iMouse: An Integrated Mobile Surveillance and Wireless Sensor System

Yu-Chee Tseng, You-Chiun Wang, Kai-Yang Cheng, and Yao-Yu Hsieh

Abstract—Wireless sensor networks (WSNs) offer a convenient manner to monitor physical environments. Incorporating the environment-sensing capability of WSNs into the currently popular video-based surveillance systems is an attractive direction. We propose an (iMouse) *integrated mobile surveillance and wireless sensor system*, which consists of many static wireless sensors and few sophisticated mobile sensors, where the former can detect unusual events in the environment, while the latter is capable of moving to emergency sites and conducting more in-depth analysis of the events. The iMouse system is thus a mobile, event-driven surveillance system. We demonstrate a prototyping system for home/office security applications. Through the system, we propose a *mobile sensor dispatch problem*, which addresses how to schedule mobile sensors to visit emergency sites with energy efficiency in mind, given arbitrary numbers of mobile sensors and emergency sites. Experimental results are presented to evaluate our schemes.

Index Terms—mobile computing, pervasive computing, sensor network, surveillance system, wireless communication.

1 Introduction

Recently, the remarkable advances of micro-sensing MEMS and wireless communication technologies have promoted the development of wireless sensor networks (WSNs). A WSN consists of many sensor nodes densely deployed in a field, each being able to collect environmental information and altogether being able to support multi-hop ad hoc routing. WSNs provide an inexpensive and convenient way to monitor physical environments. With their environment-sensing capability, WSNs can enrich human life in various applications, such as health, building monitoring, and home security.

Integrating the sensing capability of WSNs into surveillance systems is an attractive direction. Traditional surveillance systems typically collect a large volume of videos from wall-board cameras, which require huge computation or even manpower to analyze. Including the intelligence of WSNs can help reduce such overheads while provide more advanced, context-rich services. For example, in security applications, when something abnormal is detected, in-depth analyses can be conducted to find out the possible sources. In intrusion-detection applications, when trespassing is detected, a metal detector can help check if the intruder carries a weapon.

Motivated by the above observations, in this paper we propose an iMouse (*integrated mobile surveillance and wireless sensor*) system. It consists of a large number of static wireless sensors and a small number of more powerful mobile sensors. The former is used to monitor the environment, while the latter can move to the potential event emergency sites and conduct more in-depth analyses. In particular, each mobile sensor is equipped with a processing platform, which is connected to a Mote, a WebCam, and an 802.11 WLAN card, and is mobilized by a LEGO car. At normal times, static sensors continuously monitor their surrounding. When something unusual is reported, the external server will dispatch mobile sensors to visit these event locations. After moving to these locations, a mobile sensor can take snapshots of the event scenes and send images to the server through its WLAN interface.

Our iMouse system offers several attractive features. First, it provides an *on-line* and *real-time* monitoring. For example, when there are events being captured, the static sensors can immediately inform users where the events occur, while the mobile sensors can later provide detailed images of these events. Second, the iMouse system is *event-driven*, in the sense that only when an event occurs should a mobile sensor be dispatched to capture the images of that event. Thus, our system can avoid recording unnecessary images when nothing happens. Third, these more expensive mobile sensors are to be dispatched to the event locations. They do not need to cover the whole sensing field, so only a small number of mobile sensors are sufficient. Finally, iMouse is characterized by modularity and scalability; it can be improved by adding more sophisticated devices on mobile sensors to strengthen their sensing capability, without substituting existing static sensors.

Since mobile sensors are operated by batteries, how to extend their lifetimes is an important issue. We thus propose a new *mobile sensor dispatch problem*, which addresses how to schedule mobile sensors to visit emergency sites so that their energy can be conserved as much as possible. We show that if the number of emergency sites is no larger than that of mobile sensors, the problem can be transformed to a *maximum matching problem* in a bipartite graph; otherwise, we show how to group emergency sites into clusters such that each cluster can be visited by one mobile sensor efficiently.

The rest of this paper is organized as follows: Section 2 reviews some related work. Section 3 discusses the design of our iMouse system and the mobile sensor dispatch problem. Section 4 reports our prototyping experiences. Section 5 presents some experimental results. Section 6 concludes this paper.

