JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 24, 393-410 (2008)

Design and Implementation of Mobile Robot for Nodes Replacement in Wireless Sensor Networks*

JANG-PING SHEU, KUN-YING HSIEH+ AND PO-WEN CHENG+
Department of Computer Science
National Tsing Hua University
Hsinchu, 300 Taiwan
+*Department of Computer Science and Information Engineering*
National Central University
Chungli, 320 Taiwan

Most wireless sensor networks consist of a large number of static, low-power, short-lived, and unreliable sensors. In this paper, we considered sensor networks consisting of both static and mobile nodes. Integrating both types of devices enables new applications, such as nodes replacement, hole and partition recovery, and autonomous deployment and redeployment. We designed a smart mobile robot and implemented an application of nodes replacement to demonstrate its use, via our nodes replacement algorithm. In this algorithm, the mobile robots can navigate towards low-energy sensor nodes and replace them automatically, with new sensor nodes, having no location information. The navigation algorithm is based on received signal strength between the mobile robot and the communicating node. The experimental results confirm that the mobile robots successfully achieved their assigned tasks.

Keywords: mobile robot, mobile sensor, navigation algorithms, sensor networks, wireless networks

1. INTRODUCTION

Wireless Sensor Networks (WSNs) [1] are becoming an increasingly important technology, due to their remote environment monitoring capabilities. Such networks can greatly improve the accuracy of information, through the collaboration of a group of sensor nodes. Sensor nodes monitor interesting events within their sensing region, sharing their collected data and reporting their observations to a sink node, thus making meaningful information available at that sink node. Users can then retrieve useful data from the sink node, in order to monitor the status of the sensing regions. WSNs can be used in a variety of applications, such as military surveillance, health monitoring and scientific investigations in harsh physical environments [2, 3].

Sensor nodes can be classified into static sensor nodes and mobile sensor nodes. Current research in wireless sensor networks has focused on fixed sensor networks, in which the nodes are static. Static sensor nodes cannot change position by themselves, after they have been placed in the sensing area. On the other hand, mobile sensor nodes

Received December 19, 2005; revised March 23, 2006; accepted April 19, 2006.
Communicated by Yu-Chee Tseng.
* This work was supported by the National Science Council of Taiwan, R.O.C. under grants No. NSC 94-2213-E-008-023 and NSC-96-2752-E-007-003-PAE.
* The preliminary version has appeared in IEEE International Conference on Wireless and Mobile Computing, Networking and Communication WiMob'2005.

can change position autonomously, depending on their mission requirements. They are able to dynamically adjust network topology and promote the performance of sensor networks. Sensor nodes are usually distributed over a vast area, such as in disaster areas or harsh remote environments, where it is difficult for people to function. In particular cases, mobile sensor nodes are required to accomplish many difficult tasks, such as nodes replacement, location assignments, hole and partition recovery, autonomous deployment and redeployment, and dynamic sensing [3-7].

Some research efforts have been carried out on the implementation of mobile sensor nodes [8-11]. All of them used the Motes [12] series of products as their central processing control and communication units. These mobile robots provided convenient platforms for investigating related algorithms, and applications of distributed sensing, in mobile sensor networks. In addition, many studies have proposed novel applications, using mobile sensor nodes. For example, a bidding protocol [13] was proposed to help mobile sensors to heal coverage holes. Although static sensors may not be evenly deployed in covering the whole sensing area, mobile sensor nodes can move from dense areas to sparser areas, thus improving overall coverage. The authors in [6] proposed an algorithm to dynamically sense an unknown environment, using a single mobile sensor node. Mobile sensor nodes are continually moving, in order to constantly observe all points in the environment. Such applications can be used in many contexts, including urban search and rescue in the aftermath of a natural or man-made disaster. The authors in [4, 5] proposed a location estimation algorithm, using a mobile sensor node, equipped with GPS as a location information reference point; this mobile sensor node moved around the entire sensor network, periodically broadcasting its own coordinates to static nodes in the vicinity. These static sensor nodes were then able to estimate their approximate locations, using the received coordinates. Eventually, all the static sensor nodes would have their own locations.

Our research also included mobile sensor nodes. We designed and implemented a smart mobile robot, which was not only mobile, but was also equipped with wireless communication. In addition, we designed a navigation protocol to implement the application of sensor nodes replacement. The nodes replacement scheme can be used in sensor networks consisting of battery-powered sensor nodes, whose batteries may be difficult to recharge; these sensor nodes have heavy workloads and their energy is easily exhausted. Failure of a set of sensor nodes, within a network, because of energy depletion, can lead to sensor network partition and a potential loss of critical information. Thus, we attempted to use mobile robots to find these low-energy sensor nodes and replace them with new ones. Navigation is a fundamental problem in mobile robotics. A number of solutions [14, 15] have been proposed to resolve this problem. All of the approaches have assumed, however, that a map of the environment was available in advance. Our navigation protocol, on the other hand, allows the mobile robots to navigate without a map or location information. We simply used the received signal strength of the mobile robot to navigate to the target node. Our experiments have demonstrated that mobile robots can reach target nodes accurately and quickly, using our navigation protocol.

The remainder of this paper is organized as follows. Section 2 describes the system design and organization of the mobile robot. Section 3 presents the navigation algorithms applied to nodes replacement. Section 4 presents the experimental results of our navigation protocol. Finally, our conclusion is given in section 5.

2. ARCHITECTURE OF MOBILE ROBOT

In this section, we describe the architecture of the mobile robot including hardware and software implementations. Mobile robots are built from off-the-shelf components offered by Motes. Motes are a series of products for WSNs, designed by UC Berkeley, and produced by Crossbow Technology, Inc [12]. UC Berkeley also designed an event-driven operating system, TinyOS [16], and a new language, NesC for embedded sensor networks. We used the Motes MICA2 and MICA2DOT to implement our mobile robots and sensor networks, which can provide the functions of computation, communication, and sensing. In order to imbue the sensor nodes with mobility, we designed a single circuit board to drive the motors. Fig. 1 shows the block diagram of our system architecture. MICA2 was the main component supporting computation and communication capabilities for the mobile robots. It could process sensing data from the sensor boards and control motors through a motor board. We used separate power units to supply the MICA2 and the motors, in order to increase the stability of the system.

