Underground Structure Monitoring with Wireless Sensor Networks

Mo Li, Yunhao Liu
Hong Kong University of Science and Technology
{limo, liu}@cse.ust.hk

ABSTRACT

Environment monitoring in coal mines is an important application of wireless sensor networks (WSNs) that has commercial potential. We discuss the design of a Structure-Aware Self-Adaptive WSN system, SASA. By regulating the mesh sensor network deployment and formulating a collaborative mechanism based on a regular beacon strategy, SASA is able to rapidly detect structure variations caused by underground collapses. A prototype is deployed with 27 Mica2 motes. We present our implementation experiences as well as the experimental results. To better evaluate the scalability and reliability of SASA, we also conduct a large-scale trace-driven simulation based on real data collected from the experiments.

Categories and Subject Descriptors

C.2.1 [Computer Communication Networks]: Network Architecture and Design – Distributed networks; Wireless communication.

General Terms: Algorithms, Design, Measurement

Keywords

Wireless Sensor Networks, Structure Monitoring, Underground, Coal Mine

1. INTRODUCTION

A Wireless Sensor Network (WSN) is a self-organized wireless network composed of a large number of sensor nodes that interact with the physical world [3]. Various low-power and cost-effective sensor platforms have been developed based upon recent advances in wireless communication and micro system technologies. The increasing study of WSNs [4, 20, 21] aims to enable computers to better serve people by automatically monitoring and interacting with physical environments.

Environment monitoring in underground tunnels (which are usually long and narrow, with lengths of tens of kilometers and widths of several meters) has been a crucial task to ensure safe working conditions in coal mines where many environmental factors, including the amount of gas, water, and dust, need be monitored. To obtain a full-scale monitoring of the tunnel environment, sample data need be collected at many different places. A precise environment overview requires a high sampling density, which involves a large number of sensing devices. Current methods of coal mine monitoring are typically conducted in a sparse and manual way, due to the lack of corresponding techniques for constructing an automatic large-scale sensing system.

Utilizing wires to connect sensing points to the processing server requires a large amount of wire deployment, which is difficult because of poor working conditions and high maintenance costs underground. Moreover, the wired communication method makes the system less scalable; as the tunnel advances, more sensing devices need to be deployed. A wireless system takes advantage of convenient deployment and flexible adjustment. Due to the unpredictable interference caused by the proximity of working machines and miners, however, it is often impossible to maintain direct wireless communication channels between sensing devices and the processing server. The utilization of a WSN to implement the underground monitoring system benefits from rapid and flexible deployment. Additionally, the multi-hop transmitting method conforms to the tunnel structure and provides more scalability for system construction.

The unstable nature of geological construction in coal mines makes underground tunnels prone to structural changes. This instability, which could result in collapses caused by mine quakes or coasts, renders previous WSN monitoring solutions unfeasible. Among the 480 coal mine fatalities [1] reported in the past 10 years in U.S., collapses account for more than 50%. Most fatalities are the result of small collapses caused by falling roof or walls. Hence, it is of great importance to quickly detect the collapse hole regions and accurately provide location references for workers. Since a collapse may destroy part of a monitoring system, maintaining the validity of the network in extreme situations becomes a challenge, which is rarely encountered in previous WSN applications.

In this paper, we present a **S**tructure-**A**ware **S**elf-**A**daptive sensor system, SASA, which aims to address the challenges and provide a feasible framework for underground monitoring in coal mines. The design objectives of SASA include: 1) the ability to rapidly detect the collapse area and report to the sink node; and 2) the ability to maintain the system integrity when the sensor network structure is altered.

SASA employs a hole-detection algorithm to monitor the inner surface of tunnels by utilizing radio signals among sensor nodes to model the structure of the sensor network. With an appropriate placement of sensor nodes and a collaborative mechanism, SASA is able to accurately report locations of collapses, to detect and reconfigure displaced nodes, thus maintaining system integrity.

We conducted field studies in the D. L. coal mine and deployed a prototype system, which included 27 Crossbow Mica2 motes [10]. SASA can provide other functions such as gas and water

IPSN'07, April 25–27, 2007, Cambridge, Massachusetts, USA.
Copyright 2007 ACM 978-1-59593-638-7/07/0004...$5.00.

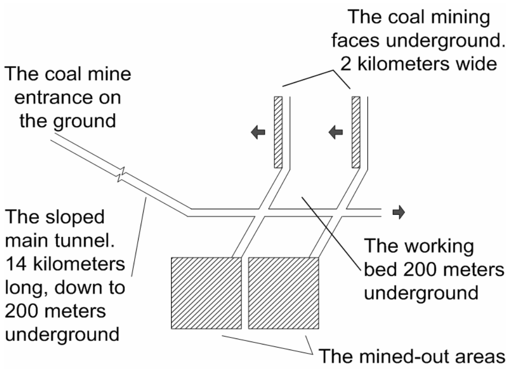

The coal mining faces underground. 2 kilometers wide
The coal mine entrance on the ground
The sloped main tunnel. 14 kilometers long, down to 200 meters underground
The working bed 200 meters underground
The mined-out areas

Fig. 1. An illustration of the D. L. coal mine

detection, oxygen density monitoring, but in this study we focus on structure monitoring and aim to provide a feasible framework which can be easily loaded with various concrete monitoring tasks. Due to resource and environment constraint, our prototype is limited in size. To better evaluate its scalability, we used a large scale trace driven simulation with the data collected from the prototype implementation. The experiment results show that SASA achieves accurate collapse detection where over 80% of the detected holes are located within 1 meter from its real position, and 99+% are less than 2 meters. In the large scale simulation, SASA is proved to be scalable with controlled detection latency and acceptable misreport ratio.

The rest of this paper is organized as follows. Section 2 discusses related works. Section 3 introduces underground coal mine environment. Section 4 presents design details of SASA. Section 5 presents the performance evaluation through both trace-driven simulation and experimental results. Section 6 concludes this work.

2. RELATED WORK

Many WSN systems have been developed to support environment monitoring [13], object tracking [8, 9], scientific observation [25], and so on. The underground environment of our system differs from most previous systems in its varying geologic structures and conditions. Trying to capture and adapt to geologic structure changes, such as collapses, requires non-trivial solutions embedded in a sensor network system.