2 Related Work

The goal of this paper is to study the feasibility of integrating WSNs into surveillance systems. Traditional visual surveil-

The authors are with the Department of Computer Science, National Chiao-Tung University, Hsin-Chu, 30010, Taiwan.
E-mail: {yctseng, wangyc, kycheng, shiehyy}@cs.nctu.edu.tw

lance systems continuously videotape scenes by cameras to capture transient or suspicious objects. Such systems typically need to automatically interpret the scenes and understand/predict actions of observed objects from the acquired videos [1]. For example, a video-based surveillance network is proposed in [2], where the information captured by each video camera is transmitted by an 802.11 WLAN card. The surveillance issue has also been discussed in the field of robotics [3], [4]. Cameras are installed on walls or robots to identify obstacles or humans in the environment. The goal is to guide robots to detour around these obstacles. Such systems normally have to extract few meaningful information from massive visual data, which requires lots of computation or manpower.

For WSNs, object tracking has been studied for static networks [5]–[7]. These works assume that objects can emit some signals so that sensors can track them. However, results reported from a WSN are typically very brief and lack of in-depth information. A video-based surveillance system for capturing intrusions by merging WSNs and video processing techniques is proposed in [8]; data from WSNs is complemented with videos to capture the possible scenes with intruders. However, cameras in this system lack mobility, so only some particular locations can be monitored. Mobilizers have also been addressed in [9]–[11] to move sensors to enhance coverage of the sensing field and in [12] to strengthen the network connectivity . The works [13], [14] address the *pursuer-evader game* by WSNs, where a pursuer needs to intercept a evader in the field by the assistance of WSNs. To our knowledge, how to integrate WSNs with surveillance systems has not been well addressed, which motivates us to propose the iMouse system.

3 Design of the iMouse System

3.1 System Architecture

The architecture of iMouse is illustrated in Fig. 1, which consists of static sensors, mobile sensors, and an external server. The static sensors form a WSN to monitor the environment and to notify the server if there are unusual events. We assume that locations of these sensors are known, which can be obtained by manual setting, GPS, or any localization schemes [15], [16]. When something unusual is reported, the mobile sensors will move to the emergency sites and conduct more advanced analyses of the events.

Each static sensor is composed of a sensing board and a Mote. The former is for sensing, while the latter for communication. In our current prototype, three types of data can be collected by a sensing board, including light, sound, and temperature. An event is defined when the sensory input is higher/lower than a predefined threshold. Different inputs can be combined to define a new event. For example, a combination of light and temperature readings can be interpreted as a potential fire emergency. To detect an explosion, a combination of temperature and sound readings can be used. For home security, an unusual sound or light reading may be used. To conserve the energy of static sensors, reporting of events is *reactive*.

Each mobile sensor has the following functionalities: moving to event locations, exchanging messages with other sensors, taking snapshots of event scenes, and transmitting images to the server. In particular, each mobile sensor is equipped with a processing board *Stargate* [17], which is connected to a LEGO car, a Mote, a WebCam, and an IEEE 802.11 WLAN card, as shown in Fig. 2. The LEGO car supports mobility. The Mote can communicate with static sensors. The WebCam is used to take snapshots. The WLAN card is to support high-speed, long-distance communications, such as transmitting images. The Stargate controls the movement of the LEGO car and the WebCam.

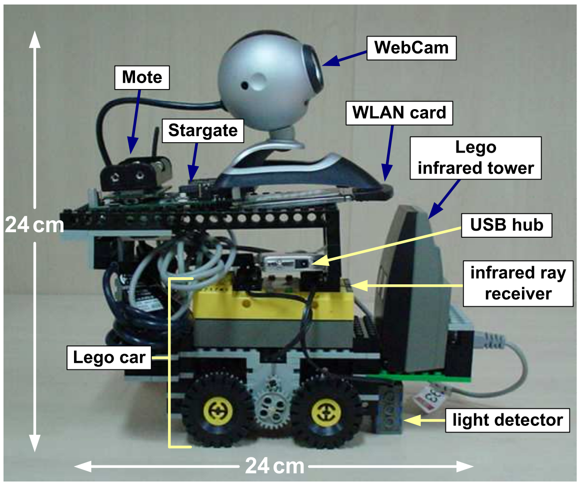

Fig. 2: The mobile sensor.

The external server provides an interface to let users obtain the system statuses and issue commands. It also maintains the network and interprets the meanings of events from sensors. On detecting potential emergency, the server will dispatch mobile sensors to visit emergency sites to obtain high-resolution images of these scenes. The dispatch algorithm discussed in Section 3.3 is also run in the server.