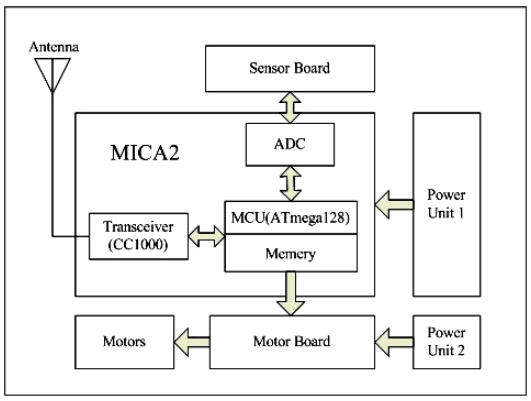


Fig. 1. Hardware architecture of mobile robot.

One of the most famous WSNs products is Motes. UC Berkeley and Crossbow Technology, Inc. cooperate to develop such a system for research in WSNs. There are many research institutes using Motes to implement wireless sensor networks. The series of products of Motes include MICA, MICA2, and MICA2Dot. We use the latest version of Motes: MICA2 and MICA2DOT in our system. Both of them have the same central processor and transceiver, but MICA2 could integrate with more kinds of sensor board than MICA2DOT.

2.1 MICA Motes and Motor Board

Our mobile robot used a MICA2 platform as its central processing and radio unit. The central processor of the MICA2 was ATmega128L running at 8 MHz. This microcontroller is a low-power AVR 8-bit processor with 128 Kbytes of flash memory, 4 Kbytes of EEPROM, and 4 Kbytes internal SRAM. The microcontroller also includes an 8-channel 10-bit ADC, three timers, and several bus interfaces including SPI, I2C, and

two USARTs. It consumes 8mA power in normal mode and less 15μA in sleep mode. The low power property suits to develop a sensor node.

The MICA2 Motes were designed for large-scale WSNs, and therefore, a variety of sensor boards are commercially available to integrate with the MICA2. Those are flexible sensor boards with a variety of sensing modalities. These modalities can be exploited in developing sensor networks for a variety of applications including vehicle detection, low-performance seismic sensing, movement, acoustic ranging, robotics, and other applications. Optional sensor boards include light and temperature sensing, a Honeywell HMC1002 2-axis magnetometer, an accelerometer, a 4 kHz sounder and a microphone.

The transceiver of the MICA2 was the ChipCon model CC1000 single-chip RF transceiver. The frequency was selectable within 433 MHz and 915 MHz bands. Maximum transmission range was 500 feet. The data rate was up to 76.8kbit/s depending on modulation techniques. The transceiver consumed 27mA in the transmitting mode and 10mA in the receiving mode; the power consumed in sleep mode was only 1μA. The radio on the MICA2 supported 26 output power levels and measured the received signal strength. These properties help to conserve energy of sensor nodes.

Our motor board was designed to control the motors in the mobile robot. The output control signals from the MICA2 were digital signals, which cannot drive motors directly. These digital signals had to be converted to analog signals, by the motor board, to be able to drive the motors. As shown in Fig. 2, MICA2 controls two motors through the motor board.

Before designing the motor board, we should select suitable motors to be used in mobile robots. Direct current (DC) motor and step motor are the most popular motors. Both of them are used for driving the mobile robots. There are many different properties of the DC and step motors. Step motors cannot suit to mobile robots because they are more energy consuming, expansive, and big size. Therefore, we choose DC motors to drive the mobile robots. We used the TOSHIBA TA7279P IC to control and drive the DC motors. The TA7279P IC was able control two separate DC motors in four modes (forward rotation, reverse rotation, stop, and brake), by using their bridge driver, which is best suited for switching between forward and reverse rotations. This IC could deliver an output current of 1.0 A in average, and 3.0 A in peak, conditions.

2.2 Platform of Mobile Robot

The platform of a mobile robot is similar to that of a tank. Hence, they can move on many different planes and have a small rotation radius, these features being good for outdoor WSN applications. The platform for the mobile robot supplied mobility, as well as power, to drive the mobile robot, being one single unit supporting the tracks, battery pack, gear box, and two motors, as shown in Fig. 3. The mobile robot had two separate power supplies. One power supported the MICA2 platform and another was supplied by 3 AAA batteries for the motors. Since the DC motors require higher current and cannot obtain the sufficient power from the battery of MICA2, a separate power supplier is used to drive the DC motors. We designed the base of the mobile robot as a single piece of material to increase its stability and reliability. To minimize the weight of the mobile robot, we chose lightweight aluminum material to construct the body.

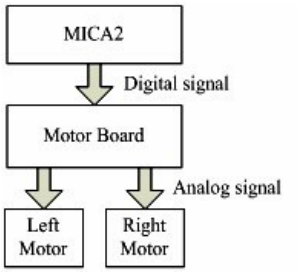


Fig. 2. Block diagrams of a motor board.

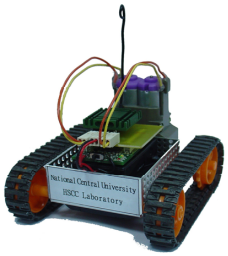


Fig. 3. Our mobile robot.

Mobile robots are programmed by using TinyOS which is based on the event-driven operating system developed at UC Berkeley for sensor networks. The TinyOS operating system, libraries, and applications are all written in nesC, a new structured component-based language. The nesC language is primarily intended for embedded systems such as sensor networks. The nesC has a C-like syntax, but supports the TinyOS concurrency model, as well as mechanisms for structuring, naming, and linking together software components into robust network embedded systems. The principal goal is to allow application designers to build components that can be easily composed into complete, concurrent systems, and yet perform extensive checking at compile time.

