63 mW, respectively. To make state switching scenario, we set simply circle shape of continuous object. It is initiated with a radius of 150 m and start from coordinates (500, 500) where is center of simulation sensor field.

Every sensor nodes are well synchronized, and we assume that application require the boundary information report every 10 s. The radius of circle objects initially increases with speed 5 m/s and for varying the diffusion speed, acceleration is randomly increase or decrease 1 m/s, then, maximum variation of speed at every report time is 10 m/s. For providing the wakeup parameters, we define wakeup region width and depth as radio range *r*. Since we ignore the initial detection of continuous objects in this experiment, sensor node in current boundary zone of circle shape of objects stay active and others keep in sleep mode.

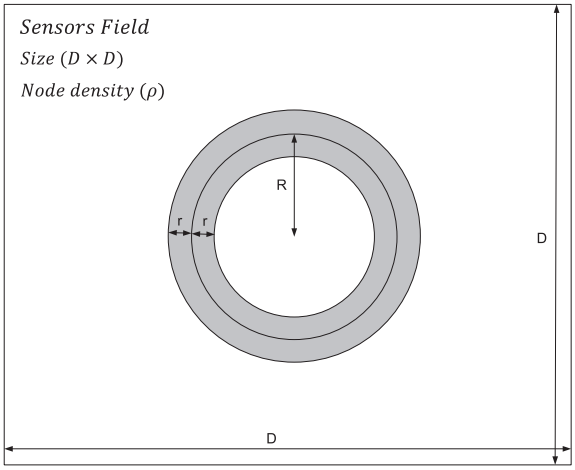


Fig. 15. Energy saving effect analysis of state switching.

6.2. Impact of state switching

In our experimental environment, we can statically analyze the energy saving effect of state switching. We already have assumed the circle shape of objects and define wakeup zone as radio range *r*. As shown in Fig. 15, if we assume sensor nodes density ρ per a unit area and average radius *R* of circle shape of objects, then expected the number of active sensor nodes in boundary zone at time *t* denoted by *BZ[t]* can be represented as follows:

$$E_{BZ}[t] = \pi\{(R+r)^2 - (R-r)^2\}\rho = 4\pi rR\rho, \tag{14}$$

then, we can estimate the total energy consumption of $E_BZ[t]$ as follow:

$$E_{all}[t] = (D^2 - 4\pi rR)\rho \times \beta + 4\pi rR\rho \times \alpha + C, \tag{15}$$

where $E_{all}[t]$ indicates the total energy consumption of $E_{BZ}[t]$ and D^2 indicates square sensor field with width *D*. Each of α, β and *C* represents constant values of active mode power consumption, sleep mode power consumption, and total transmitting and receiving power consumption. *C* is based on the fact that in the proposed boundary detection scheme, every active sensor nodes broadcast and receive event detection message once respectively.

Then, total energy saving $E_{saving}[t]$ compared with all-active-scheme can be estimated as below:

$$E_{saving}[t] = (\alpha - \beta)(D^2 - 4\pi rR)\rho. \tag{16}$$

Thus, energy saving effect of the proposed state switching scheme is depends on the size of sensor filed and sensor nodes density. In this experiment, based on described characteristics of IRIS hardware we can save almost average 9000 mW energy saving in every second. Our experiment just use small size of sensor filed, but in real environment, huge sensor field could be deployed for large scale of continuous objects, and it is obvious that tremendous energy saving can be expected proportional to the size of sensor filed through the sensor state switching scheme.

However, the proposed state switching scheme has the communication overhead for waking up sleeping sensor nodes of the next predicted boundary compared with the all-active scheme. In the proposed scheme, the communication overhead is both the cost for broadcasting a wakeup message from each MBN of the previous boundary to all sleeping sensor nodes in its wakeup zone of the next predicted boundary and the cost for receiving the wakeup message by all sleeping sensor nodes in all wakeup zones of the next predicted boundary. With the example In Fig. 14, the wakeup message broadcasting cost is $N_{P\_MBN} \times e_t$ and the wakeup message receiving cost is $4\pi rR\rho \times e_r$, where N_{P_MBN} is the number of MBNs in the previous boundary, and e_t and e_r are the communication costs for transmitting and receiving, respectively. When the width for the distance between two MBNs is given as w, N_{P_MBN} is approximately $2\pi(R-r)/w$. Thus, the communication overhead for waking up sleeping sensor nodes of the next predicted boundary in the proposed state switching scheme is as follows:

$$E_{overhead}[t] = (2\pi(R-r)/w) \times e_t + 4\pi rR\rho \times e_r. \quad (17)$$

However, the communication overhead $E_{overhead}[t]$ is considerably small compared with total energy saving $E_{saving}[t]$.