3.2 System Operations and Control Flows

First, we give a fire emergency scenario to illustrate how iMouse works (refer to Fig. 1). On receiving the server's command, the static sensors will form a tree-like network to collect sensing data. Suppose that static sensors A and C report unusual high temperatures and thus are suspected of fire emergency in their neighborhoods. The server will then notify the users and dispatch mobile sensors to visit these sites. On visiting A and C, the mobile sensors will take snapshots and perform in-depth analyses. For example, the reported images may indicate the source of fire, any inflammable material in the vicinity may be identified, and people still left in the building may be located.

Next, we discuss the control flows. Each static sensor runs the algorithm in Fig. 3. The server periodically floods a *tree-maintenance* message to maintain the WSN. It also records each static sensor's location and state, which is set to *normal* initially. *Tree-maintenance* messages will help the static sensors to track their parent nodes. To distinguish new from old messages, each *tree-maintenance* message is associated with a unique sequence number. The goal is to form a spanning tree in the WSN.

When receiving a sensory input above the threshold, an event is detected and should be reported to the server. However, to avoid sending duplicate messages, each sensor will keep a variable *event flag* to indicate whether it has already reported that event. When a sensor detects an event and the *event flag* is false, the sensor reports that event and set the flag to true. The server will collect multiple events and assign them

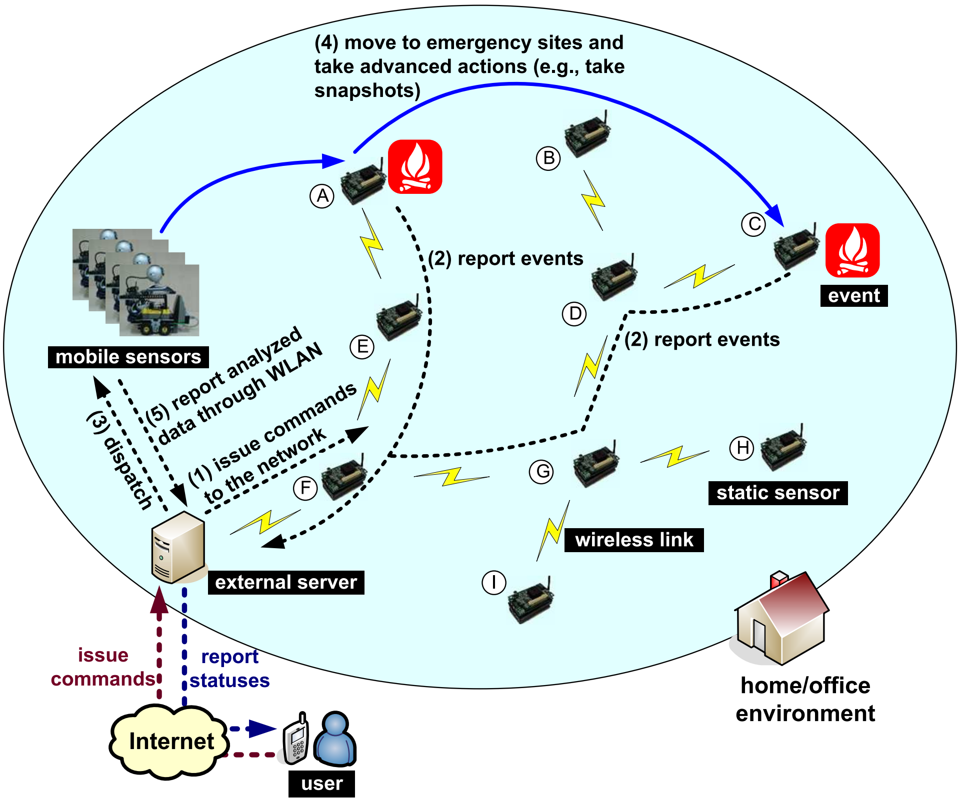

Fig. 1: System architecture of the iMouse system.