3. NAVIGATION PROTOCOLS

In this section, we introduce how the mobile robots replace low-energy sensor nodes with new sensor nodes. Since no location information existed in these static sensor nodes, the mobile robots did not know the location of the target node. Thus, we proposed a navigation algorithm to guide the mobile robots towards the low-energy sensor nodes via received signal strength. The mobile robot could utilize the received signal strength from a sensor node to approach its neighboring node. The mobile robot used this method to navigate from the sink node to the destination node, according to the routing sequence path.

In the following subsections, we describe how we created a routing path from any sensor node to the sink node and a navigation path from any low-energy sensor node to sink node. Each sensor node can send packets, through the routing path, to request help when its energy is low. Every low-energy sensor nodes will create navigation path simultaneously as it requests help. Thus, we present a navigation algorithm to allow the mobile robot to navigate to the low-energy sensor node, through the navigation path. Finally, we discuss how to serve several low-energy requests, simultaneously.

3.1 Initialization

In this subsection, we present a simple way to create a routing path from any sensor node to the sink node, by flooding. When its remaining energy is low, each sensor node uses the routing path to request help. The mobile robot can navigate to help through the routing path. We assumed that the sensor network was connected and had no location information, with each sensor node having a unique ID. There was one sink node, with

several mobile robots, which had the ability to install new sensor nodes. The tasks of the mobile robots were to move to the low-energy sensor nodes and replace them by deploying new ones.

When a sensor node detected that its energy was nearly exhausted, it would send a message to the sink node. We utilize flooding to create routing paths from every sensor node to the sink node. At the network startup stage, the sink node broadcasts a "*RouteCreate*" packet to the whole network, as shown in Fig. 4. When a "*RouteCreate*" packet was received, each sensor node recorded the ID of the sender, which was its next hop (up-link) to the sink node. If a sensor node received several "*RouteCreate*" packets from different sensor nodes, it kept the sender ID having the least number of hops to the sink node and rebroadcasts the packet. Except this, all other packets would be dropped. For example, in Fig. 4, node *F* recorded node *E* as the next hop to the sink node. Eventually, we were able to create a routing path, from every sensor node to the sink node.

After creating the routing paths, each sensor node can send a "*Help*" packet to the sink node for node replacement, according to the created routing path. The "*Help*" packet will record the ID of each node passed along the routing path. In Fig. 5, assume that node *F* is a low-energy sensor node. It will send a "*Help*" packet to the sink node through the created routing path. The "*Help*" packet will record the path from node *F* to node *A* that is <*F*, *E*, *D*, *C*, *B*, *A*>. Thus, a navigation path from the sink node to node *F* has been created.

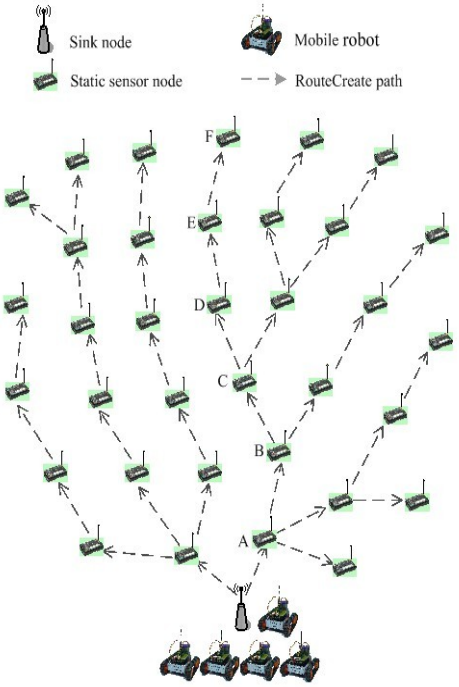


Fig. 4. Creating a routing path from each sensor node to the sink node.

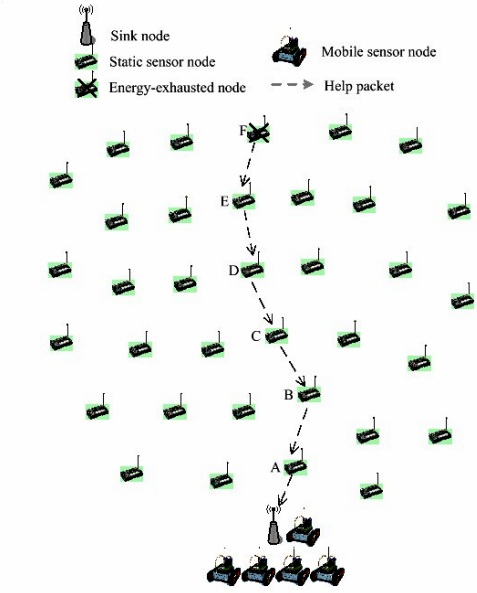


Fig. 5. Creating navigation path from sink to the low-energy node *F*.