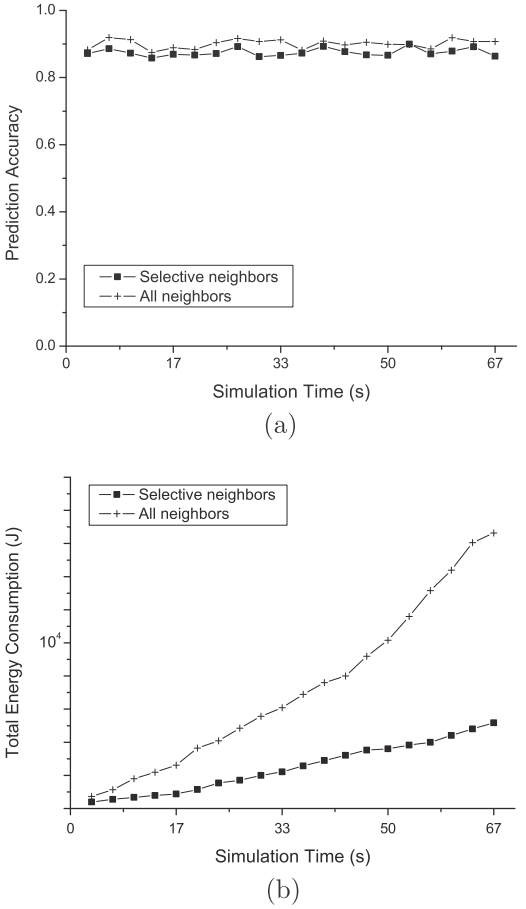


Fig. 16. Impact of state switching.

Fig. 16(b) shows the total energy consumption of our boundary zone-based state switching scheme (i.e. selective neighbors) and an all-active scheme (i.e. all neighbors). This figure includes both the saving energy and the overhead energy by our boundary zone-based state switching. As shown in the figure, our boundary zone-based state switching is more efficient than an all-active scheme because the saving energy is much higher than the overhead energy in our boundary zone-based state switching. Fig. 16(a) shows the prediction accuracy of our boundary zone-based state switching scheme (i.e. selective neighbors) and an all-active scheme (i.e. all neighbors). In spite of much energy saving, boundary zone-based state switching scheme has the similar boundary prediction result compared with the all-active scheme.

Since BNs and RNs (called MBNs in our paper) have different roles, we compare the energy consumption for BNs and RNs according to their roles. Since the tracking of continuous objects only focuses on the boundary information of the continuous objects, if sensor nodes detect a continuous object at a time $t1$, BNs and RNs among them are generated by the algorithm of the proposed scheme at the time $t1$. Because the continuous object changes (i.e. moves or diffuses), the boundary of the continuous object is changed at next time $t2$. Thus, new BNs and RNs are generated at the time $t2$. Thus, we compared the energy consumption for BNs and RNs at a time $t1$. As a continuous object happens, sensor nodes around it detect it and exchange an event message between one-hop neighbor sensor nodes. Then, a detecting sensor node becomes a BN if it checks all event messages from its neighbor sensor nodes and meets two conditions to be a BN. Thus, the energy consumption of a BN is as follows when a sensor node has N neighbor sensor nodes:

$$BN(E) = N(e_t + e_r). \quad (18)$$

If BNs are generated on the boundary of the continuous object at the time $t1$, Some BNs among them become RNs due to having shorter expire times to be a RN than the other BNs. Then, each RN broadcasts a RN selection message to its one-hop neighbor BNs. The energy consumption for this is $1e_t$. Next, the RN sends a SBN (Slave Boundary Node in our paper) selection message to the most left BN and the most right BN. The energy consumption for this is $2e_t$. Last, the RN sends a wakeup message to the wakeup region on the next boundary of the continuous object. The energy consumption for this is $1e_t$. Thus, the energy consumption of a RN is as follow:

$$RN(E) = BN(E) + 1e_t + 2e_t + 1e_t = BN(E) + 4e_t \quad (19)$$

As shown in the Eqs. (18) and (19), RNs consume more energy than BNs but the difference is only $4e_t$.