Hole problems in WSNs have been surveyed by Ahmed et al. in [2], which divides holes into four categories: coverage holes, routing holes, jamming holes and sink/black/worm holes. None of the works cited above correlate the sensor holes to physical structure variations, or discusses the holes caused by topology changes. Karp and Kung [11] propose the Greedy Perimeter Stateless Routing (GPSR) protocol, which aims to utilize nodes' location information to provide efficient routing in WSNs. It employs perimeter mode routing to forward packets around holes. Aiming at efficient routing, the GPSR does not localize the holes. Fang et al. [6] define stuck nodes and propose BOUNDHOLE to find the sensor holes utilizing the strong stuck nodes. Recently, Wang et al. [22] propose a topological method to detect the hole boundaries in sensor networks. However, they are both theoretical works with strong assumptions or simplifications on the network model. Several researches focus on event boundary estimations in WSNs. Nowak and Mitra [15] try to construct a hierarchical structure for detecting concerned phenomenal areas based on the multiscale partitioning methods. In [24], Wood et al. map the jammed area by constructing boundary outlines. Ding et al. [5] propose a localized event boundary detection algorithm, which takes randomly emerged faulty sensors into account and tries to eliminate their degradation of the detection process.

Being effective in ordinary environment, existing works do not consider the breakage possibly brought to the network during coal mine collapses. The network topology can be suddenly changed and sensor nodes in a collapse region may not function properly. Hence, directly employing those approaches in this collapse detection context will lead to poor detection accuracy and high power consumption.

3. APPLICATION SCENARIO

We cooperated with the S. H. Coal Corporation and selected the D. L. coal mine as our experimental environment. It is one of the most automated coal mines, yielding the second largest production of coal worldwide. The D. L. coal mine is a typical slope mine, as illustrated in Figure 1. A slightly sloped 14-kilometer long main tunnel starts from the entrance above the ground surface and goes 200 meters deep underground to the working bed. The main tunnel is the primary passage for miners and equipments.

The state-of-the-art means of underground mining – longwall mining technology – was adopted in the D. L. coal mine. Today, longwall mining accounts for about one third of all underground coal tonnage. In a continuous, smooth motion, a rotating shear on the mining machine moves back and forth across the face of a block of coal, cutting the coal. Coal drops onto a conveyor and is removed from the mine. Each longwall mining machine has a hydraulically operated steel canopy which holds up the upper strata and protects miners at the face. There are currently two 2 kilometer wide faces being mined.

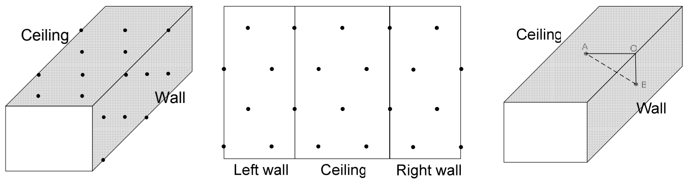
To monitor the underground environment in a coal mine, we designed and implemented the SASA system along the main tunnel and working spaces to fulfill the following requirements.

Remote management **-** Since it is preferable to remotely maintain and manage the entire monitoring system, efficient and robust communications and routing mechanisms are required under all conditions.

In-situ interactions - Providing geographical references could greatly facilitate locating miners underground. Besides stationary sensors deployed on the walls, poles and floors, miners carry mobile sensors as well.

Awareness of structural variations - One major goal of SASA is to instantly and accurately detect the collapse region. SASA aims to provide an infrastructural framework for underground monitoring with various environment sensors. For collapse detection, although we can achieve by equipping each node with acceleration sensors, it tends to make the system cost-inefficient. SASA achieves this goal through developing a node collaborating mechanism.

Maintenance of system validity - A collapse may change the system structure. Maintaining the validity of the monitoring system in extreme situations is necessary; robust service is expected. An efficient recovery mechanism is required.



(a) (b) (c)

Fig. 2. Sensor node deployment

4. SASA SYSTEM DESIGN

In this section, we present the design of the Structure-Aware Self-Adaptive sensor network, SASA.

4.1 Overview

In SASA, stationary sensor nodes are deployed on the walls and ceiling of tunnels to form a mesh network, as illustrated in Figure 2(a). To facilitate hole detection, SASA unfolds the two walls of the tunnel and builds a 2-D representation of the 3-D deployment on the inner surface of the tunnel, as depicted in Figure 2(b). The location pre-configured in each node is a 2-D location coordinate on the 2-D surface.

Nodes placed in the 3-D real environment are configured with 2-D coordinates on the unfolded 2-D surface. SASA conducts a transformation between the two locations with the knowledge of the longitudinal section of the tunnel such that the 2-D location uniquely corresponds to the 3-D location. In practice, the relationships between neighboring nodes in the 3-D real environment are the same as in the 2-D representation, except for a small area in corners where ceilings meet walls. As Figure 2(c) shows, the distance between any two nodes in the 3-D real environment is less than or equal to the distance between the pair in the unfolded 2-D view. Thus, the real connectivity of our sensor network is no less than shown in the 2-D representation. Later we will show that the neighbor set defined in our system in the 2-D representation is preserved in the 3-D real environment, and the correctness of the hole detecting algorithm is preserved.

In a real application, the sensors deployed in different tunnels are differentiated by being marked with different tunnel numbers. This way, holes in different tunnels can be identified.

We also require each miner to carry two sensors together with their regular devices. As the miners are moving, these mobile sensors are utilized to calculate miners' locations based on the stationary mesh nodes. This is crucial to the rescue operation when an underground accident happens. When any exceptional situation is detected, alarm messages are created and transmitted to the sink triggering an external safety system to inform operators outside the tunnel.

The hardware layer for our system is built on the widely used Mica2 mote platform [10], developed at UC Berkeley. The MPR400 radio board employed has a 7.3MHz microprocessor, with 128K bytes of program flash memory and 512K bytes of measurement flash memory. An 868/916 MHz tunable Chipcon CC1000 multi-channel transceiver with a 38.4 kbps transmitting rate is employed for wireless communication with a 500 foot outdoor range. A sensor board is connected to the Mica2 mote performing environmental data collection. The collected data is delivered to the Mica2 mote for further processing.

In this work, SASA focuses mainly on the construction and maintenance of the monitoring sensor network for collapse hole detection. According to statistics in coal mines, such a collapse may occur at any time and any place. Other functions such as gas and water monitoring are also supported by SASA but are outside the scope of this paper. The main functions of SASA include:

- ***Detecting and locating the collapse hole –*** This is the primary function of SASA. Successfully locating the hole region after collapse assists instant rescue and following repair.
- ***Accident reporting –*** The accident reporting messages need be rapidly and reliably routed from the collapse region back to the sink. SASA aims to provide a systematic solution for it.
- ***Displaced node detection and reconfiguration –***After the collapse, the original sensor nodes in the hole region may be relocated. The original location configurations of these nodes then become outdated, which may lead to incorrect location references and improper routing actions, thus reducing the stability and reliability of SASA system. Consequently, it is necessary to rapidly detect these nodes and reconfigure the nodes with correct locations in order to maintain system validity.