Algorithm 1 Create Routing Paths

Sink node
Begin
Broadcast *RouteCreate<sink*, 1> packet;
End

Static sensor node
Begin
Hopcount_to_sink = ∞;
When receiving *RouteCreate<sender*, *hop_ count>* packet **do**
If the *hop_count* < *Hopcount_to_sink* **Then**
Hopcount_to_sink = *hop_count*;
Nexthop_to_sink = *sender*;
Rebroadcast the *RouteCreate<ID*, *hop_ count* + 1>;
Else
Discard the received packet;
End if
End

Algorithm 2 Create a Navigation Path

Notations:
path[]: an array of recording the ID list from the low-energy sensor node to sink node;
Help<path[]>: a packet contains the navigation path;
Sink node
Begin
When receiving a *Help<path[]>* packet **do**
Send the *path[]* to a mobile robot and perform Algorithm 3;
End

Static sensor node
Begin
When a node detecting its energy is low **OR** receiving a *Help<path[]>* packet **do**
Add its node ID to *path[]*;
Send the *Help<path[]>* packet to the next hop node;
End

3.2 Navigation between Two Sensor Nodes

The mobile robot used the received signal strength to navigate from one node to another; it could, therefore, move to the next sensor node by continuously monitoring the signal strength of the beacons sent from the node at the next hop. Eventually, the mobile robot was able to reach its destination. The received signal strength decreased as the distance increased. On average, signal strength decreases with distance according to the following equation [17]:

$$P_{receive} = P_{transmit} \frac{const}{x^{\alpha}},$$

where $P_{receive}$ and $P_{transmit}$ are the power of the received and transmitted signals, respectively. The distance between the receiver and transmitter is x meters and α is an exponent that characterizes the steepness of the decrease. The value α depends on the propagation environment. The value of α is 2 when radio waves propagate in free space. There is usually additional attenuation in wireless networks, which brings $\alpha > 3$ with the precise value dependent on terrain and other environmental factors, such as buildings. In metropolitan areas, $\alpha = 4$ is often used in this expression. According to this equation, the mobile robot will receive weaker and weaker signal strength as it gradually moves away from the sender, just as the received signal strength will become stronger as it approaches the sender. We can use this feature to determine whether the mobile robot is approaching or retreating from the sender. The mobile robot can use the received signal strength to navigate towards the sender.

Here, we illustrate how the signal strength is used to navigate from one sensor node to another. First, we have defined a turning point to be used in our navigation protocol. The location, where the mobile robot can receive the maximum signal strength value along a straight line, is within a block of the line. Therefore, we chose the midpoint of this block to be our turning point. We can then find the turning point in any straight line, according to the received signal strength. In Fig. 6, the mobile robot received the strongest signal strength at the turning point of the straight line L. Note that, the line connects the chosen turning point and the communicating sensor node may not perpendicular to the straight line L. This is because the signal strength is affected by the environment such as fading and multi-path. Thus the signal strength detected by the mobile robot is not accurate.

When the mobile robot wants to approach one sensor node, it asks the sensor node to send a short beacon packet within a fixed period. The mobile robot can then use the received signal strength from sender to find the turning point in its moving line. Initially, the mobile robot will go forward at will and detect the changing received signal strength. If it senses that the received signal strength is increasing in its moving direction, this means that it is approaching the turning point; otherwise it is moving away from the turning point. At this moment, the mobile robot will immediately reverse the direction in which it is moving, in order to approach the turning point. After arriving at the turning point, it knows that the sender is either to the right or left side of the moving line. In our algorithm, we chose first, to turn right. The mobile robot then searches the next turning point in its moving line, after changing direction. By repeating this procedure, the mobile robot will eventually approach the target node.

The algorithm is described below with a specific example. Fig. 7 illustrates a mobile robot moving from the sink node to its neighboring node A. First, the mobile robot goes forward in an arbitrary direction. Then, in step 1, it senses that the received signal strength from node A is decreasing as it moves in its chosen direction. This means that the mobile robot is moving away from the turning point. The mobile robot then brakes and turns to go in the opposite direction, in step 2. It continues in this direction as long as the received signal strength is increasing, in step 3. In step 4, the mobile robot receives the strongest signal strength at the turning point. If the mobile robot continues to go forward, the received signal strength will begin to decrease, in step 5. Finally, it returns to

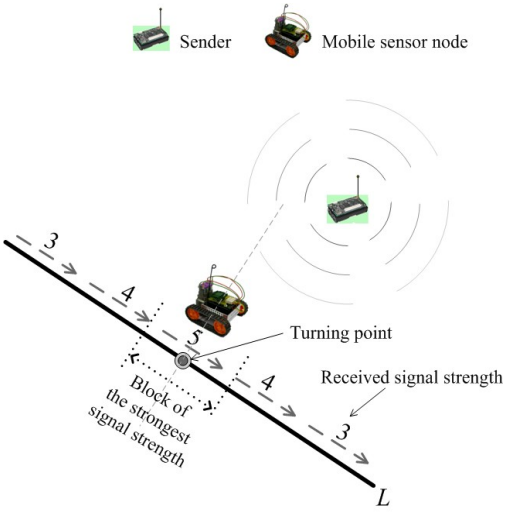


Fig. 6. Turning point of a straight line *L*.

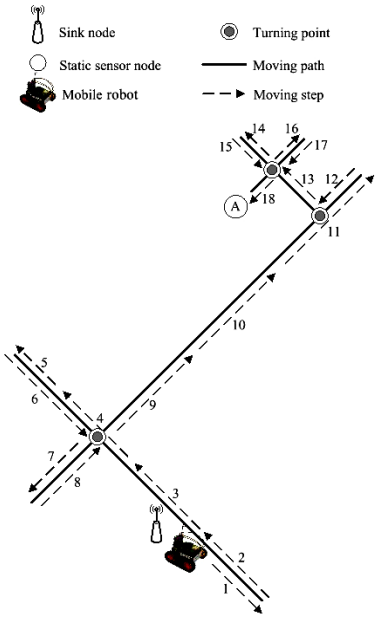


Fig. 7. The moving steps of a mobile robot from the sink node to its neighboring node *A*.

the turning point, which has the largest received signal strength, in step 6. At this moment, the location of node *A* is either on the right or left side of the straight line. In our algorithm, we chose to turn right, as shown in step 7. In this case, the mobile robot discovers that the direction is wrong, because the received signal strength is getting weaker as it moves in this direction. It brakes and goes back immediately, in step 8. Repeating this procedure, the mobile robot reaches the next turning point in its path after turning right. Eventually, the mobile robot approaches node *A*.