Fig. 17 shows the distribution of the residual energy for the total energy (100%) of every sensor nodes. In the all-active scheme, about 40% sensor nodes have 20–40% residual energy because they are continuously awake and detect the continuous object. The other sensor nodes of about 48% have about 60–70% residual energy because they are only awake and do not need to detect the continuous object far from them. On the other hands, in our boundary zone-based state switching scheme, many sensor nodes

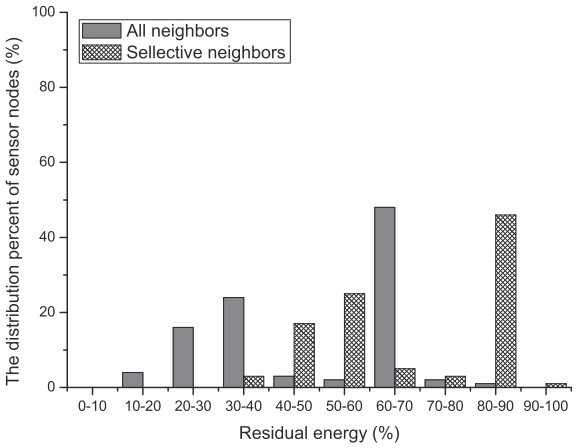


Fig. 17. The distribution of residual energy.

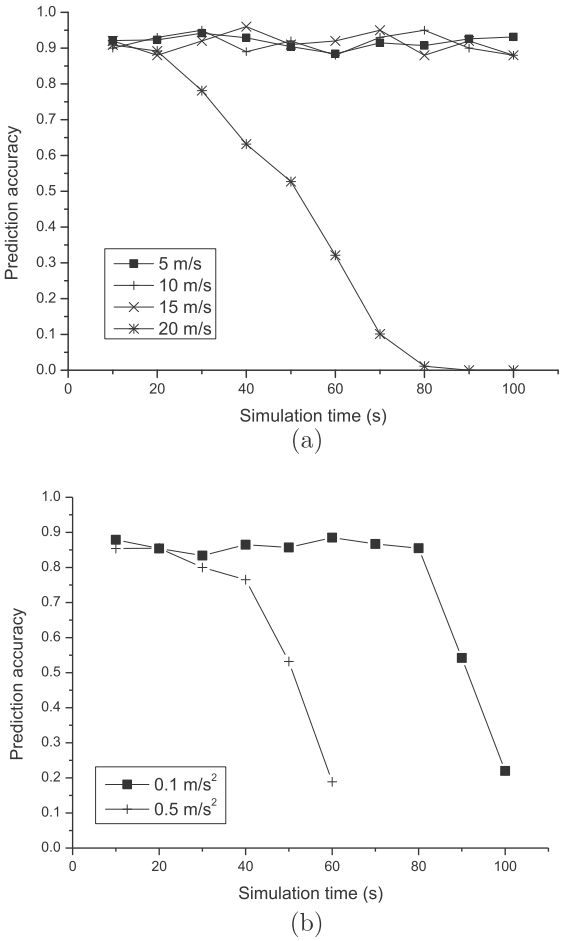


Fig. 18. Impact of diffusion velocity.

have high residual energy. About 25% sensor nodes have about 50–60% residual energy because they use state switching and conduct the roles for BN. About 15% sensor nodes have about 40–50% residual energy because they also use state switching and conduct the roles for RN. The other sensor nodes of 45% have about 80–90% residual energy because they are much time in sleep mode by state switching and do not need to conduct as BNs or RNs due to not detecting the continuous object far from them.

6.3. Impact of diffusing velocity

In this simulation, we evaluate an impact of different diffusing speed. Diffusing speed is basis for estimation of diffusing zone in each master node. For different diffusing speed scenario, we use same simulation environment as previous simulation except that diffusing speed. We simulate with four different diffusing speeds that are 5 m/s, 10 m/s, 15 m/s, and 20 m/s and each of speed do not get changed during simulation time. As shown in Fig. 18(a), fixed slow or fast diffusing speed do not influence in prediction accuracy, because current local measured diffusing speed always is same as future diffusing speed as well as previous speed. The problem occurrence timing in proposed scheme is when the speed is rapidly changed. Since the depth of diffusing zone is radio range r (50 m) and report time is 10 s, if the variation of speed is less than 5 m/s, then ideally sensor nodes in diffusing zone can detect the boundary.