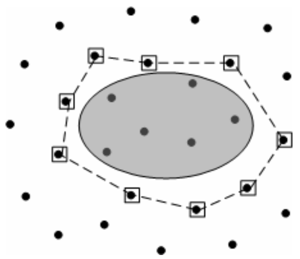


Fig. 3. The sensor hole and its outline nodes

4.2 Design rationale

In SASA, to get rapid and accurate detection of the collapse hole, we exploit the relation between sensors within and outside of the collapse region. SASA does not rely on any additional devices for achieving this task. Although equipping accelerometers for the sensor nodes might helps, it brings excessive cost for the system. Each accelerometer costs $50+ and is much more expensive for more tolerance (10g+) on impulse. Adding accelerometers in sensor nodes also complicates the design of hardware. SASA system aims to provide a framework for general monitoring applications. System efficiency will drop with any add-in block.

When a collapse occurs, the sensor nodes in the accident region are moved, and a hole of sensor nodes emerges. For a reasonable density of sensor node deployment, the sensor node hole should reflect the actual collapse hole to a certain degree. When the sensor hole emerges, as shown in Figure 3, the nodes on the hole edge will have a loss of neighbor nodes, and these nodes outline the hole.

The basic idea in detecting a hole is to let sensor nodes maintain a set of their neighbors. When a node suddenly finds that a subset of its neighbors has disappeared, it should be aware that it is now likely to be an edge node of a hole. A straightforward method of maintaining neighbor sets is to require that nodes periodically probe their neighbors. However, this approach is costly in terms of traffic overhead. To address this issue, we propose a beacon mechanism, which requires each node to actively report its existence. By carefully deploying the sensor nodes into a regular mesh network and determining a criterion for hole detection by neighbor losses, our algorithm can provide approximation of the collapse hole region through the *edge nodes* around the hole region. The hole region approximation is calculated in the sink. A data aggregation strategy is employed to reduce the instant traffic.

The node reconfiguration process afterward is divided into two phases: displaced node detection and node reconfiguration. In the displaced node detection phase, both centralized and decentralized mechanisms are employed to achieve short detection latency. In the node reconfiguration phase, a displaced node estimates its new location based on surrounding normal nodes. Iterative calculation is conducted to get an accurate estimation.

Besides these structure-aware behaviors, SASA provides mobile nodes localizations through our pre-deployed mesh sensor network. As many data gathering algorithms for WSNs have been proposed [11, 14, 19], and most of them could be applied in our SASA system, we will not discuss routing in this work. In this implementation, SASA simply employs the greedy mode of GPSR [11].

Many key issues have been examined in SASA design and implementation. Our discussion in this paper will focus on *node beaconing mechanism*, *hole detection*, *accident reporting*, as well as *displaced nodes detection and reconfiguration*, as follows.

4.3 Node beaconing mechanism

In order to monitor structural change, each node is responsible for inspecting its surrounding nodes. Intuitively, to require each node to dynamically probe its neighbor nodes is simple but inefficient. In the sensor network, a transmission between two nodes can only be achieved by node locally broadcasting. The broadcast creates a collision domain where all other nodes in this domain must remain silent in order to avoid collisions. If we consider the message broadcasting manipulation as the cost unit, the active probing strategy has a traffic cost of $O(nk)$, where n represents the network size, and k is the average number of neighbors per node. Replies from the neighbors are $O(k)$.

SASA adopts a beacon mechanism, in which nodes passively listen to their neighbors: each node periodically broadcasts beacon messages that include its location. This beacon mechanism benefits from the “wireless multicast advantage” (WMA) [23] in WSNs and could effectively reduce the traffic cost down to $O(n)$. To avoid collisions, we set a small random variation for the beacon interval, which prevents multiple nodes from broadcasting beacon messages simultaneously.

4.4 Hole detection

A node maintains a neighbor list in its memory. Upon receiving a beacon message, it updates the corresponding entry. A timer T_1 is then set to determine the entry expiration: an entry not updated by the time it expires represents the loss of the neighbor. In our experiment T_1 is set to be 3 times the beacon interval. Upon a collapse, nodes beside a hole become edge nodes. They are able to rapidly detect loss of neighbors.

However, regulating the neighbor set of a node is challenging because the RSS (Radio Signal Strength) value between nodes is highly dynamic in the coal mine environment making it hard to be an indicator. Consequently, a naïve method, in which all the nodes whose beacon messages could be received were taken as neighbors, failed in our prototype implementation experiment. It was observed that the neighbor set of a node is highly unstable, even if all the nodes worked normally. Also, nodes often had different sizes of neighbor sets, if initially the nodes were not regularly spaced. All of these factors made it hard to determine a criterion for detecting the hole via neighbor loss detection.

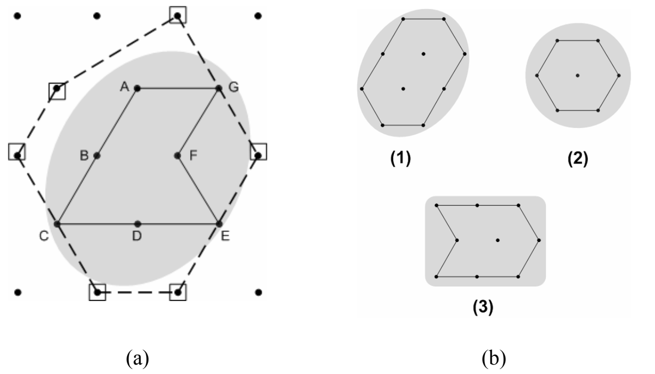


Fig. 4. (a) Hole and edge nodes; (b) Hole polygon examples

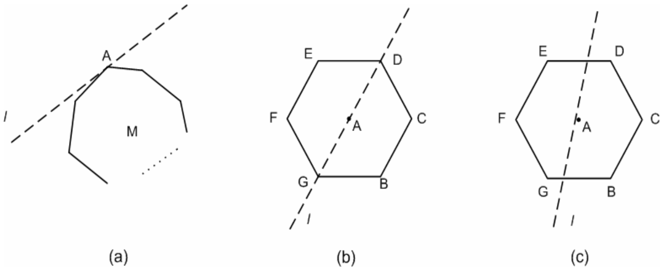


Fig. 5. (a) The convex hull of edge points and hole points. (b) & (c) Two cases of the relationship between line l and M

To address this issue, SASA deploys sensor nodes in a cellular hexagonal placement such that the node distribution is uniform, as illustrated in Figure 2(b). In the 2-D representation, every pair of adjacent nodes are separated by the same interval which can be varied from several meters to tens of meters determined by the size of detection area, required precision, and the signal range of sensor nodes. Every node (excluding boundary nodes), if taken as the center of a regular hexagon, has 6 adjacent nodes on the 6 vertices of the hexagon. In our experiment, we selected a 3 meter interval deployment. Keeping effective radio signals at 3 meters might result in maximum radio ranges of 4 to 5 meters (due to the individual differences in nodes) with interspaces [7]. Under this setting, a sensor node may receive beacon messages from nodes other than the 6 adjacent ones. However, in the neighbor list, we limit each node’s neighbor set to the 6 adjacent nodes, i.e. the nodes other than the 6 nodes will not be maintained in neighbor entries although their beacons may be received. Such a scheme of neighbor maintenance provides us a firm set of neighbors for each node and thus a regular method to determine the edge nodes.