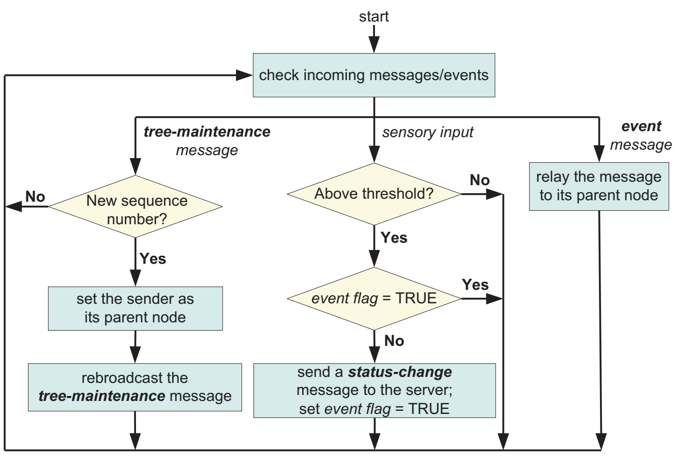

Fig. 3: The algorithm executed by static sensors.

to mobile sensors in batches. When a mobile sensor visits an event site, it will ask the local sensor to clear its *event flag*.

3.3 The Mobile Sensor Dispatch and Traversal Problems

Next, we will discuss how to dispatch mobile sensors to event sites and determine their traversal paths. Since mobile sensors are operated by batteries, our goal is to assign mobile sensors to emergency sites so that their energies can be conserved as much as possible. Specifically, we consider a set L of m emergency sites to be visited by a set S of n mobile sensors, where each site needs to be visited by one mobile sensor. We allow an arbitrary relationship between m and n. The goal is to maximize the total remaining energy of mobile sensors after sites are visited.

Our dispatch solution depends on the relationship of m and n. When $m \leq n$, the problem can be converted to one of finding a maximum matching in a weighted bipartite graph $G = (S \cup L, S \times L)$, where the vertex set is $S \cup L$ and the edge set is the product $S \times L = \{(s_i, l_j) | s_i \in S, l_j \in L\}$. The weight of (s_i, l_j) is set to $e_i - e_{mv} \times d(s_i, l_j)$, where e_i is the current energy of s_i, e_{mv} is the energy cost for a mobile sensor to move by one unit and $d(s_i, l_j)$ is the distance from s_i's current location to l_j. The solution is the maximum matching P of G, which can be found by traditional maximum-weight matching solutions [18]. Alternatively, we may set our objective to minimizing the total moving distances of mobile sensors. This can still be done by maximum-matching by setting the weight of (s_i, l_j) to $-e_{move} \times d(s_i, l_j)$.

When $m > n$, some mobile sensors have to visit multiple sites. Our possible solution is to divide emergency sites into n clusters (for example, by the classical K-means method [19]), and assign each group to one mobile sensor. In this case, the cost of each mobile sensor will include moving to the closest site in each group and then traversing the rest of the sites one by one. Given a set of locations to be visited, we can to use a heuristic to the traveling salesman problem (TSP) [18] to determine the order of traversal.

4 Implementation and Experimental Results

We use the MICAz Motes [20] as static sensors. A MICAz is a 2.4 GHz, IEEE 802.15.4-compliant module allowing low-power operations and offering a data rate of 250 kbps with a DSSS radio. For mobile sensors, we use the Stargate [17] as their processing platforms. A Stargate consists of a 32-bits, 400-MHz Intel PXA-255 XScale RISC processor with 64 MB main memory and 32 MB extended flash memory. It also has a daughter board with an RS-232 serial port, a PCMCIA slot, a USB port, and a 51-pin extension connector, which can be attached to a Mote. It drives a WebCam through a USB port, and an IEEE 802.11 WLAN card through its PCMCIA slot. The Stargate controls the LEGO car [21] via a USB port connected to a LEGO infrared tower, as shown in Fig. 2. The LEGO car has an infrared ray receiver in the front to receive commands from the tower and two motors on the bottom to drive wheels.

Navigating a mobile sensor or a robot is in general a very difficult problem, unless some auxiliary devices are provided. Reference [22] uses wall-board cameras to capture the locations of mobile sensors while the work [23] suggests to use signal strength to do so. In our current prototype, we use the light sensors on the LEGO car to navigate mobile sensors. This is realized by different colors of tapes that we stick on the ground. With this mechanism, the LEGO car can be navigated on a board easily. In our prototyping, an experimental 6×6 grid-like sensing field is implemented, as shown in Fig. 4. Black tapes represent roads and golden tapes represent intersections. Two mobile sensors and 17 static sensors are placed on the sensing field to construct the system. For static sensors, a light reading below 800 is to simulate an event, so we cover a static sensor with a box to model a potential emergency. Note that using a grid-like sensing field and a grid-like static sensor deployment is only for ease of implementation. In general, the topology of the static WSN can be irregular.

