An Integrated Mobile Surveillance and Wireless Sensor (iMouse) System and Its Detection Delay Analysis

Yu-Chee Tseng
Department of Computer Science
National Chiao Tung University
Hsin-Chu, 30010, Taiwan
yctseng@csie.nctu.edu.tw

You-Chiun Wang
Department of Computer Science
National Chiao Tung University
Hsin-Chu, 30010, Taiwan
wangyc@csie.nctu.edu.tw

Kai-Yang Cheng
Department of Computer Science
National Chiao Tung University
Hsin-Chu, 30010, Taiwan
kycheng@csie.nctu.edu.tw

ABSTRACT

Wireless sensor networks (WSN) provide an inexpensive and convenient way to monitor physical environments. Integrating the context-aware capability of WSN into surveillance systems is an attractive direction. We thus propose an integrated mobile surveillance and wireless sensor (*iMouse*) system, which consists of a large number of inexpensive static sensors and a small number of more expensive mobile sensors. The former is to monitor the environment, while the latter can move to certain locations and takes more advanced actions. The iMouse system is a mobile, context-aware surveillance system. We demonstrate our current prototyping for home security applications. Besides, we analyze its *event detection delay* under an *any-sensor-detection* model.

Categories and Subject Descriptors: C.2 [Computer-Communication Networks]: Data communications; C.2.1 [Network Architecture and Design]: Wireless communication

General Terms: Design

Keywords: pervasive computing, robotics, surveillance system, wireless communication, wireless sensor network

1. INTRODUCTION

Recent advances in wireless communications and MEMS technologies have made WSN possible. A WSN consists of many tiny, low-power devices equipped with sensors, transceivers, and actuators [1]. It provides an inexpensive and convenient way to monitor physical environments. With its context awareness, WSN may enrich human life in many ways. Applications of WSN include surveillance, biological detection, and habitat, agriculture, and health monitoring [1, 2, 9, 12].

Integrating the context-aware capability of WSN into surveillance systems is an attractive direction that deserves investigation. Surveillance systems typically collect a large volume of audio/video information, which requires intensive computation/manpower to analyze. Including the intelligence of WSN can help reduce such overheads and even provide more advanced, context-rich services. For example, in security applications, when something abnormal is detected, in-depth analyses may be conducted to find out the possible sources.

In this work, we propose an integrated mobile surveillance and wireless sensor (*iMouse*) system. The iMouse system consists of a large number of inexpensive static wireless sensors and a small number of more expensive mobile sensors. The former is to monitor the environment, while the latter can move to certain locations (such as potential emergency sites) and takes more advanced actions (such as taking pictures of the emergency scenes and conducting in-depth analyses). The iMouse system is a mobile, context-aware surveillance system. We demonstrate our current prototyping for home security applications. In particular, each mobile sensor has a mini-computer, which is connected to a data collector, a WebCam, and an 802.11 WLAN card, and is mobilized by a Lego car. At normal times, a mobile sensor can collect sensory data and report to an external server. When special events are detected, it can move to the event locations, take snapshots of the scenes, and send pictures to the server through its 802.11 interface. We demonstrate applications of iMouse through a fire emergency example.

In iMouse, when an event occurs, the *event detection delay* is an important factor that affects the responsiveness of the system. This depends on the network deployment and the location where the event appears. We adopt a probabilistic approach to model this problem and analyze the delay under an *any-sensor-detection* model.

The rest of this work is organized as follows. Section 2 reviews some related work. Section 3 discusses detailed design and implementation of our iMouse system. Section 4 presents our analyses on event detection delay. Conclusions are drawn in Section 5.

2. RELATED WORK

Our objective is to study the feasibility of combining surveillance systems with WSN. Most visual surveillance systems deal with the real-time monitoring of persistent and transient objects. The primary goals of these systems are to provide an automatic interpretation of scenes and to understand/predict actions of the observed objects from the information acquired from cameras or CCTV (closed circuit television) [14]. The surveillance issue has also been discussed in the field of robotics [7, 10]. The system is assumed to have a robot with many static cameras installed on locations such

MSWiM'05, October 10–13, 2005, Montreal, Quebec, Canada.