Definition: A node defines itself as an *edge node* if the two adjacent neighbor nodes are detected lost during a time period T_2.

Another timer T_2 is set for determining the edge nodes. The timer T_2 is slightly larger than the timer T_1 of detecting neighbor loss. Upon a collapse, this criterion generates a set of edge nodes. These edge nodes act as landmarks to display the hole region.

Definition: *hole polygon* is defined as the largest polygon outlined by the collapsed sensor nodes with every edge ending at two adjacent nodes.

For example, the polygon *ACEFG* in Figure 4(a) forms a hole polygon. A hole polygon functions as a geometric representation for the hole region. We provide more examples of hole polygons in Figure 4(b). Since every edge node has at the least two neighbors in the hole polygon, each is at the most 2.6 meters away from the hole region, and the outline drawn by these edge nodes is at the most 2.6 meters away from the hole polygon. This gives an upper bound. We give a proof that the convex hull of the edge nodes (see figure 4(a)) encloses the hole polygon, which is the lower bound of the outline drawn by the edge nodes.

Theorem: The convex hull of edge nodes in SASA encloses the hole polygon.

Proof: We prove it by contradiction. To give a geometric abstraction, we refer to all the edge nodes as edge points, and all the vertices of the hole polygon as hole points.

Suppose there is at least one hole point outside of the convex hull of edge points. We draw a convex hull *M* of both the edge points and hole points as shown in Figure 5(a). There must be one hole point which is the hull point. Without loss of generality, suppose the point is A.

As shown in Figure 5(a), we can draw a line *l* across A such that all other points of *M* are on one side of *l*. This is guaranteed by the characteristic of a convex hull. Point A has two adjacent hole points on the hole polygon out of its 6 adjacent neighbor points. According to the relationship between line *l* and the 6 adjacent neighbor points of A, there are two cases as shown in Figure 5(b) and (c).

Case 1: In Figure 5(b), line *l* crosses two neighbor points. If *M* is bounded on the right side of *l* (it holds the same rationale as when *M* bounded on the left side of *l*), the two adjacent hole points of A can only be B and C. In this case, point D and G must be edge nodes since they both have two hole points as neighbors. This contradicts the assumption that point A is a hull point of convex hull *M*.

Case 2: In Figure 5(c), line *l* crosses no neighbor points. If we suppose *M* is bounded on the right side of *l*, the two adjacent hole points of A can only be either B and C or D and C. In both cases, either B or D is a hole point adjacent to point A, which makes either point E or G an edge node. Since both point E and G are outside of *M*, a contradiction is formed. Therefore, there is no hole point outside of the convex hull of the edge points, and thus the theorem.■

The algorithm for calculating the convex hull of *n* points has a computational complexity of $O(n \cdot logn)$. So it provides a lightweight method for the sink to achieve this bound.

In practice, multiple nodes breaking down in a region at the same time can be considered the result of a collapse, whereas a single node failure in a certain region is likely the result of a power off or node failure. Our hole detecting algorithm is made tolerant of the interferences from single node failures since the failure of at least two adjacent nodes are necessary to define an edge node. Nevertheless, if two adjacent nodes fail simultaneously, the algorithm fails. As a marginal effect, a small hole affecting only one sensor node can not be detected by this algorithm. This sets the threshold of the size of detectable holes. However, this threshold can be lowered down by increasing the density of deployed sensors.

The Mica2 motes adopted in SASA employ a CSMA transmitting protocol for multiple accesses in wireless communication channels. Although this protocol is effective for collision avoidance, collisions are still a major problem in a densely deployed sensor network due to the hidden terminal problem, especially when the communication density is high. Such collisions waste network bandwidths and greatly increase the packet loss rate.

To reduce collisions, SASA tries to maintain a comparatively low communication density, which is highly dependent on the beacon mechanism. A lower beacon density helps keep lower communication density while leading to a longer detecting latency. So how to balance this tradeoff is important and will be examined in the experiment section.

4.5 Accident reporting

When edge nodes detect a hole, they report to the sink with the locations of edge nodes so that the hole can be outlined by calculating the convex hull. A relatively effective but expensive approach is to deliver messages by flooding. When a collapse occurs, however, all the edge nodes might flood report messages at the same time, creating a traffic peak and increasing the collision probability. To reduce the collisions [16], we introduce 1) a randomized forwarding latency, and 2) a data aggregation strategy.

We insert a flag into the beacon messages that indicates whether the beaconing node is an edge node. The edge nodes wait a short time before sending out its report. Upon receiving other edge nodes' beacon messages, an edge node records them locally. When this edge node sends out its report message, it aggregates all the recorded locations of its nearby edge nodes in one report message. If an edge node receives a report message containing its own location, it is aware of the fact that other edge node has already aggregated its location. This node will simply forward this message instead of generating a new one. The total amount of traffic is thus reduced.

The sink reply is employed to maintain the reliable transmission of report messages. An aggregated reply message including all the received locations of edge nodes is flooded out from the sink. The edge nodes not included retransmit report messages. SASA limits the number of retransmissions so that the edge node will not keep transmitting report messages if it has been isolated from the sink. Such isolation is possible as the network can be disconnected by a large collapse. In our system implementation, this phase is simplified and merged into the *node reconfiguration* process.