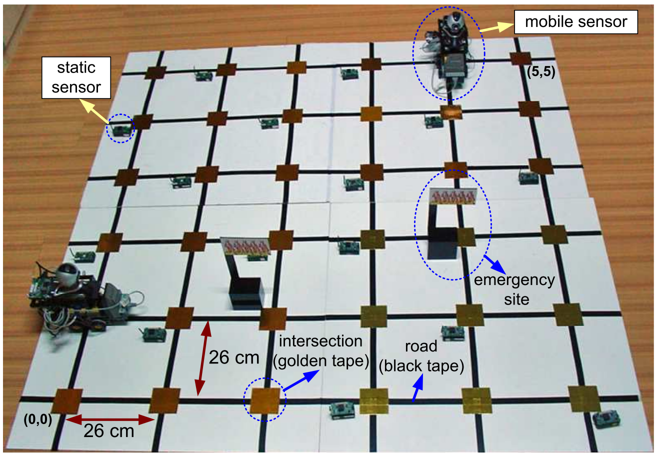

Fig. 4: A 6×6 grid-like sensing field in our experiment.

We evaluate the dispatch time of mobile sensors. The dispatch time is affected by three factors: the time that a mobile sensor crosses one grid-unit (about 26 cm), the time that a mobile sensor makes a 90-degree turn, and the time that a mobile sensor takes snapshots and report the results. These take 2.5, 2.2, and 4.0 seconds, respectively, in our current prototype.

Fig. 5 shows some experimental results with one mobile sensor placed at (0,0) and two mobile sensors placed at (0,0) and (5,5) initially. We generate some random events and evaluate the dispatch time (from the server being notified of these events to all event sites are visited) and the average time of each site (from an event being detected to a mobile sensor visiting the site). Clearly, using two mobile sensors can significantly reduce dispatch time and waiting time.

At the external server, we provide a user interface to monitor the status of the system and to control mobile sensors,

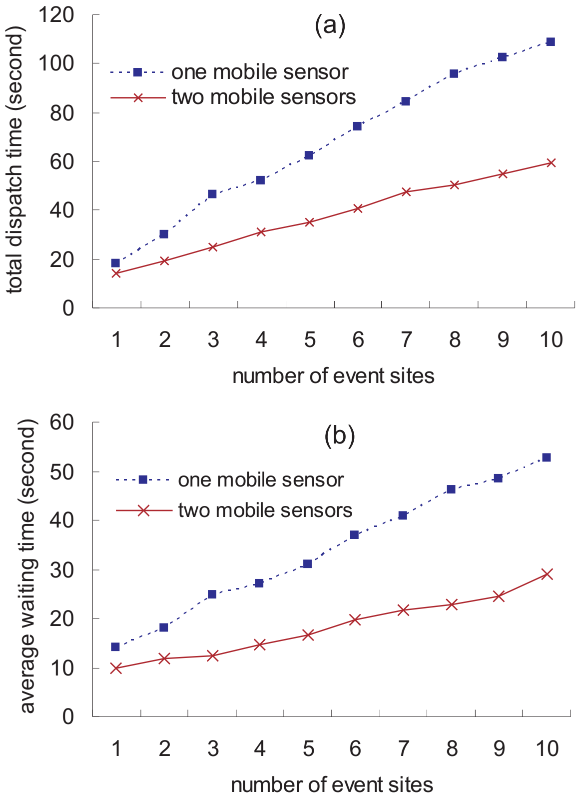

Fig. 5: Experimental performance of (a) dispatch time and (b) average waiting time.

as shown in Fig. 6. It includes six major components: *configure*, *system-command*, *sensor-status*, *action-control*, *monitoring*, and *log* areas. The configure area is to input configuration information of the system, such as mobile sensors' IP addresses, ports, and sensors' positions. The system-command area provides an interface to let users control the overall system, such as issuing a tree-maintenance message, adjusting the WSN's topology, connecting and disconnecting a specified mobile sensor, and so on. The sensor-status area shows the current status of a static sensor being queried. The action-control area is used to control the actions of a mobile sensor, including movement and taking snapshots. The monitor area shows the network topology of the WSN and the patrolling paths of mobile sensors. When a sensor detects an event, a fire icon will be shown in the corresponding site. Finally, the log area shows some status messages of the system.