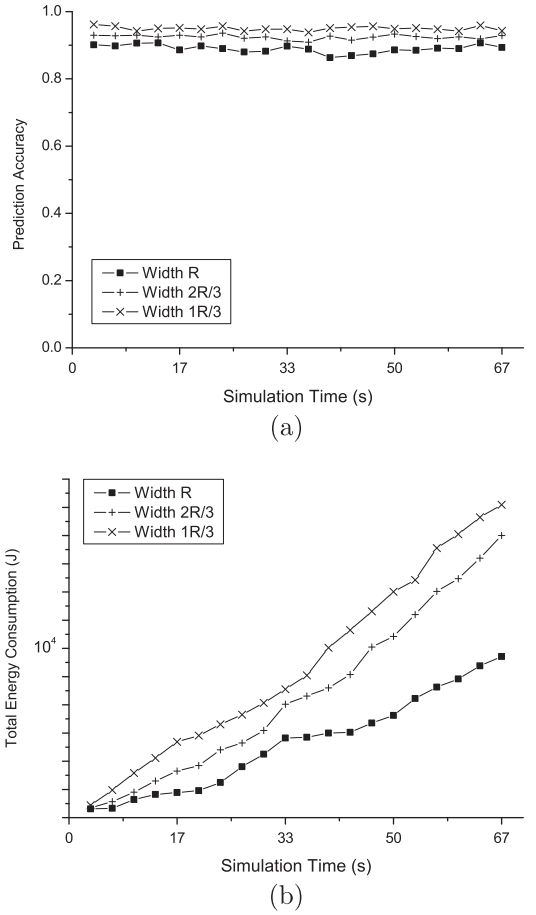


Fig. 19. Impact of wakeup region width.

An interesting acceleration problem is shown in Fig. 18(b). We simulate two case of uniform acceleration. Since our local diffusing velocity measurement do not consider the acceleration, the gap between actual increasing velocity and instant local velocity enlarge as time goes on. At some moment, it cross over the threshold of estimated diffusing zone, and eventually every active nodes fail to detection boundary. But in real environment, continuous object usually diffuse outward steadily though it sometimes can changes unexpectedly speed or direction.

6.4. Analysis of wakeup zone parameters

In the proposed switching scheme, in order to provide flexible size of wakeup zone, we define three parameters for wakeup zone; wakeup region width and wakeup region depth. In general, we can have intuition that a big size of wakeup zone will decrease the error rate caused by wrong prediction, but it could wake up unnecessary nodes, which result in more energy consumption. Thus, those parameters have trade-off between energy efficiency and tracking quality caused by the prediction error. This trade-off is shown in Figs. 19 and 20.

Fig. 19 shows the impact of different wakeup region widths. We evaluate the performance with setting of widths as radio ranges R, $2R/3$ and $1R/3$. The default value of the other parameters, and depth is $1R/2$. The narrower width results in more energy consumption and better prediction accuracy, since a narrower width activates more number of predictive boundary nodes and their neighbor nodes.

Fig. 20 shows the impact of different wakeup region depths. In this experiment, we divide the region depths into radio ranges R, $2R/3$ and $1R/3$. The default value of the other parameters, and width is R. The longer radius makes somewhat rapidly increasing energy consumption, but the impact to prediction accuracy is not too much in comparison with energy consumption. This is because linear increasing of radius causes the activation of exponential increasing sensor nodes, which result in a lot of unnecessary active sensing nodes.

6.5. Impact of latency about waking and moving

Since the proposed scheme uses a sleep-wake scheduling method for the energy efficiency, RNs on the current boundary of a continuous object should wake sleeping sensor nodes on the next boundary of the continuous object. To make the proposed scheme work well, the latency for