4.6 Displaced nodes detection and reconfiguration

During a collapse, the sensor nodes in the hole region are displaced with new nodes surrounding them. When a node becomes an edge node, we also need to determine whether or not it has been moved. The other challenge is that not all the displaced nodes become edge nodes immediately after a collapse. For example, a node and all its neighbors may fall into one place to-

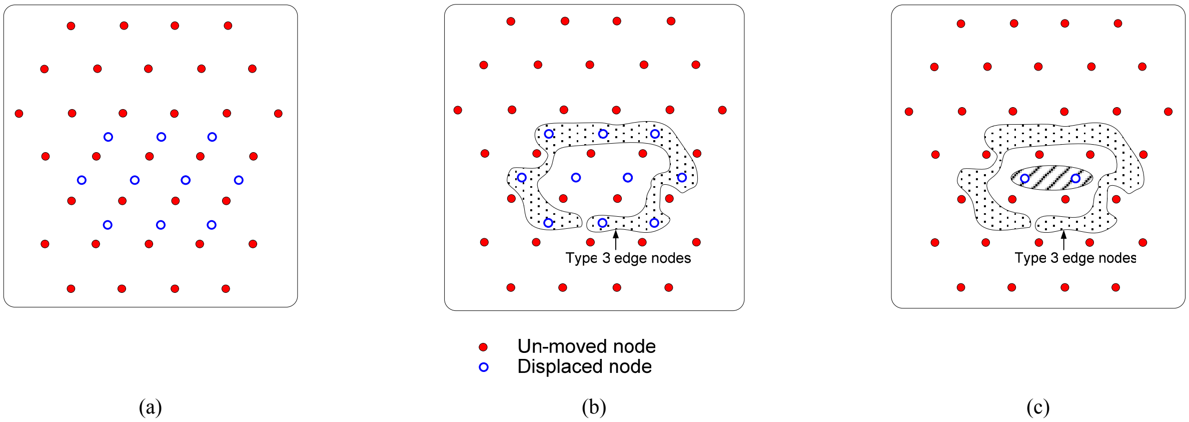


Fig. 6. An example for the distributed detection algorithm. (a) Several nodes fall into a place together with all its neighbors; (b) Type3 edge nodes stop beaconing and inner displaced ones find neighbor loss; (c) Inner nodes define themselves as edge node and knows displacement

gether, as shown in Figure 6(a). Since the inner-displaced nodes do not find any neighbor loss, they will not define themselves as edge nodes. In this application, we need to detect displaced nodes and reconfigure their locations.

The basic idea will be trivial if we utilize the global information. When the sink receives report messages with the edge nodes' locations and approximate the hole region, it broadcasts the convex hull area, informing the nodes in the hole region of their displacement. Every node within the convex hull will start detecting its surroundings and check its location from beacon messages. An average location can be calculated and compared with its own configured location. If the two locations differ beyond some threshold, it knows its displacement.

To shorten the message length and save power, SASA uses a rectangle enclosure to approximate the convex area, which costs 16 bytes to represent the 4 vertices and simplifies the calculation of each node, as illustrated in Figure 7. Though the approximation is less accurate, it is adequate to describe the hole.

The major issue of such a centralized approach is that it often suffers long latency and low accuracy due to the high link loss rate in coal mines, especially when a collapse area in a long tunnel is far from the sink. In extreme cases, the network could be disconnected by a large collapse, although such large collapses are rare according to the past 20 year history of the D. L. coal mine. Indeed, since the small scale collapses frequently precede the more dangerous and more easily located large scale collapses, we can use the detection of small collapses as an early warning, alert or indication of the possibility of a large collapse in order to evacuate or repair the dangerous area/structure. It is already too late when large collapses occur, so rapidly detecting and reporting small collapse locations are significant for coal mine safety. Hence, the primary focus of SASA is on locating small scale collapses.

In order to further reduce detection latency and improve accuracy, we also propose a distributed algorithm. Recall that the definition of *edge node* is a node that has lost at least two contiguous neighbors. There are three types of edge nodes as follows: 1) the edge nodes that lose neighbors but themselves do not move; 2) edge nodes that fall into an area where no normal node exists; 3) edge nodes that fall into other normal node range. For the type 1 nodes, their locations are correct, so they do not need any reconfiguration. For type 2 edge nodes, they have no impact on normal nodes, so they do not need any action as well. Indeed, a node cannot easily recognize whether it belongs to type 1 or to type 2.

So our focus is on type 3 edge nodes. A node defines itself as type 3 edge node if and only if: 1) it is an edge node and 2) it detects newly emerged neighbors. A type 3 edge node stops beaconing immediately, as illustrated in Figure 6(b). This operation will lead the neighboring displaced nodes to become edge nodes, if they are not yet, as shown in Figure 6(c). In a recursive manner, all the nodes removed from hole region will become edge nodes and detect their location variations.

The recovery latency is correlated with the recursive process, which may have several phases, so it is longer than that of the centralized algorithm when the collapse area is close to the sink. However, since it is a local algorithm, the recovery latency is independent of the distance to the sink. Combining the two detecting algorithms provides us efficient and reliable recovery for various situations. SASA employs both mechanisms.

When the displaced nodes are discovered, we can simply turn them off or reconfigure their locations to conform to their new positions. The SASA adopts node reconfiguration to conserve as many working nodes as possible to maintain an adequate node density. Although many schemes [12, 17, 18] have been proposed for localization in general WSNs, we find most of them infeasible in our context, since the highly dynamic radio signal strength in the underground environment makes it extremely difficult for the ranging operations of those schemes. We try to explore simple but effective solutions.

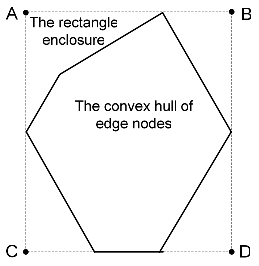


Fig. 7. The rectangle enclosure

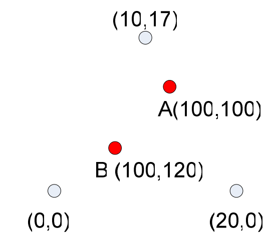


Fig. 8. Reconfiguration of A and B

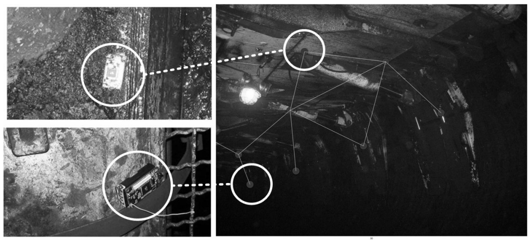


Fig. 9. SASA deployment

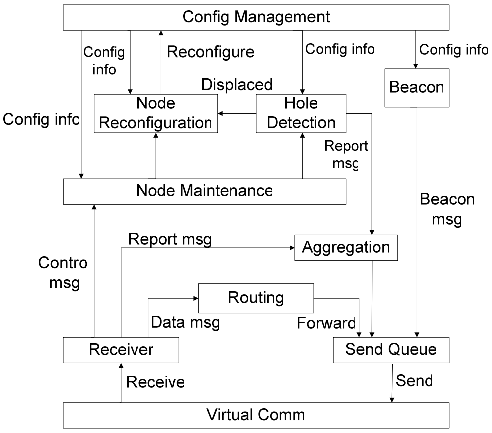
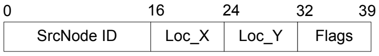


Fig. 10. The block diagram of the system architecture



SrcNode ID: Node ID of sender
Loc_X: Sender's x coordinate value
Loc_Y: Sender's y coordinate value
Flags:
Bit 0: Sender's working status - 0/1 normal working /reconfiguring
Bit 1: If the sender has been reconfigured - 0/1 N/Y
Bit 2: If the sender is an edge node - 0/1 N/Y
Bit 3 - 7: Idle

Fig. 11. Data payload format of beacon messages

If we let the nodes calculate average locations from surrounding nodes, as some of the surrounding nodes may also come from hole, the calculation could lead to an inaccurate result. Therefore, we design an iterative method for location calculation. Suppose two nodes, *A* and *B*, drop into a new area surrounded by 3 resident nodes as shown in Figure 8. Initially they have their original location. When node *A* first detects the surrounding four nodes, it calculates a new location as (32.5, 29.25) and replaces the original location. Then when node *B* detects its surroundings, it utilizes the new location of node *A* and calculates a new location as (15.63, 11.56). Thus, when node A iteratively calculates its new location, it will get a more accurate result of (11.41, 7.14). This iterative process continues and the calculated locations of node A and B tend to the center of the three original resident nodes, which is a close approximation for their new locations.