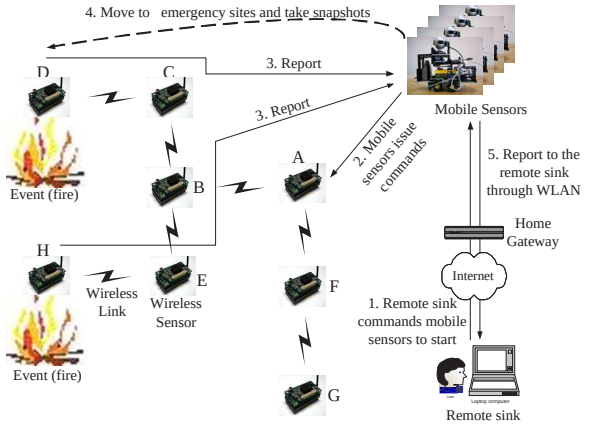

Figure 1: System architecture of the iMouse system.

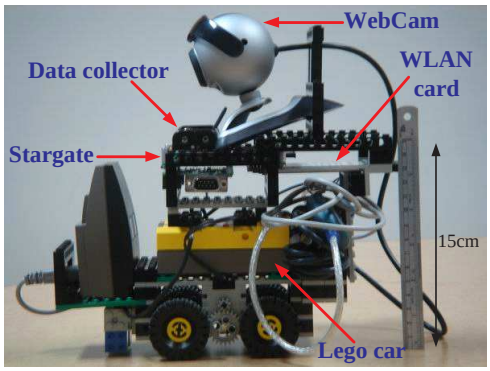

Figure 2: The mobile sensor.

as walls. These cameras are used to find obstacles or humans in the field, so that the robot can detour around these obstacles. On the side of WSN, the event/object tracking issue has been intensively studied [6, 8, 11, 15]. Most works assume that the intrusion objects can emit some signals (such as noise or light), or the objects themselves are phenomenal (such as diffused gas or chemical liquid [8]). However, results reported from a WSN are typically very brief and lack of more in-depth information. This motivates us to study the feasibility of integrating WSN with surveillance systems to support intelligent context-aware surveillance services.

3. DESIGN OF THE IMOUSE SYSTEM

3.1 System Architecture

Fig. 1 shows the system architecture of the iMouse system. It consists of a static WSN and few mobile sensors. At normal time, the WSN collects and reports environment information to the mobile sensors. When necessary, the mobile sensors are able to move to the event locations, conduct more advanced analyses of the event scenes, and report the analysis results to the remote sink.

Each sensor of the WSN consists of a data collector and a sensing board. The data collector is used to communicate with other sensors. The sensing board is used to collect environment data. In our current prototype, three types of data can be collected: voice, temperature, and light. Reporting of events is reactive, and an event is defined when the sensory input is higher than a threshold. Different inputs can be combined to define an event. For example, for fire emergency, a combination of light and temperature thresholds can be used.

Mobile sensors have five major functionalities: issuing commands to the WSN, gathering data from the WSN, moving to some target areas, taking snapshots, and reporting analyses to the remote sink. Each mobile sensor is empowered by a microprocessor called Stargate [5], which is connected to a data collector, a Lego car, a WebCam, and an IEEE 802.11 WLAN card, as shown in Fig. 2. The data collector can communicate with static sensors to issue commands or gather data. The Lego car [13], produced by MindStorms, supports mobility. The WebCam is to take photos of the emergency scenes. To eliminate the wiring problem, the 802.11 WLAN card is used to talk to the home gateway. The Stargate is the brain of a mobile sensor. For example, it decides the visiting sequence of potential emergency sites.

3.2 A Fire Emergency Scenario

Below, we give a fire emergency scenario to demonstrate how the iMouse system works (refer to Fig. 1). On receiving the remote sink's command, the WSN will form a spanning tree to collect environment data. Suppose that sensors D and H reply high temperatures and are thus suspected of fire emergency in their neighborhood. On receiving such notifications, the mobile sensors will coordinate and decide who will be delegated to which sensors via which shortest path. On visiting D and H, the mobile sensor(s) will take snapshots of these sites from different angles. After returning to the home gateway, the mobile sensor(s) will send these snapshots back to the remote sink for further actions.

3.3 Implementation Details

3.3.1 *Hardware Specifications*

We use the MICAz by CrossBow [4] as sensor nodes, which are 2.4 GHz, IEEE 802.15.4-compliant Mote modules offering a data rate of 250 kbps with a DSSS radio. The Stargate, also manufactured by Crossbow, consists of a 32-bits, 400-MHz Intel XScale RISC processor with 64 MB main memory. It drives a WebCam through the USB port, and an 802.11 WLAN card through the PCMCIA slot. The Stargate controls the Lego car via a Lego tower (9713 IR-TRANSMITTER). The Lego car has an infrared ray receiver in the front and two motors on the bottom. It also has a light sensor, which we use for navigation purpose. This is realized by different colors of the tapes that we stick on the ground. With this mechanism, the Lego car can localize itself in the sensing field.