In our navigation protocol, we adopted two transmission power levels for the sensor nodes to navigate the mobile robot. Using low transmission power not only saves energy for the sender, but can also allow the mobile robot to navigate close to the sender. This is because the received signal strength is more sensitive to low-power levels. In our protocol, we changed maximum-power transmission to a low-power level as the received signal strength was equal to the maximum value. An example is shown in Fig. 8. When a mobile robot senses that the received signal strength is equal to the maximum value of the maximum-power transmission, it will ask the sender to change the transmission power to low-power level. As the mobile robot senses that the level of received signal strength is equal to the maximum value in the low-power level, this means that the mobile robot has approached the target sensor node.

Here, we illustrate how a mobile robot navigates to a low-energy sensor node, through multiple hops. If the sink node receives a "*Help*" packet from a low-energy sensor node, it sends the navigation path to the mobile robot. Then, the mobile robot sends a "*Notify*" packet to the first sensor node of the navigation path. After the sensor node receives the "*Notify*" packet, it sends a beacon packet, with the maximum-power level,

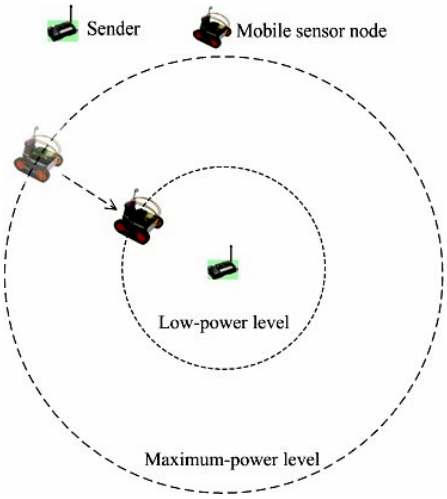
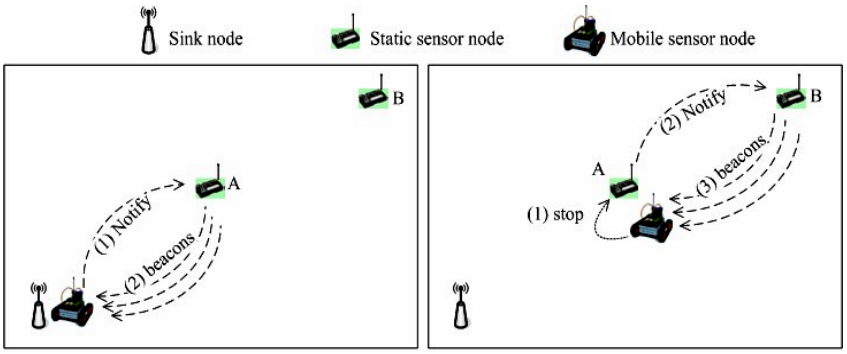


Fig. 8. Adjust power level when navigating the mobile robot.



(a) (b)

Fig. 9. Messages exchange when navigating node by node.

to the mobile robot, within a predefined period (one second in our experiments). The mobile robot uses the received signal strength to approach to the sensor node as shown in Fig. 9 (a). If the mobile robot senses that the received signal strength from the current node is equal to the maximum value in the maximum-power transmission level, then the mobile robot asks the current sensor node to transmit the beacon with the low-power level. As the mobile robot received the maximum signal strength again, it will ask the current node to stop sending the beacon packets and sends a "*Notify*" packet to the next hop node as shown in Fig. 9 (b). After this, the mobile robot can move to the low-energy sensor node, hop by hop, along the navigation path.

Note that, in order to reduce the navigation time, the mobile robot can move toward the next hop node and bypass the current guiding node if it can receive the signal sent from the next hop node. In our protocol, the mobile robot will send the "*Notify*" packet in every few seconds to the next hop node during it approaches the current sensor node by using the low-power level. This is because the mobile node is near by the current guiding

node and has possibility near to the next hop. If the mobile robot can receive the acknowledgement from the next hop node, it asks the current sensor node to stop sending the beacon packets and then moves to the next hop node directly. Therefore, the mobile robot can directly navigate to the next sensor node, without moving hop by hop from sink to the destination node.

Algorithm 3 Navigating Mobile Sensor Node to the Low-Energy Sensor Node

Mobile robot
Input: navigation path: *path[]*
Begin
Let *i* = size of *path[]*;
flag = 0; /* used to identify whether the received *RSSI* of mobile robot different from the previous one */
Mobile robot moves in an arbitrary direction and sends a *Notify* packet to the *i*th node in *path[]*;
When receiving a *beacon* packet from the *i*th node **do**
Case 1: *RSSI* = 7 /* the strongest received signal strength */
If the transmission *beacon* packet is sent by maximum-power level **Then**
Send an *AdjustPower* packet to inform the *i*th node to change transmission power to low-power level;
Else
If *i* = 1 **Then** /* The mobile robot arrives at the destination node. */
Send a *Stop* packet to inform the destination node to stop the *beacon* packets;
Else
Send a *Stop* packet to the *i*th node;
i = *i* – 1;
Send a *Notify* packet to the *i*th node of the *path[]*;
End if
End if
Case 2: *RSSI* < 7
If the received *RSSI* is larger than the previous one **Then**
Set *flag* = 1;
End if
If the received *RSSI* is equal to the previous one **Then** do nothing;
If the received *RSSI* is less than the previous one **Then**
If flag = 0 **Then**
Reverse the moving direction;
Set *flag* = 1;
Else
Return to an estimated turning point and turn the moving direction of mobile robot to right;
Set *flag* = 0;
End if
End if
End

Static sensor node
Begin
Case 1: when receiving a *Notify* packet
Send *beacon* packets to the mobile robot in every second with the maximum-power level;
Case 2: when receiving a *Stop* packet
Stop to send *beacon* packets to the mobile robot;
Case 3: when receiving an *AdjustPower* packet
Send *beacon* packets with the low-power level;
End

3.3 Handling Multiple Requests

A sensor node will send a "*Help*" request to the sink node immediately, when its remaining energy is low. After the sink node receives a "*Help*" packet, it can wait a period of time P to see whether other sensor nodes have the same request. Assume there are m mobile robots and n requests in a period of time P. The sink node can assign a mobile robot to serve all the n requests (one-to-many service) or assign n mobile robots to serve the n requests simultaneously (many-to-many service) if $m \geq n$. There is trade-off between the one-to-many and many-to-many services. In the many-to-many service, signal interferences may be incurred due to multiple nodes sending beacons' packets at the same time. In addition, multiple mobile robots consume more power energy. The many-to-many service has a shorter service time than the one-to-many service, however.