To accelerate the iterative calculation process, the nodes which are aware of their location variations stop beaconing until they have calculated their new locations. A bit of the beacon message is used to indicate whether the beaconing node is a reconfigured one.

Table I SYSTEM PERFORMANCE

Hole detection percentage (%)	100%
Average hole detection error (m)	0.73
Average reconfiguration 2D error (m)	0.87
Average reconfiguration 3D error (m)	2.62

5. EXPERIMENT AND PERFORMANCE

5.1 Prototype implementation

A prototype system with 27 Mica2 motes is implemented in the D. L. coal mine as illustrated in Figure 9. The system is distributed on a tunnel wall about 12 meters wide and 5 meters high. The motes are preconfigured with their location coordinates and placed manually at surveyed points with an interval of 3 meters as specified in our proposed hexagon mesh regulation. The *CC1000control* component of each Mica2 mote is adjusted such that when the motes broadcast beaconing messages, the maximum signal range is minimized in order to reduce collisions while guaranteeing desired 4 meter signal coverage. The signal range is increased for flooding or forwarding messages to maintain transmitting efficiency.

Figure 10 shows the block diagram of SASA architecture implemented in TinyOS on the Mica2 motes. The "Config Management" component manages the configuration information of the node, including the node ID and its configured location. The "Hole Detection" and "Node Reconfiguration" components are constructed on the "Node Maintenance" component, which deals with various control information from surrounding node beacon messages and the centralized control messages.

An indicator "*node_status*" is used to switch the system between the two working statuses: normally working (for hole detection) or in reconfiguration. The "Beacon" component periodically broadcasts the current config information of the sensor node with the message format shown in Figure 11. It is taken as application data payload in the TinyOS RF message with the destination of local broadcast *TOS_BCAST_ADDR* and the specified handler ID *AM_BEACONMSG* = 131.

For the analysis of the hole detecting performance in this experiment, 20 different sensor holes are selected from collapses recorded in S. H. Coal Corporation history. Their sizes range from 48m^2 to 132m^2. For each instance, we randomly redistribute the displaced sensor nodes from the hole region in the tunnel 10 times.

Table I presents the statistics of our system performance in the 200 testing samples. The metrics are defined as following.

The ***hole detection percentage*** reflects the effectiveness of the system in detecting the hole. A hole is counted as undetected if less than 3 edge node reports are received by the sink.

The ***hole detection error*** is measured by the distance between the real and detected position of the hole region. The position of the hole is represented by the geometric center of the hole region.

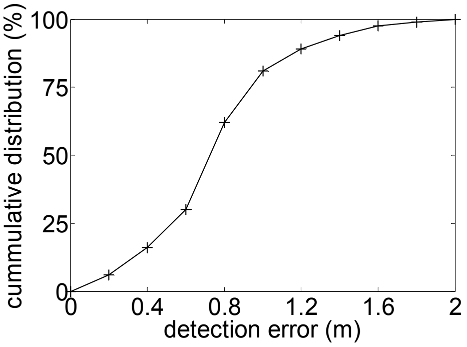


Fig. 12. Hole detection accuracy

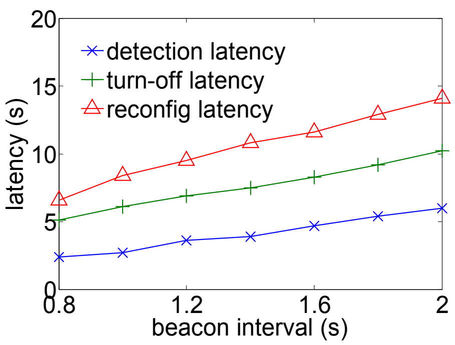


Fig. 14. Processing latencies VS. beacon interval

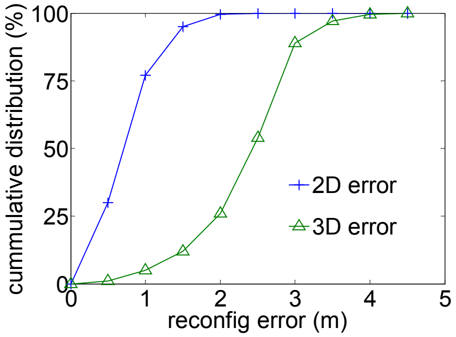


Fig. 13. Reconfiguration accuracy

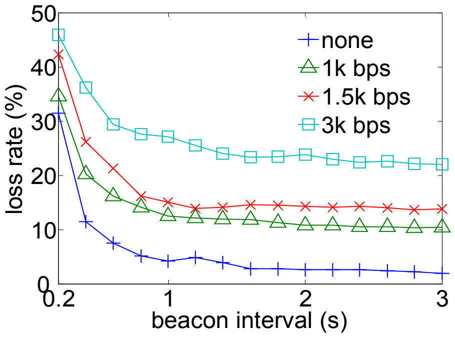


Fig. 15. Packet loss rate VS. beacon interval

The ***reconfiguration error*** is the localization error in the reconfiguration process. The 2D error is the error of the reconfigured node position on the 2D representation of the tunnel surface, and the 3D error is the error of the reconfigured node location in the 3D real space. Though both errors affect the system performance, the 2D error exerts a dominating effect on the system validity, and the 3D error degrades the accuracy for mobile node localization. More precise reconfiguration process achieves better system resilience.

Figure 12 plots the hole detection error where over 80% of the detected holes are located within 1 meter from its real position, and 99+% are less than 2 meters. The detection error comes mainly from the mismatch between the outlined hole region and the real hole region. The loss of report messages due to collisions also introduces error. Figure 13 plots the cumulative distributions of the 2D and 3D errors of node reconfiguration. We can see that all the 2D errors and over 80% of the 3D errors are below 3 meters.