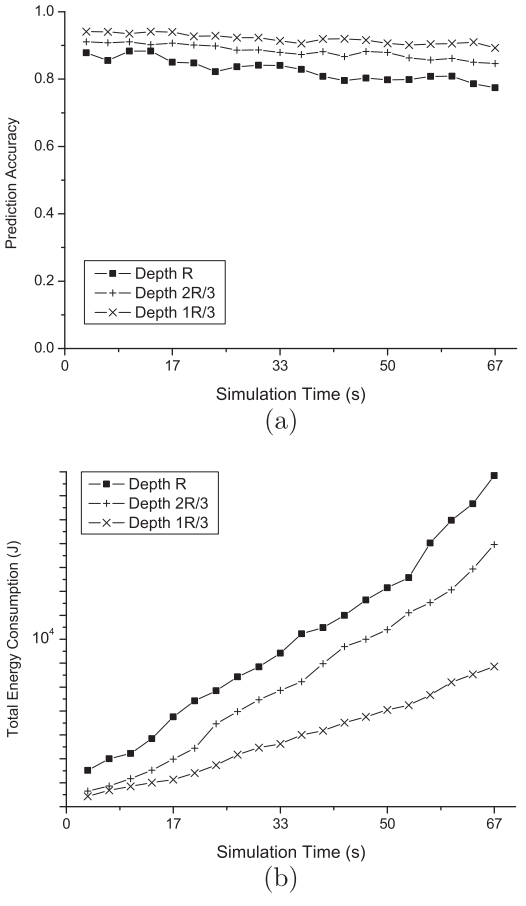


Fig. 20. Impact of wakeup region depth.

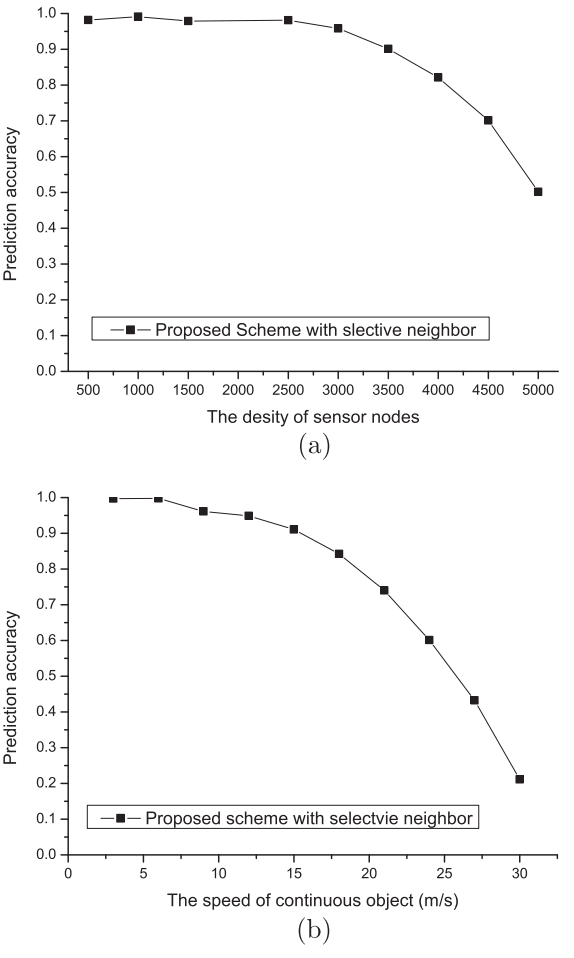


Fig. 21. Impact of latency about waking and moving.

waking the sleeping sensor nodes on the next boundary should be smaller than the latency for moving to the continuous object from the current boundary to the next boundary. If the latency for waking the sleeping sensor nodes on the next boundary is larger than the latency for moving of the continuous object from the current boundary to the next boundary, because sensor nodes on the next boundary cannot detect the continuous object due to their unawakened states, the proposed scheme may have very low prediction accuracies. Generally, the latency of waking the sleeping sensor nodes on the next boundary is affected by the number of sensor nodes on the next boundary prediction regions (i.e. the density (number) of sensor nodes in the network when seeing with a large view). As the density of sensor nodes increases, the latency of waking the sleeping sensor nodes on the next boundary grows and thus the proposed scheme decreases the prediction accuracy. Fig. 21(a) shows the prediction accuracy for the density of sensor nodes through our simulations. On the other hand, the latency for moving of a continuous object from the current boundary to the next boundary is apparently affected by the speed of the continuous object. If the speed of a continuous object increases, the continuous object can move to the next boundary before waking sleeping sensor nodes on the next boundary and thus the proposed scheme decreases the prediction accuracy. Fig. 21(b) shows the prediction accuracy for the speed of a continuous object through our simulations.