5 Simulation Results

The above experiments only consider grid networks. To understand the sensor dispatch problem in general irregular WSNs, we have developed a simulator. We evaluate the average waiting time for an event location being visited and the total energy consumption of mobile sensors. We compare a greedy algorithm against the K-means algorithm. Given a set S of mobile sensor and a set L of emergency sites, the greedy algorithm contains a sequence of iterations. In each iteration, the mobile sensor which has the smallest distance to the nearest location is assigned to that location. This is repeated until all mobile sensors are assigned to locations. If there are unvisited locations, the first mobile sensor reaching its destination will pick its next location in the same greedy

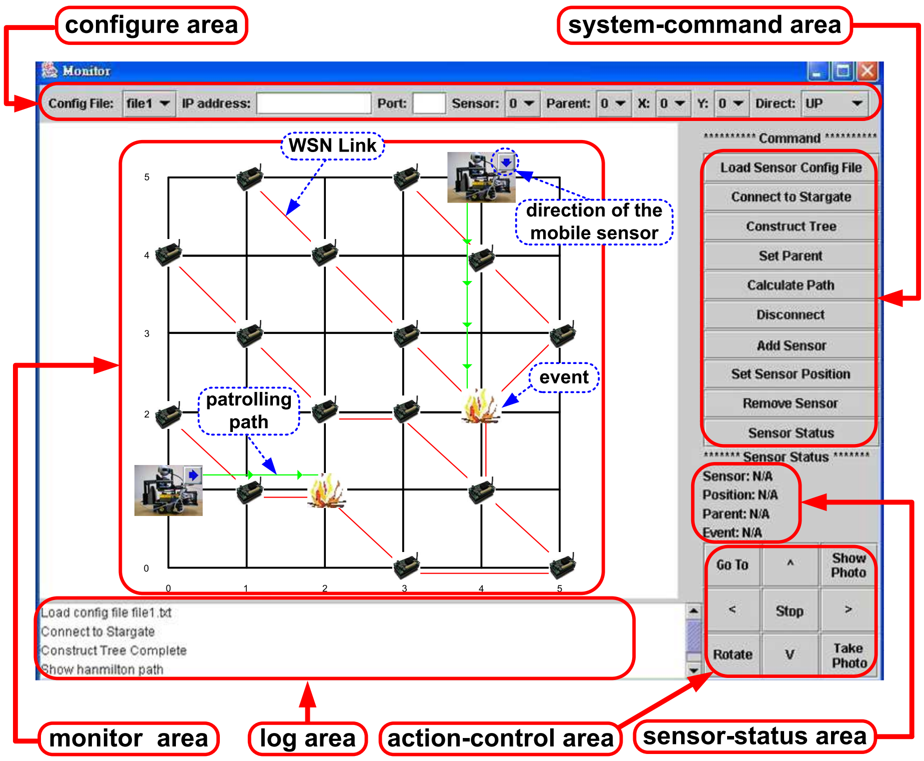

Fig. 6: User interface at the external server.

manner. This is repeated until all locations are visited. The K-means algorithm is as reviewed in Section 3.3.

In our simulation, the sensing field is a 15m $\times$ 15m square. Moving one meter is assumed to take one Joule and need ten seconds. From one location to another, a mobile sensor can move in a straight line. Each experiment has 100 rounds, where in each round a certain number of events are generated at random locations. After each round, mobile sensors will stay at their final destinations and wait for the next schedules. Each simulation result is marked with a 90% confidence interval.

Fig. 7 shows our comparison results under different numbers of event sites and mobile sensors. From Fig. 7(a) and (c), we see that the greedy algorithm performs better because of its time-critical nature. The unbalanced job assignments to mobile sensors by the K-means algorithm cause some events sites to wait longer time even if some mobile sensor are idle. On the other hand, Fig. 7(b) and (d) show that the K-means algorithm is more energy-efficient because of its clustering approach, which can exploit locality of events.

6 Conclusions

The proposed iMouse integrates WSN technologies into surveillance technologies to support intelligent mobile surveillance services. On one hand, these mobile sensors can help improve the weakness of traditional WSNs that they only provide rough environmental information of the sensing field. By including mobile cameras, we can obtain much richer context information to conduct more in-depth analysis. On the other hand, surveillance can be done in an event-driven manner. Thus, the weakness of traditional surveillance systems can be greatly improved because only critical context information is retrieved and proactively sent to users.