A short beacon interval leads to short processing latency for both hole detection and node reconfiguration. Figure 14 plots three kinds of processing latencies against the beacon interval. The *detection latency* represents the time from when the hole emerges until it is detected. The *turn-off latency* represents the latency when we choose to simply turn off the detected displaced nodes, and the *reconfig latency* represents the latency when we choose to reconfigure the displaced nodes according to the normal nodes surrounding them. All three types of latencies are proportionally increased as the beacon interval increases. We observe that for each beacon interval, the *reconfig latency* is longer than the *turn-off latency*. This difference is due to the time needed for nodes to recursively calculate their new locations.

Figure 14 suggests a short beacon interval for pursuing short processing latencies. However, frequent beaconing brings large overhead, leading to heavy collisions and increased packet losses. In experiments, the communication quality between two neighboring nodes is tested for various beacon intervals under different traffic pressures. As shown in Figure 15, the packet loss rate rapidly drops as the beacon interval increases while under short beacon intervals (less than 0.8s), then becomes stable around a fixed level, and the loss rate is heightened as the exerted traffic overhead increases.

Based on these observations, we are able to carefully select a proper beacon interval for a specific application workload to balance communication quality and the processing latency. We can make a shorter beacon interval to reduce the processing latency if the application workload is light or make a longer beacon interval to reduce the packet loss rate if the application workload is heavy while the application is tolerant to the processing latency.

5.2 Simulation

The experiments on the SASA prototype present a partial image of our system performance, with some basic phenomena observed. In order to have a more extensive picture of the performance of SASA with thousands of sensor nodes and to evaluate its scalabil-

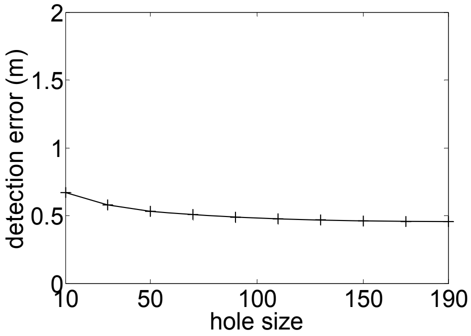


Fig. 16. Hole detection error VS. hole size

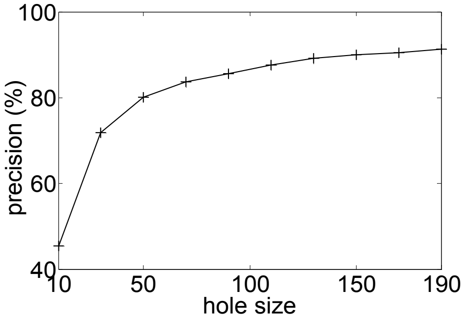


Fig. 17. Hole detection precision VS. hole size

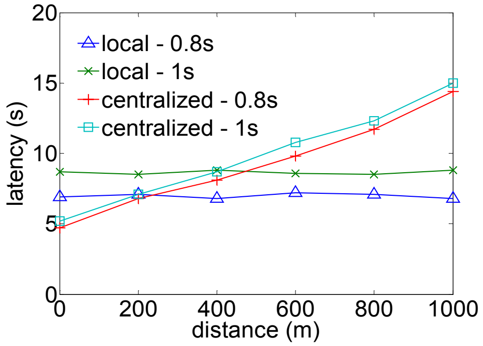


Fig. 18. Reconfiguration latency VS. hole distance

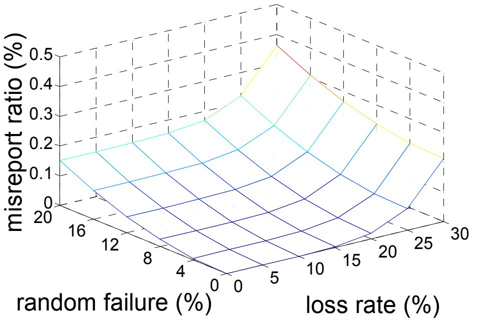


Fig. 19. Misreport ratio VS. packet loss rate & node failures

ity, we conduct a large-scale simulation based on the data collected from our prototype experiment.

In this trace-driven simulation, 2000 nodes were simulated on a 1000m × 20m plane with 3 meter interval in the hexagon mesh regulation. A transmitting rate of 16 packets/s is used in the simulation for the nodes' communication channels. This transmitting rate was selected based on data from our experiment on the Mica2 motes in the coal mine. The sizes of beacon messages and report messages are both 14 bytes including the headers. Each node is assumed to have a desired 4 meter transmitting range of beaconing, and 20 meter maximum communication radius when needed.

The hole detection accuracy is tested for various hole sizes. Figure 16 exhibits the detection error as the hole size varies. The detection error is stable and decreases slightly as the hole size increases. The error is kept below 0.7m for all trials. A larger hole includes more edge nodes, giving a more accurate outline of the hole region.

We define another metric, *hole detection precision* $p = D^2/H \cdot G \times 100\%$, where H and G represent the area of the convex hull of the hole nodes and the area of the outlined hole region by the edge nodes, respectively. D is the area of the overlaps of H and G. This metric describes the tightness of the outlined hole region. A tighter outline requires a more precise shape and size suiting the real hole region. Figure 17 plots the detection precision against the hole size. As we have discussed in Section 4.2, the outline drawn from the edge nodes is bounded within one hop from the hole nodes. When the hole size increases, the outline of the edge nodes actually becomes tighter, and the detection precision is dramatically increased. SASA achieves 80+% detection precision as the hole size is above 50.

In our next experiment, we compare the reconfiguration latency of the local algorithm and the centralized algorithm. A hole containing 30 nodes is presumed, located at different distances from the sink. The nodes in the hole are displaced to other places but kept unseparated, creating the worst case scenario for the local recovery algorithm in terms of convergence time. Two beacon intervals are tested (0.8s and 1s).

Figure 18 plots the results. Clearly, when the hole is close to the sink, the centralized algorithm benefits from rapid information collection and reaction from the sink, and has a shorter latency. When the distance of the hole increases, the reconfiguration latency increases linearly in the centralized algorithm, due to the increase of the round trip time from the sink. The local algorithm is not affected, and its latency is determined by the beacon interval. The hole distance of 200m (for 0.8s beacon interval) and 400m (for 0.9s beacon interval) set critical points between centralized and local algorithms. The local algorithm provides shorter processing latency when the hole distance is farther than the critical points. The combination of the two algorithms provides good reconfiguration latency for the whole distance axis.

Here we must mention that for the above three tests, the communication channel is assumed to have a packet loss rate of 15%, which comes from our prototype experiment. Apparently, such a constant communication quality is not always realistic in the real environment where the traffic distribution is imbalanced, especially in the edge node reporting phase where the report messages are triggered and congregated almost simultaneously. However, while the traffic model and interference relationship is obscure and hard to determine, we choose to simplify this influential factor and hope to gain an elementary knowledge of the characteristics of our scaled system.