Below, we have proposed a greedy method to serve multiple requests, using a single mobile robot. When a mobile robot receives n requests from the sink node, it will first serve the sensor node with the least hops to the sink node. After the mobile robot has moved to serve the first low-energy sensor node and deploy a new one, it will flood a "*Search*" packet to find other low-energy sensor nodes. The low-energy sensor nodes will reply with a packet to the mobile robot after receiving the "*Search*" packet. When the mobile robot receives the first reply packet from any of the low-energy sensor nodes, it will move to the first replying node. After arriving at the first replying node, the mobile robot will find the remaining sensor nodes. This procedure will be repeated until the mobile robot serves all n requests. In Fig. 10, the sink node has received three requests from nodes X, Y, and Z. The sink node commands one mobile robot to serve the three sensor nodes. The mobile robot first moves to node X, since this node has the least hops to the sink node. After the mobile robot moves to node X and deploys a new sensor node, it floods a "*Search*" packet to find the nodes Y and Z. (We have assumed that the mobile robot received the first reply from node Y.) The mobile robot moves to node Y along the reverse of the replying path. In the same way, the mobile robot creates a navigation path to node Z by flooding a "*Search*" packet at node Y. Note that, a sequence number is inserted into each "*Search*" packet to distinguish them, one from the other. After receiving the reply from node Z, the mobile robot moves to node Z, before returning to the sink node.

The algorithm can be more efficient, if the sink node has the knowledge of hop counts between any two-sensor nodes in the network. Therefore, it is unnecessary to flood a "*Search*" packet throughout the whole network, if the mobile robot has the

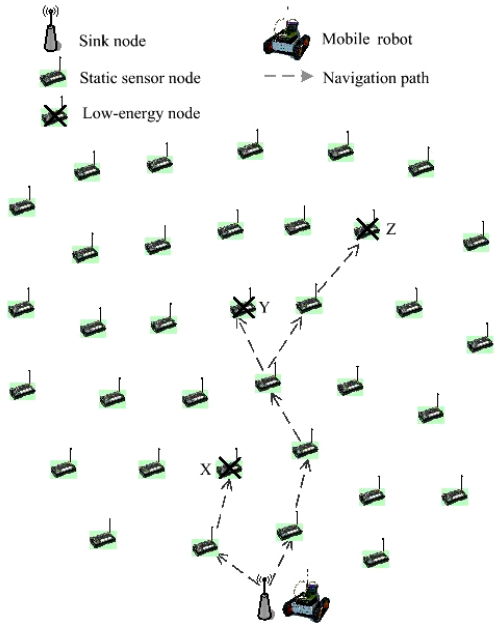


Fig. 10. Navigation paths of one-to-three service.

information of hop counts between any two-sensor nodes in the network. A "TTL" (time to live) variable can be used to limit the number of intermediate nodes allowed to forward "*Search*" packets. As the "*Search*" packet is forwarded, the "TTL" value decreases by one on each hop, and the "*Search*" packet is discarded, if the value equals zero. This can reduce the forwarding of many unnecessary packets and save the energy consumption of each sensor node. Moreover, we can design an algorithm to find the minimum cost to visit all the low-energy sensor nodes if sink node has the knowledge of the whole network topology. However, this problem is equivalent to the traveling salesman, an NP-complete problem [18], which cannot be solved in a reasonable time, as the number of target nodes n is large. However, several heuristic algorithms can be used to efficiently solve the problem, and obtain sub-optimal solutions [18].

4. EXPERIMENT RESULTS

In order to evaluate the performance of our navigation protocols, we performed three different experiments. In the first experiment, we used a single mobile robot to find the location of a three-hops-away target node. We evaluated the distance accuracy and navigation time of the mobile robot moving towards the destination node. In the second experiment, we used two mobile robots, to simultaneously serve two different sensor nodes, respectively. In the last experiment, we employed a single mobile robot to serve three low-energy sensor nodes in turn.

The sink node was composed of a laptop, a Mote Interface Board, and a MICA2. The MICA2 and MICA2DOTs were used as static sensor nodes in the sensor network.

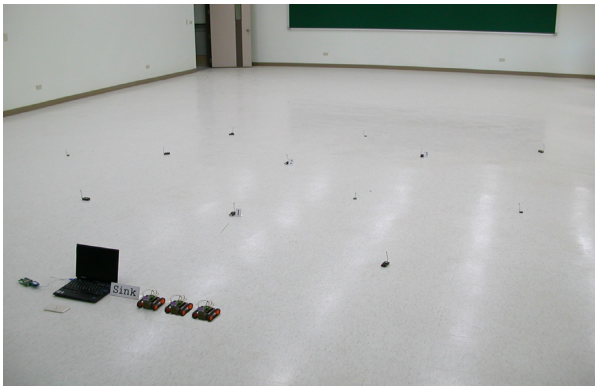


Fig. 11. Our experimental environment.

Table 1. The ranges of received RSSI in the maximum-power transmission and low-power transmission levels.