An experimental 2×2 grid-like sensing field (Fig. 3) is demonstrated. On the ground, golden tapes represent intersections and black tapes represent roads. The origin $(0, 0)$ is at the lower left corner. Four sensors are placed at $(1, 1)$, $(0, 2)$, $(2, 0)$, and $(2, 2)$, respectively. The transmission range is manually set to two units to fit into the relatively small sensing field. A light reading below 800 is to simulate a potential fire emergency.

3.3.2 *Protocol Specifications*

Each static sensor runs the algorithm in Fig. 4(a). Initially, it waits for commands from mobile sensors. A *tree-construct* command will trigger the static sensor to check

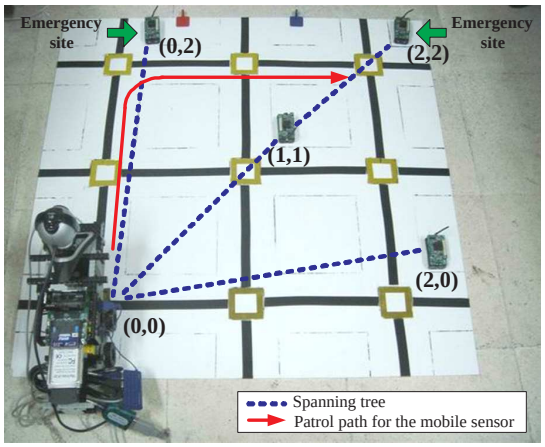

Figure 3: A 2×2 grid-like field in our experiment.

if its tree parent is null or has expired. If so, it sets the sender as its parent. Then it will re-broadcast the command. To distinguish new from old commands, each *tree-construct* command is assigned an unique sequence number. The goal is to form a spanning tree of the network. Then the sensor turns on its sensing devices. On detecting potential emergencies, sensors will transmit a *status-report* to mobile sensors along the spanning tree. Each mobile sensor runs the algorithm in Fig. 4(b). If a *tree-construct* command is received from the remote sink, it will broadcast it to the WSN. It then waits for *status-report* from static sensors and forwards the report to the sink. On receiving the sink's *patrol* command, the mobile sensors have to take further actions. The traveling-salesman algorithm APPROX-TSP-TOUR [3] is used to compute their patrolling paths. A spanning tree of all potential emergency sites is formed, and then a heuristic is used to partition the tree into a number of regions, each to be visited by a mobile sensor. For each region, the patrolling path is the preorder tree walk. Photos are saved in the Stargate's memory, and will be forwarded to the sink when the mobile sensor returns to the origin.

Two types of packets, *command* and *data*, are defined. The former is initiated by mobile sensors and the latter is initiated by static sensors. The formats are shown in Fig. 5. The *Original_ID*, *Source_ID*, and *Dest_ID* fields are the initiator, transmitter, and receiver of the packet. The *Seq_Num* field, together with the *Original_ID* field, guarantees the uniqueness of a message. The *Hop_Count* field of the command packet helps establish the spanning tree, and the *Data* field contains the sensing value and status of the originating sensor.

3.3.3 User Interfaces

We provide an interface to monitor the WSN's status and to control mobile sensors at the remote sink, as shown in Fig. 6. It includes six major components: *Config*, *Command*, *Status*, *Control*, *Monitoring* and *Log* areas. The Config area is to input configuration information, such as mobile sensors' IP addresses, ports, sensors' positions, etc. The Command area is to load the configuration file (such as sensors' positions) to sensors, establish connection from the sink to mobile sensors, issue tree-construct commands, change the parent of a static sensor, calculate the patrolling paths of mobile sensors, disconnect all mobile sensors (to reset all

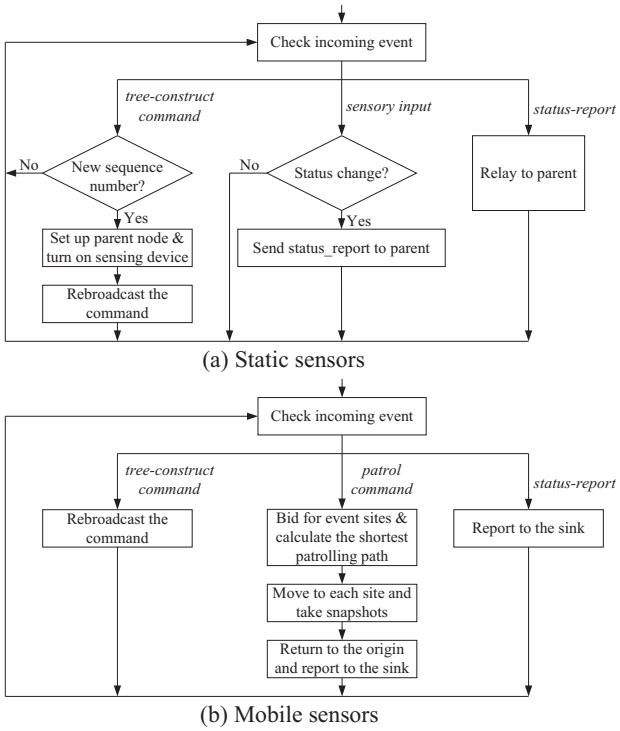

Figure 4: The algorithms run by the sensors.