The system stability is also investigated by varying the wireless channel loss rate and artificially introducing random node failure into the system. In Figure 19, the loss rate means the packet loss rate between any two communicating nodes, and the random failure rate is the ratio of artificially-introduced node failures per simulated minute. The misreport ratio increases as the two parameters increase. We thereby should decrease the beacon frequency in order to preserve a better communication channel.

6. CONCLUSION AND FUTURE WORK

In this paper, we discuss SASA, a Structure-Aware Self-Adaptive wireless sensor network system, for underground monitoring in coal mines. By regulating the mesh sensor network deployment and formulating a collaborative mechanism based on the regular beacon strategy, SASA is able to rapidly detect structural variations caused by underground collapses. The collapse holes can be located and outlined, and the detection accuracy is bounded. We also provide a set of mechanisms to discover the relocated sensor nodes in the hole region.

We deployed a prototype in the coal mine to test system validity. System error was measured during both the detection and reconfiguration processes. The detection latency, packet loss rate and network bandwidth were also measured. Based on the data we collected in experiments, we conducted a large-scale simulation to evaluate the system scalability and reliability.

Several issues remain to be addressed further. First, when a collapse occurs, the stationary mesh network could be ruined and become unreliable, then the mobile nodes carried on miners or tramcars could be utilized as intermediate supporters. How to organize mobile nodes to form efficient collaborative groups is a challenging issue. Second, the proposed mechanism only detects single holes. Since multi-collapses and aftershocks are possible and have happened in underground tunnels, extending this work beyond single-hole detection is of great importance. These works are currently in progress in our lab.

ACKNOWLEDGEMENT

The authors would like to thank the shepherd, Deepak Ganesan, for his constructive feedback and valuable input. Thanks also to Gerald J. Mileski, Elaine Ni, and anonymous reviewers for reading this paper and giving valuable comments. This work is supported in part by the Hong Kong RGC grant HKUST6152/06E, the HKUST Digital Life Research Center Grant, the National Basic Research Program of China (973 Program) under grant No. 2006CB303000, and NSFC Key Project grant No. 60533110.

REFERENCES

[1] " Mine Safety and Health Administration", http://www.msha.gov.

[2] N. Ahmed, S. S. Kanhere and S. Jha, "The Holes Problem in Wireless Sensor Networks: A Survey," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 9, pp. 4 - 18, 2005.

[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam and E. Cayirci, "A Survey on Sensor Networks," *IEEE Communications Magazine*, vol. 40, pp. 102 - 114, 2002.

[4] A. Chakrabari, A. Sabharwal and B. Aazhang, "Multi-hop communication is order-optimal for homogeneous sensor networks," in Proceedings of IPSN, 2004.

[5] M. Ding, D. Chen, K. Xing and X. Cheng, "Localized Fault-Tolerant Event Boundary Detection in Sensor Networks," in Proceedings of IEEE INFOCOM, 2005.

[6] Q. Fang, J. Gao and L. J. Guibas, "Locating and Bypassing Routing Holes in Sensor Networks," in Proceedings of IEEE INFOCOM, 2004.

[7] M. J. Feyerstein, K. L. Blackard, T. S. Rappaport, S. Y. Seidel and H. H. Xia, "Path loss, Delay Spread, and Outage Models as Functions of Antenna Height for Microcellular System Design," *IEEE Trans. on Vehicular Technology*, vol. 43, pp. 487 - 498, 1994.

[8] C. Gui and P. Mohapatra, "Power Conservation and Quality of Surveillance in Target Tracking Sensor Networks," in Proceedings of ACM MobiCom, 2004.

[9] G. He and J. C. Hou, "Tracking targets with quality in wireless sensor networks," in Proceedings of IEEE ICNP, 2005.

[10] J. Hill and D. Culler, "Mica: A Wireless Platform For Deeply Embedded Networks," *IEEE Micro*, vol. 22, pp. 12 - 24, 2002.

[11] B. Karp and H. T. Kung, "Greedy Perimeter Stateless Routing for Wireless Networks," in Proceedings of ACM MobiCom, 2000.

[12] L. Lazos, R. Poovendran and S. Capkun, "Robust Position Estimation in Wireless Sensor Networks," in Proceedings of IPSN, 2005.

[13] A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," in Proceedings of WSNA, 2002.

[14] T. Melodia, D. Pompili and I. F. Akyildiz, "Optimal Local Topology Knowledge for Energy Efficient Geographical Routing in Sensor Networks," in Proceedings of IEEE INFOCOM, 2004.

[15] R. Nowak and U. Mitra, "Boundary Estimation in Sensor Networks: Theory and Methods," in Proceedings of IPSN, 2003.

[16] V. Rajendran, K. Obraczka and J. J. Garcia-Luna-Aceves, "Energy-efficient collision-free medium access control for wireless sensor networks," in Proceedings of ACM SenSys, 2003.

[17] S. Ray, R. Ungrangsi, F. D. Pellegrini, A. Trachtenberg and D. Starobinski, "Robust Location Detection in Emergency Sensor Networks," in Proceedings of IEEE INFOCOM, 2003.

[18] A. Savvides, C. Han and M. B. Srivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in Proceedings of ACM MobiCom, 2001.

[19] S. Subramanian and S. Shakkottai, "Geographic Routing with Limited Infomatin in Sensor Networks," in Proceedings of IPSN, 2005.

[20] S. Vural and E. Ekici, "Analysis of hop-distance relationship in spatially random sensor networks," in Proceedings of ACM MobiHoc, 2005.

[21] C. Y. Wan, S. B. Eisenman, A. T. Campbell and J. Crowcroft, "Siphon: overload traffic management using multi-radio virtual sinks in sensor networks," in Proceedings of ACM SenSys, 2005.

[22] Y. Wang, J. Gao and J. S. B. Mitchell, "Boundary Recognition in Sensor Networks by Topological Methods," in Proceedings of ACM MobiCom, 2006.

[23] J. E. Wieselthier, G. D. Nguyen and A. Ephremides, "On the Construction of Energy-Efficient Broadcast and Multicast Trees in Wireless Networks," in Proceedings of IEEE INFOCOM, 2000.

[24] A. D. Wood, J. A. Stankovic and S. H. Son, "JAM: A Jammed-Area Mapping Service for Sensor Networks," in Proceedings of RTSS, 2003.

[25] N. Xu, S. Rangwala, K. K. Chintalapudi, D. Ganesan, A. Broad, et al., "A Wireless Sensor Network for Structural Monitoring," in Proceedings of ACM SenSys, 2004.