Power level / Signal strength	(Low-power level) – 13 dBm	(Maximum-power level) 5 dBm
7	0 ~ 15 cm	0 ~ 122 cm
6	10 ~ 31 cm	117 ~ 185 cm
5	25 ~ 73 cm	179 ~ 368 cm
4	66 ~ 112 cm	362 ~ 601 cm
3	106 ~ 189 cm	594 ~ 986 cm

Our experimental environment was situated in a large, free-space classroom, as shown in Fig. 11. The sensor network consisted of 12 static sensor nodes, one sink node, and three mobile robots. The static sensor nodes were manually irregularly deployed. The task of the mobile robots in our experiment was to navigate from the sink node to the low-energy sensor nodes.

The radio on the MICA2 could be adjusted for a range of output power levels. The MICA2 provided 26 different kinds of power levels from – 20 dBm to 5 dBm. For our navigation protocol, we selected two power levels; the maximum-power level was 5 dBm (3.16 milliWatt) and the low-power level was – 13 dBm (0.05 milliWatt). The radio on the MICA2 also provided a measurement of the received signal strength, referred to as RSSI which was 10-bits of data. The highest bit of RSSI was used to indicate whether the MICA2 had received a signal or not. The other 9 bits represented the value of the RSSI. However, the method of using the 9 bits to indicate signal strength is too sensitive. The value of the 9 bits would continuously change, even with the mobile robot being stationary. Therefore, we chose the highest three bits of data as the received signal strength and defined eight degrees of RSSI from 0 to 7. Table 1 shows the detection range of various signal strengths, for the two selected power levels. Since the last three degrees of RSSI (0 – 2) were unstable, we only adopted the signal strength from degrees 3 to 7.

In order to allow the mobile robot to easily receive the beacon packets from the beacon node, we used the maximum-power level to guide the mobile robots in the beginning of navigation. When the degree of RSSI received by the mobile robot reached 7, this indicated that the distance between the mobile robot and beacon node was about 1 meter.

In the meantime, the mobile robot notified the beacon node to send the beacon using low transmission power, to save energy and so that the mobile robot could navigate closer to the beacon node. Finally, the mobile robot would approach the beacon node, if the received RSSI was 7, under low transmission power.

In the first experiment, we used a single mobile robot to navigate from the sink node to a target node three hops away from the sensor node. The static sensor nodes were randomly deployed and any two adjacent nodes were fixed at 1 or 2 meters. Our experiment observed the distance accuracy and navigation time of the mobile robot moving to its destination node. The distance accuracy was the distance between the target sensor node and the mobile robot after the mobile robot had arrived at the target sensor node. The navigation time was the average time taken for navigating from one sensor node to another. We performed 10 experiments for two different distances – 1 meter and 2 meters – between two adjacent nodes. In each experiment, we randomly chose a sensor node three-hops-away. In these experiments, there was no obvious difference in distance accuracy for the different distances between two adjacent nodes. The average distance accuracy for two adjacent nodes at a distance of 1 meter and 2 meters was 7.2 centimeters and 7.5 centimeters, respectively. This was because the termination condition of the mobile robot was dependent only on the received signal strength. The variances of distance accuracy for 1 meter and 2 meters were 1.35 and 1.49, respectively. However, the navigation time was proportional to the distance between any two adjacent nodes. The average navigation time was 28 seconds and 70 seconds, in the case of 1 meter and 2 meters, respectively. The variance of navigation time for 1 meter and 2 meters was 5.33 and 17.11, respectively.

In the second experiment, we used two mobile robots to navigate to two target sensor nodes in a one-to-one corresponding manner. Twelve static sensor nodes were arbitrarily disposed on the floor and the distance between any two adjacent nodes was 1 meter. We randomly chose two different static sensor nodes, which were three-hops-away from the sink node, as the targets, in each of the 10 experiments. This was used to show the possible effects of multiple communications occurring in the same sensor network. Each mobile robot navigated towards its own target node independently. Our experiment showed that some beacon packets may be lost, when two adjacent nodes send packets at the same time. The beacon packets can collide with each other as two adjacent nodes transmit their radio signals simultaneously. Therefore, the mobile robots may spend more time navigating to the target sensor nodes, than when serving a single target node. The average navigation time between two adjacent nodes in this case was 36 seconds and its variance was 8.22. After performing 10 experiments, the average distance accuracy was 7.4 centimeters and its variance was 1.25.

In the last experiment, we used a single mobile robot to serve three low-energy sensor nodes. We deployed 12 static sensor nodes irregularly on the floor, with the distance between any two adjacent nodes being about 1 meter. In each experiment, we randomly chose three sensor nodes, which were three-hops-away from the sink node, as the target nodes. At the beginning of each experiment, the mobile robot first moves to the node which has the least hops to the sink node. Then the mobile robot flooded a "*Search*" packet to find the navigation paths of the other two sensor nodes. Since flooding is not a reliable transmission protocol, the mobile robot would flood a "*Search*" packet again, if it did not receive any reply within 5 seconds. When the mobile robot received the first

reply packet from one of the two low-energy sensor nodes, it moved to the first replying node. This procedure was terminated after three target sensor nodes had been served by the mobile robot. After performing 10 experiments, the average distance accuracy was about 7.3 centimeters and the average navigation time between two adjacent nodes was about 29 seconds. The variance of distance accuracy was 1.23 and the variance of navigation time was 6.44. The experiment also showed that the mobile robot moved, on average, only 6 hops to serve the three target nodes. In the many-to-many service, it took totally 6 hops to serve three target sensor nodes. This showed that the total energy consumption of the one-to-many service was less than that of the many-to-many service.

As shown by the experimental results, our proposed navigation protocol allows a mobile robot to navigate a multi-hop destination successfully, without having location information. Mobile robots can also travel within the entire sensor network, with no extra equipment. The videos of our above experiments can be found at the website: http://axp1.csie.ncu.edu.tw/paper_related/paper_related.htm.

5. CONCLUSIONS

In this paper, we designed a smart mobile robot to implement our nodes replacement application to allow mobile robots to precisely navigate towards the low-energy sensor nodes. We used the received signal strength to determine the direction taken by the mobile robot. The mobile robot was able to move to the destination node, using a hop-by-hop approach. Finally, we proposed a greedy scheme to serve multiple low-energy sensor nodes, using a single mobile robot.