7. Conclusion

In this paper, we concentrate on energy efficiency and accuracy for monitoring large-scale phenomena, called continuous objects. Firstly, we try to reduce Boundary Nodes (BNs) and Master Boundary Nodes (MBNs) that are mainly related to communication costs and monitoring accuracy. While existing studies are based on the radio range to choose those nodes, i.e., it means there are a large number of BNs and MBNs and the number depends on the node density, we select BNs only close to boundaries of objects and allocate MBNs to a certain distance. In addition, we consider sleep state switching of sensor nodes that is the most efficient way for energy saving. For this, we propose a state switching scheme with boundary prediction relying on the BN and MBN selection mechanism. Finally, by a variety of computational simulations we prove that our proposals show better performance regarding energy efficiency and accuracy than previous works.

References

[1] R. Shorey, A.L. Ananda, M.C. Chan, W.T. Ooi, Mobile, Wireless, and Sensor Networks: Technology, Applications, and Future Directions, IEEE Press, Wiley-Interscience, Hoboken, N.J., 2006. pp. 173–196.

[2] A. Cerpa, J. Elson, M. Hamilton, J. Zhao, Habitat monitoring: application driver for wireless communications technology, in: ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean, Costa Rica, April 2001, pp. 20–41.

[3] Q. Huang, C. Lu, G.C. Roman, Mobicast: just-in-time multicast for sensor networks under spatiotemporal constraints, in: Proc. ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), April 2003.

[4] Q. Huang, C. Lu, G.C. Roman, Spatiotemporal multicast in sensor networks, in: Proc. The ACM Conference on Embedded Networked Sensor Systems (SenSys), November 2003, pp. 205–217.

[5] Y.S. Chen, S.Y. Ann, VE-Mobicast: a variant-egg-based mobicast routing protocol in wireless sensor networks, in: Proc. The 40th IEEE International Conference on Communications (ICC), vol. 5, Seoul, Korea, May 2005, pp. 3020–3024.

[6] Y. S. Chen, Y. J. Liao, HVE-mobicast: a hierarchical-variant-egg based mobicast routing protocol for wireless sensornets, in: Proc. The IEEE Wireless Communications and Networking Conference (WCNC), vol. 2, Las Vegas, NV, USA, April 2006, pp. 697–702.

[7] H. T. Kung, D. Vlah, Efficient location tracking using sensor networks, in: Proc. IEEE Wireless Communications and Networking Conference (WCNC) 2003, New Orleans, March 2003, pp. 1954–1961.

[8] C. Lin, Y. Tseng, Structures for in-network moving object tracking in wireless sensor networks, in: Proc. First International Conference on Braodband Networks (BroadDNets) 2004, San Jose, October 2004, pp. 718–727.

[9] C. Lin, W. Peng, Y. Tseng, Efficient in-network moving object tracking in wireless sensor networks, IEEE Trans. Mobile Comput. 5 (8) (2006) 1044–1056.

[10] W. Zhang, G. Cao, Optimizing tree reconfiguration for mobiletarget tracking in sensor networks, in: Proc. the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), Hong Kong, March 2004, pp. 2434–2445.

[11] W. Zhang, G. Cao, DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks, IEEE Trans. Wireless Commun. 3 (5) (2004) 1689–1701.

[12] M. Fayyaz, Classification of object tracking techniques in wireless sensor networks, Wireless Sensor Network 3 (4) (2011).

[13] W.P. Chen, J.C. Hou, L. Sha, Dynamic clustering for acoustic target tracking in wireless sensor networks, in: Proc. 11th IEEE International Conference on Network Protocols (ICNP), Atlanta, November 2003, pp. 284–294.

[14] T. Vercauteren, D. Guo, X. Wang, Joint multiple target tracking and classification in collaborative sensor networks, IEEE J. Sel. Areas Commun. 23 (4) (2005) 714–723.

[15] G. Jin, X. Lu, M. Park, Dynamic clustering for object tracking in wireless sensor networks, in: Proc. Ubiquitous Computing Systems (UCS) 2006, Seoul, October 2006, pp. 200–209.

[16] S. Pino-Povedano, F.J. Gonzalez Serrano, Distributed tracking and classification of targets with sensor networks, in: Proc. 16th International Conference on Software, Telecommunications and Computer Networks (SoftCOM), Split, September 2008, pp. 213–217.

[17] S. Goel, T. Imielinski, Prediction-based monitoring in sensor networks: taking lessons from MPEG, ACM Comput. Commun. Rev. 31 (5) (2001).

[18] Y. Xu, W.C. Lee, On localized prediction for power efficient object tracking in sensor networks, in: Proc. 1st International Workshop on Mobile Distributed computing, Providence RI, May 2003, pp. 434–439.