The prototyped iMouse system can be improved/extended in several ways. First, the way to navigate mobile sensors can be further improved. For example, localization schemes can be integrated to guide mobile sensors instead of using color tapes. Second, the coordination among mobile sensors, especially when they are on-the-road, can be exploited. Third, how to utilize mobile sensors to improve the network topology deserves further investigation.

References

[1] M. Valera and S.A. Velastin, "Intelligent distributed surveillance systems: a review," *IEE Proc.–Vision, Image and Signal Processing*, vol. 152, no. 2, pp. 192–204, 2005.

[2] W.C. Feng, E. Kaiser, W.C. Feng, and M.L. Baillif, "Panoptes: scalable low-power video sensor networking technologies," *ACM Trans. Multimedia Computing, Comm., and Applications*, vol. 1, no. 2, pp. 151–167, 2005.

[3] A. Hoover and B.D. Olsen, "Sensor network perception for mobile robotics," *Proc. IEEE Int'l Conf. Robotics and Automation*, pp. 83–88, 2000.

[4] J.H. Lee and H. Hashimoto, "Controlling mobile robots in distributed intelligent sensor network," *IEEE Trans. Industrial Electronics*, vol. 50, no. 5, pp. 890–902, 2003.

[5] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a moving object with a binary sensor network," *Proc. Int'l Conf. Embedded Networked Sensor Systems*, pp. 150–161, 2003.

[6] X. Ji, H. Zha, J.J. Metzner, and G. Kesidis, "Dynamic cluster structure for object detection and tracking in wireless ad-hoc sensor networks," *Proc. IEEE Int'l Conf. Comm.*, pp. 3807–3811, 2004.

[7] W. Zhang and G. Cao, "DCTC: dynamic convoy tree-based collaboration for target tracking in sensor networks," *IEEE Trans. Wireless Comm.*, vol. 3, no. 5, pp. 1689–1701, 2004.

[8] E. Ardizzone, M.L. Cascia, G.L. Re, and M. Ortolani, "An integrated architecture for surveillance and monitoring in an archeological site," *Proc. ACM Int'l Workshop Video Surveillance and Sensor Networks*, pp. 79–85, 2005.

[9] N. Heo and P.K. Varshney, "Energy-efficient deployment of intelligent mobile sensor networks," *IEEE Trans. Systems, Man and Cybernetics–Part A: Systems and Humans*, vol. 35, no. 1, pp. 78–92, 2005.

[10] G. Wang, G. Cao, and T.L. Porta, "Movement-assisted sensor deployment," *Proc. IEEE INFOCOM*, pp. 2469–2479, 2004.