We performed three experiments in all: in one, a single mobile robot served one target sensor node; in another, two mobile robots served two target sensor nodes (many-to-many service); and lastly, one mobile robot served three target sensor nodes (one-to-many service). The experiment results showed that the mobile robots could accurately navigate to the target sensor nodes and the average distance accuracy, in the three experiments, was around 7.4 centimeters. Due to signal interference, the many-to-many service required a longer navigation time to reach each beacon node than the single target service. In addition, total energy consumption of the one-to-many service was less than that of the many-to-many service.

REFERENCES

1. C. Y. Chong and S. P. Kumar, "Sensor networks: evolution, opportunities, and challenges," in *Proceedings of the IEEE*, Vol. 91, 2003, pp. 1247-1256.
2. A. Mainwaring, J. Polastre, R. Szewczyk, D. Culler, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, 2002, pp. 88-97.
3. H. Yang and B. Sikdar, "A protocol for tracking mobile targets using sensor networks," in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications*, 2003, pp. 71-81.
4. A. Galstyan, B. Krishnamachari, K. Lerman, and S. Pattem, "Distributed online localization in sensor networks using a moving target," in *Proceedings of the 3rd In-*

ternational Symposium on Information Processing in Sensor Networks, 2004, pp. 61-70.
5. K. F. Ssu, C. H. Ou, and H. C. Jiau, "Localization with mobile anchor points in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, Vol. 54, 2005, pp. 1187-1197.
6. M. A. Batalin and G. S. Sukhatme, "Efficient exploration without localization," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 2, 2003, pp. 2714-2719.
7. P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme, "Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 4, 2004, pp. 3602-3608.
8. G. T. Sibley, M. H. Rahimi, and G. S. Sukhatme, "Robomote: a tiny mobile robot platform for large-scale ad-hoc sensor networks," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 2, 2002, pp. 1143-1148.
9. M. B. McMickell, B. Goodwine, and L. A. Montestruque, "MICAbot: a robotic platform for large-scale distributed robotics," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Vol. 2, 2003, pp. 1600-1605.
10. S. Bergbreiter and K. S. J. Pister, "CotsBots: an off-the-shelf platform for distributed robotics," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vol. 2, 2003, pp. 1632-1637.
11. J. P. Sheu, P. W. Cheng, and K. Y. Hsieh, "Design and implementation of a smart mobile robot," in *Proceedings of the IEEE International Conference on Wireless and Mobile Computing*, *Networking and Communications*, Vol. 3, 2005, pp. 422-429.
12. Crossbow Technology Inc., http://www.xbow.com.
13. G. Wang, G. Cao, and T. LaPorta, "A bidding protocol for deploying mobile sensors," in *Proceedings of the 11th IEEE International Conference on Network Protocols*, 2003, pp. 315-324.
14. F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte Carlo localization for mobile robots," in *Proceedings of IEEE International Conference on Robotics and Automation*, Vol. 2, 1999, pp. 1322-1328.
15. W. S. Lin, M. K. Chuang, and G. Tien, "Autonomous mobile robot navigation using stereovision," in *Proceedings of the IEEE International Conference on Mechatronics*, 2005, pp. 410-415.
16. TinyOS, http://www.tinyos.net.
17. D. J. Goodman, *Wireless Personal Communications Systems*, Addison Wesley, USA, 1998.
18. R. Neapolitan and K. Naimipour, *Foundations of Algorithms*, Jones and Bartlett, London, UK, 2003.

Jang-Ping Sheu (許健平) received the B.S. degree in Computer Science from Tamkang University, Taiwan, R.O.C., in 1981, and the M.S. and Ph.D. degrees in Computer Science from National Tsing Hua University, Taiwan, R.O.C., in 1983 and 1987, respectively.

He is currently a Chair Professor of the Department of Computer Science, National Tsing Hua University. He was a Chair of Department of Computer Science and Information Engineering, National Central University from 1997 to 1999. He was a Director of Computer Center, National Central University from 2003 to 2006. His current research interests include wireless communications and mobile computing. He was an associate editor of Journal of the Chinese Institute of Electrical Engineering, Journal of Information Science and Engineering, Journal of the Chinese Institute of Engineers, and Journal of Internet Technology. He is an associate editor of the IEEE Transactions on Parallel and Distributed Systems, International Journal of Ad Hoc and Ubiquitous Computing, and International Journal of Sensor Networks.

He received the Distinguished Research Awards of the National Science Council of the Taiwan, R.O.C. in 1993-1994, 1995 to 1996, and 1997 to 1998. He received the Distinguished Engineering Professor Award of the Chinese Institute of Engineers in 2003. He received the certificate of Distinguished Professorship, National Central University in 2005. He received the K. T. Li Research Breakthrough Award of the Institute of Information and Computing Machinery in 2007. Dr. Sheu is a senior member of the IEEE, a member of the ACM, and Phi Tau Phi Society.

Kun-Ying Hsieh (謝坤穎) was born in Taoyuan, Taiwan on January 16, 1978. He got his B.S. degree in July 2001 in Computer Science and Engineering, in Yuan Ze University, Chungli, Taiwan. Currently he is a Ph.D. student in the Department of Computer Science and Information Engineering as National Central University, Chungli, Taiwan. His research interests include mobile computing, wireless ad hoc network, and wireless sensor network.

Po-Wen Cheng (鄭博文) was born in Hsinchu, Taiwan on March 6, 1980. He received his Master degree in July 2004 in the Department of Computer Science and Information Engineering as National Central University, Chungli, Taiwan and his B.S. degree in July 2001 in Computer Science and Engineering from Yuan Ze University, Chungli, Taiwan. Currently he is an Associate Engineer in SoC Software and Application Technology Department, Industrial Technology Research Institute.