[19] Y. Xu, J. Winter, W.C. Lee, Prediction-based strategies for energy saving in object tracking sensor networks, in: Proc. The IEEE International Conference on Mobile Data Management (MDM), Berkeley, California, January 2004, pp. 346–357.

[20] Y. Xu, J. Winter, W.C. Lee, Dual prediction-based reporting for object tracking sensor networks, in: Proc. The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous), 2004, pp. 154–163.

[21] X. Ji, H. Zha, J. J. Metzner, G. Kesidis, "Dynamic cluster structure for object detection and tracking in wireless ad-hoc sensor networks, in: Proc. IEEE International Conference on Communication (ICC) 2004, vol. 7, July 2004, pp. 3807–3811.

[22] W. Chang, H. Lin, Z. Cheng, CODA: a continuous object detection and tracking algorithm for wireless ad hoc sensor networks, in: Proc. IEEE Consumer Communications and Networking Conference (CCNC) 2008, January 2008, pp. 168–174.

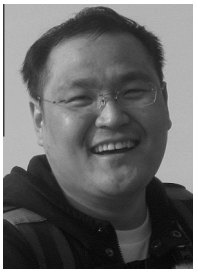
[23] B. Park, S. Park, E. Lee, S. Kim, Detection and tracking of continuous objects for flexibility and reliability in sensor networks, in: Proc. IEEE International Conference on Communication (ICC), 2010.

[24] C. Zhong, M. Worboys, Energy-efficient Continuous Boundary Monitoring in Sensor Networks, Technical Report, 2007. <http://ilab1.korea.ac.kr/papers/ref2.pdf>.

[25] C. Zhong, M. Worboys, Continuous contour mapping in sensor networks, in: Proc. IEEE Consumer Communications and Networking Conference (CCNC) 2008, January 2008, pp. 152–156.

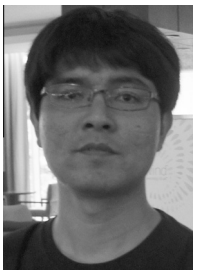
[26] J. Kim, K. Kim, S.H. Chauhdary, W. Yang, M. Park, DEMOCO: energy-efficient detection and monitoring for continuous objects in wireless sensor networks, IEICE Trans. Commun. E91-B (11) (2008) 3648–3656.

[27] J. Chen, M. Matsumoto, EUCOW: energy-efficient boundary monitoring for unsmoothed continuous objects in wireless sensor network, in: Proc. IEEE 6th Mobile Adhoc and Sensor System (MASS) 2009, October 2009, pp. 906–911.
[28] US Naval Observatory (USNO) GPS Operations, 2001. <http://tycho.usno.navy.mil/gps.html>.
[29] N. Bulusu, J. Heidemann, D. Estrin, GPS-less low cost outdoor location for very small devices, in: Proc. IEEE Personal Communications (Special Issue on Smart Space and Environments), vol. 7, 2000, pp. 28–34.
[30] P.K. Liao, M.K. Chang, C.C.J. Kuo, Distributed edge detection with composite hypothesis test in wireless sensor networks, in: Proc. IEEE Global Telecommuncation Conference (Globecom), December 2004.
[31] C. Intanagonwiwat, R. Govindan, D. Estrin, Directed diffusion: a scalable and robust communication paradigm for sensor networks, in: Proc. the 6th Annual International Conference on Mobile Computing and Networking (MobiCom), Boston, August 2000, pp. 56–67.
[32] A. Perrig, R. Szewczyk, J.D. Tygar, V. Wen, D. Culler, SPINS: security protocols for sensor networks, Wireless Networks 8 (5) (2002) 521–534.
[33] N. Hayashibara, A. Cherif, T. Katayama, Failure detectors for large-scale distributed systems, in: Proc. the 21st IEEE Symposium on Reliable Distributed Systems (SRDS), Suita, October 2002.
[34] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, T. Tuan, PicoRadios for wireless sensor networks: the next challenge in ultra-low-power design, in: Proc. 2002 IEEE International Solid-State Circuits Conference (ISSCC), San Francisco, February 2002, pp. 200–201.
[35] Rockwell Science Center, WINS Project. <http://wins.rsc.rockwell.com>.
[36] V. Raghunathan, C. Schurgers, S. Park, M.B. Srivastava, Energy-aware wireless microsensor networks, IEEE Signal Process. Mag. 19 (March) (2002) 40–50.
[37] J.M. Kahn, R.H. Katz, K.S.J. Pister, Next century challenges: mobile networking for smart dust, in: Proc. the 5th Annual International Conference on Mobile Computing and Networking (MobiCom), New York, August 1999, pp. 483–492.
[38] J. Polastre, R. Szewczyk, D. Culler, Telos: enabling ultra-low power wireless research, in: Proc. IEEE International Conference on Information Processing in Sensor Networks (IPSN), April 2005.
[39] Crossbows Data Sheet of IRIS mote, <http://www.xbow.com/Products/Product_pdf_files/.../IRIS_Datasheet.pdf>.
[40] C. Guo, L. Zhong, J.M. Rabaey, Low-power distributed MAC for ad hoc sensor radio networks, in: Proc. IEEE Global Telecommuncation Conference (Globecom), San Antonio, December 2001.
[41] J. Rabaey, J. Ammer, T. Karalar, S. Li, B. Otis, M. Sheets, T. Tuan, PicoRadios for wireless sensor networks: the next challenge in ultra-low-power design, in: Proc. The International Solid-State Circuits Conference (ISSCC), San Francisco, CA, February 2002.
[42] J. Elson, L. Girod, D. Estrin, Fine-grained network time synchronization using reference broadcasts, in: Proc. The 5th Symposium on Operating Systems Designs and Implementation(OSDI), Boston, Mass, USA, 2002.
[43] S. Ping, Delay Measurement Time Synchronization for Wireless Sensor Networks, Tech. Rep. IRB-TR-03-013, Intel Research Berkeley Lab, June 2003.
[44] Scalable Network Technologies, Qualnet. <http://www.scalable-networks.com>.