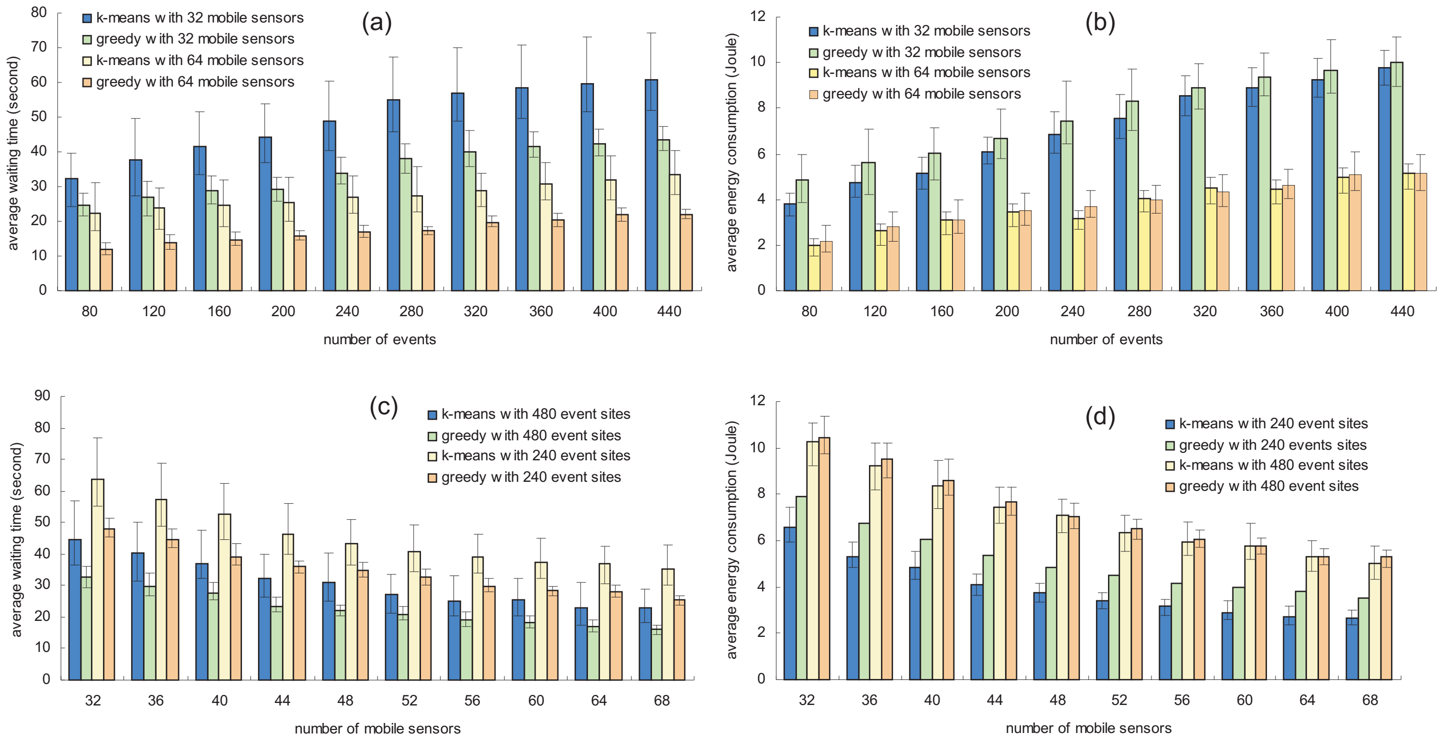

Fig. 7: Comparison of average waiting time and average energy consumption.

[11] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," *Proc. IEEE INFOCOM*, pp. 1293–1303, 2003.

[12] P. Basu and J. Redi, "Movement control algorithms for realization of fault-tolerant ad hoc robot networks," *IEEE Network*, vol. 18, no. 4, pp. 36–44, 2004.

[13] M. Demirbas, A. Arora, and M. Gouda, "A pursuer-evader game for sensor networks," *Proc. Sixth Symp. SelfStabilizing Systems*, pp. 1–16, 2003.

[14] C. Sharp, S. Schaffert, A. Woo, N. Sastry, C. Karlof, S. Sastry, and D. Culler, "Design and implementation of a sensor network system for vehicle tracking and autonomous interception," *Proc. IEEE European Workshop Wireless Sensor Networks*, pp. 93–107, 2005.

[15] L. Doherty, K.S.J. Pister, and L.E. Ghaoui, "Convex position estimation in wireless sensor networks," *Proc. IEEE INFOCOM*, pp. 1655–1663, 2001.

[16] T. He, C. Huang, B.M. Blum, J.A. Stankovic, and T.F. Abdelzaher, "Range-free localization schemes for large scale sensor networks," *Proc. ACM Int'l Conf. Mobile Computing and Networking*, pp. 81–95, 2003.

[17] Crossbow, "SPB400 – Stargate Gateway," http://www.xbow.com.

[18] J.R. Evans and E. Minieka, *Optimization algorithms for networks and graphs (second edition)*, Marcel Dekker Inc., 1992.

[19] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Academic Press, 2001.

[20] Crossbow, "MOTE-KIT2400 – MICAz Developer's Kit," http://www.xbow.com.

[21] MINDSTORM, "Robotics Invention System," http://mindstorms.lego.com/eng/default.asp.

[22] D. Johnson, T. Stack, R. Fish, D.M. Flickinger, L. Stoller, R. Ricci, and J. Lepreau, "Mobile Emulab: a robotic wireless and sensor network testbed," *Proc. IEEE INFOCOM*, 2006.

[23] J.P. Sheu, P.W. Cheng, and K.Y. Hsieh, "Design and implementation of a smart mobile robot," *Proc. IEEE Int'l Conf. Wireless And Mobile Computing, Networking And Comm.*, pp. 422–429, 2005.