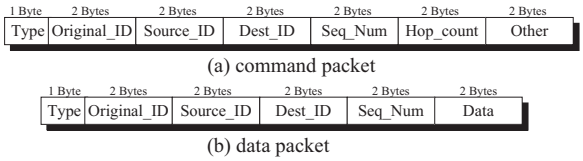

Figure 5: Packet formats.

environment parameters), add a new sensor in the WSN, set a sensor's position, remove a sensor from the network, and check the status of a static sensor. The Status area shows the status of a static sensor. The Control area is to control the action of a mobile sensor (motion and photoing). The Monitoring area shows the WSN's topology and the patrolling paths of mobile sensors. When a sensor detects an event, a fire icon is shown in the corresponding site. A camera icon is shown when a snapshot has been taken for the site. Finally, the Log area shows some status messages.

4. ANALYSIS OF DETECTION DELAY

In this section, we propose a model to analyze the event detection delay of the iMouse system. We are given a sensing field A, on which there are n homogeneous sensors. Each sensor has a sensing distance r. We assume that these n sensors form a connected network. Besides, we assume that the time axis is divided into fixed-length *slots* and the working schedule of each sensor is modeled by *cycles*, where each cycle consists of T slots. Each cycle is divided into an *active phase* and an *idle phase*. The former consists of the first D slots, and the latter the rest of the $T - D$ slots. Sensors only conduct detection jobs in their active phases. However, sensors do not synchronize their clocks, so their cycles are not

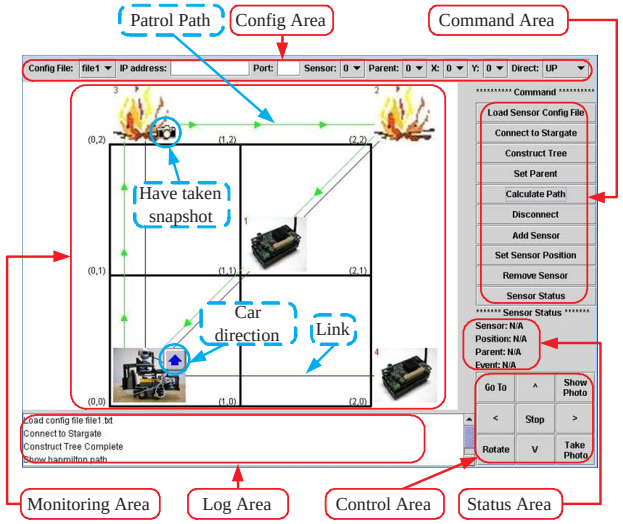

Figure 6: User interface at the remote sink.

necessarily aligned. Our goal is to evaluate the detection delay when an event appears. To take errors into account, we assume that in an active slot, a sensor has a probability of p to successfully detect the event if the event is within its sensing range. We consider an *any-sensor-detection* model in this work. Specifically, to capture the event, the network needs at least one sensor to successfully detect the event.

Suppose that at time slot 0, an event appears at location (x, y). Let $M(x, y)$ be the number of sensors whose sensing ranges cover (x, y). We classify these $M(x, y)$ sensors into T groups so that each group i consists of m_i sensors whose first active slot is at the ith slot, $i = 1..T$. Clearly, $\sum_{i=1}^{T} m_i = M(x, y)$. Taking all combinations of m_i's into consideration, the detection delay can be written as

$$Delay(x,y) = \sum_{m_1=0}^{M(x,y)} \sum_{m_2=0}^{M(x,y)-m_1} \cdots \sum_{m_{T-1}=0}^{M(x,y)-(m_1+\cdots+m_{T-2})} \left(\frac{M(x,y)!}{m_1! \cdots m_T!} \times (\frac{1}{T})^{M(x,y)} \right) \times \delta(m_1, ..., m_T),$$