Soochang Park He received the Ph.D. degree in Computer Science and Engineering from Chungnam National University in Aug. 2011. He is currently a research associate in Institut Mines-Telecom, Telecom SudParis. His research interests are the areas of computer communication and networking. He is mainly interested in IP routing with AS-level configuration; mobility management in both IPv4 and IPv6 including host-based and network-based; and routing, mobility management, and QoS in MANETs and WSNs.

Seung-Woo Hong He received the B.S. degree in Computer Science from Kyungsung University in 1999, the M.S. degree in Computer Science from Pusan National University in 2001, and the Ph.D. degree in Computer Science from Chungnam National University in 2011. He is currently working in Network OS Research Team at Electronics and Telecommunications Research Institute. He is mainly interested in routing in IP Networks and WSNs.



Euisin Lee He received the B.S., M.S., and Ph.D. degrees from Computer Engineering at Chungnam National University, Daejeon, Korea, in 2005, 2007, and 2012, respectively. He studied as a Post-Doctoral researcher in the department of Computer Science at University of California at Los Angeles from 2012 to 2014. He joined the School of Information and Communication Engineering at Chungbuk National University in 2014 and is currently working as an assistant professor. His research interests are in the areas of computer communication and networking. He is mainly interested in routing, multicasting, mobility management, location service, and QoS (Real-time and Reliability) in Mobile Ad hoc Networks (MANETs), Wireless Sensor Networks (WSNs), Vehicular Ad hoc Networks (VANETs), and Smart Grid.

Sang-Ha Kim He received the B.S. degree from Seoul National University, Seoul, Korea, in 1980, and the M.S. in chemical physics and Ph.D. degrees in computer science from University of Houston, Houston, USA, in 1984 and 1989, respectively. He joined the System Engineering Research Institute (SERI) in Korean Institute of Science and Technology (KIST), Seoul, Korea, as a senior research scientist in 1990. After two years, He moved into Chungnam National University, at which he is currently working as a professor.

Noël Crespi, professor, holds Masters degrees from the Universities of Paris 11 and Kent, a diplome d'ingénieur from Telecom ParisTech, a Ph.D and an Habilitation from Paris VI University. From 1993 he worked at CLIP, Bouygues Telecom and then at France Telecom R&D in 1995. In 1999, he joined Nortel Networks as Telephony Program manager. He joined Institut Telecom in 2002 and is currently professor and Program Director, leading the Service Architecture Lab. He coordinates the standardisation activities for Institut Telecom at ITU-T, ETSI and 3GPP. He is also an adjunct professor at KAIST and is on the 4-person Scientific Advisory Board of FTW, Austria.