where the first term is the probability to observe a particular combination $(m_1, ..., m_T)$, and the second term $\delta(m_1, ..., m_T)$ is the expected delay for this particular combination. As the event may appear in any location inside A, the average delay is $\frac{\Sigma_{\forall(x,y)} Delay(x,y)}{\Sigma_{\forall(x,y)} 1}$. To compute $\delta(m_1, ..., m_T)$, let x_i be the number of active sensors at the ith slot, $i = 1..T$. These x_i sensors can be classified into three types: (1) sensors that turn into active at the ith slot, (2) sensors that turn into active between the first and the $(i-1)$th slots, and (3) sensors that turn into active at or before the 0-th slot. This leads to $x_i = m_i + \sum_{j=1}^{i-1} m_{i-j} + \sum_{j=0}^{D-i-1} m_{T-j}$. We also define x_{aT+b} as the number of active sensors at the $(aT+b)$th slot for any $a \geq 1$. Since cycles repeat every T slots, we have $x_{aT+b} = x_b$.

The probability that there is at least one sensor successfully detecting the event in the first slot is $(1-(1-p)^{x_1})$. For $i \geq 2$, the probability that the event is not detected in the first $(i-1)$ slots but is successfully detected in the ith slot is $(1-(1-p)^{x_i})(1-p)^{x_1+\ldots+x_{i-1}}$. Hence, the expected detection delay under the any-sensor-detection model is $\delta(m_1, ..., m_T) = \sum_{a=0}^{\infty} \sum_{b=1}^{T} (aT+b)(1-(1-p)^{x_b}) \cdot (1-p)^{a \times (x_1+\cdots+x_T)+x_1+\cdots+x_{b-1}}$.

5. CONCLUSIONS

The proposed *iMouse* system combines two areas, WSN and surveillance technology, to support intelligent mobile surveillance services. On one hand, the mobile sensors can help improve the weakness of traditional WSN that they only provide vague environmental information by including some mobile cameras to conduct in-depth analysis of the sensing field. On the other hand, the WSN provides context awareness and intelligence to the surveillance system. Therefore, the weakness of traditional dumb surveillance system is greatly improved because the real critical images/video sections can be retrieved and sent to users.

6. ACKNOWLEDGMENTS

Y. C. Tseng's research is co-sponsored by the NSC Program for Promoting Academic Excellence of Universities under grant number 93-2752-E-007-001-PAE, by Computer and Communications Research Labs., ITRI, Taiwan, by the Communications Software Technology Project of III, Taiwan, and by Intel Inc.

7. REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *IEEE Communications Magazine*, 40(8):102–114, 2002.

[2] J. Burrell, T. Brooke, and R. Beckwith. Vineyard computing: sensor networks in agricultural production. *IEEE Pervasive Computing*, 3(1):38–45, 2004.

[3] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, 2001.

[4] Crossbow. MOTE-KIT2400 - MICAz Developer's Kit, http://www.xbow.com, 2004.

[5] Crossbow. SPB400 - Stargate Gateway, http://www.xbow.com, 2004.

[6] C. Gui and P. Mohapatra. Power conservation and quality of surveillance in target tracking sensor networks. In *ACM/IEEE Int'l Conf. on Mobile Computing and Networking*, pages 129–143, 2004.

[7] A. Hoover and B. D. Olsen. Sensor network perception for mobile robotics. In *IEEE Int'l Conf. on Robotics and Automation*, pages 83–88, 2000.

[8] X. Ji, H. Zha, J. J. Metzner, and G. Kesidis. Dynamic cluster structure for object detection and tracking in wireless ad-hoc sensor networks. In *IEEE Int'l Conf. on Communications*, pages 3807–3811, 2004.

[9] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Next century challenges: mobile networking for "Smart Dust". In *ACM/IEEE Int'l Conf. on Mobile Computing and Networking*, pages 271–278, 1999.

[10] J. H. Lee and H. Hashimoto. Controlling mobile robots in distributed intelligent sensor network. *IEEE Trans. on Industrial Electronics*, 50(5):890–902, 2003.

[11] J. Liu, M. Chu, J. Liu, J. Reich, and F. Zhao. Distributed state representation for tracking problems in sensor networks. In *Information Processing in Sensor Networks*, pages 234–242, 2004.

[12] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson. Wireless sensor networks for habitat monitoring. In *ACM Int'l Workshop on Wireless Sensor Networks and Applications*, pages 88–97, 2002.

[13] MINDSTORM. Robotics invention system, http://mindstorms.lego.com/eng/default.asp, 2004.

[14] M. Valera and S. A. Velastin. Intelligent distributed surveillance systems: a review. *IEE Proceedings - Vision, Image and Signal Processing*, 152(2):192–204, 2005.

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