A Real-Time Communication Framework for Wireless Sensor-Actuator Networks

Edith C. H. Ngai Michael R. Lyu
Department of Computer Science and Engineering
The Chinese University of Hong Kong
+852 2609 {8438, 8429}
{chngai, lyu}@cse.cuhk.edu.hk

Jiangchuan Liu
School of Computer Science
Simon Fraser University
+1 604-291-4336
jcliu@cs.sfu.ca

Abstract—Wireless sensor-actuator network (WSAN) comprises of a group of distributed sensors and actuators that communicate through wireless links. Sensors are small and static devices with limited power, computation, and communication capabilities responsible for observing the physical world. On the other hand, actuators are equipped with richer resources, able to move and perform appropriate actions. Sensors and actuators cooperate with each other: While sensors perform sensing, actuators make decisions and react to the environment with the right actions. WSAN can be applied in a wide range of applications, like environmental monitoring, battlefield surveillance, chemical attack detection, intrusion detection, space missions, etc. Since actuators perform actions in response to the sensed events, real-time communications and quick reaction are necessary. To provide effective applications by WSAN, two major problems remain: How to minimize the transmission delay from sensors to actuators, and how to improve the coordination among the actuators for fast reaction. To tackle these problems, we designed a real-time communication framework to support event detection, reporting, and actuator coordination. This paper explores the timely communication and coordination problems among the sensors and actuators. Moreover, we proposed two self-organized and distributed algorithms for event reporting and actuator coordination. Some preliminary results are presented to demonstrate the advantages of our approach.

Keywords – Sensor-actuator networks, real-time communications, event reporting, actuator coordination

TABLE OF CONTENTS

1. INTRODUCTION

The advance of hardware and engineering technology has turned distributed embedded systems such as sensors, actuators, and various mobile devices [1] into reality. Wireless sensor network (WSN), which is formed by a group of sensors, has become extremely popular in recent years with its capability of monitoring the environments [2]. However, sensors are passive devices for collecting data only and not interactive to the environments. Wireless sensor-actuator network (WSAN), which includes both actuators and sensors, then becomes an extension to WSN. Actuators are mobile devices that can make decisions and perform appropriate actions in response to the sensor measurements. They are resource-rich devices equipped with more energy, stronger computation power, longer transmission range, and usually mobile. One example of actuators is robots, which can communicate and perform different actions. On the other hand, sensors are small and low-cost devices with limited energy, sensing, computation, and transmission capability.

Sensors and actuators collaborate together to monitor and react to the surrounding world. Sensors perform sensing and report the sensed data to the actuators, while the actuators then carry out appropriate actions in response. WSAN can be applied in a variety of commercial, industrial, scientific, and military applications like environmental monitoring, sensing and maintenance in large industrial plants, military surveillance, medical sensing, attack detection, and target tracking. Apart from the above, the technologies developed can be applied to aerospace industries as well. For example, a number of sensors and actuators can be deployed on a planet for exploration. The sensors can collect data on the planet and report interesting data to the actuators. Then, actuators can go to particular locations for more detailed observations. They may collect some stone samples for bringing back to the space ship, capture high-resolution pictures, or record videos for deeper investigations.

A number of applications in WSAN require a quick response from the actuators to react to the environments. For example, actuators with water sprinkler are expected to arrive the scene of fire immediately to stop the spread of fire.

1——————————

[1] 0-7803-9546-8/06/$20.00© 2006 IEEE
[2] IEEEAC paper #1622, Version 2, Updated Oct, 20 2005

Similarly, actuators are expected to react as soon as possible in applications like intruder detection or object tracking. They have to make sure the person or the object is still in the reported area when they arrive [3].

In this paper, we propose a real-time communication framework, which provides timely reactions to the environments upon detection of an event. We focus on event-driven applications in a self-organized network. This network has no centralized control to the sensors and actuators. Moreover, the sensors report to the actuators only when their sensed data fall in the range of interest. For example, a group of sensors will report an event when the detected temperature is over a certain degree. An event can be any incident happening in the environments being monitored, such as a fire, a leakage of gas, or an attack.

In comparing with WSN, senor-to-actuator and actuator-to-actuator communications become a special feature in WSAN. Moreover, sensors in WSAN may have multiple potential destinations for event reporting, which is different from a single and static sink for data collection in WSN. WSAN usually contains multiple actuators available for reaction, so a good actuator-to-actuator coordination is necessary for providing a fast and effective response. Our solution explores the different capabilities and functionalities of sensors and actuators and offers efficient communication and coordination among them. It consists of two steps: First, a real-time and distributed event-reporting algorithm for sensors to send the application data to the actuators; and second, an efficient coordination algorithm for determining which actuators to perform the actions. The event-reporting algorithm allows the sensors to transmit data to actuators via the paths with minimum delay. Also, the data with more importance will be transmitted with higher priority. The actuator coordination algorithm allows the actuators to share the event information and make decisions on the proper reactions quickly.

The remaining of this paper is organized as follows. Section II presents the related work. In Section III, we outline our real-time communication framework, describe its workflow, and list out some notations. We present the details of the event detection and report in Section IV, followed by the actuator coordination and reaction in Section V. Finally, Section VI concludes this paper and offers some future directions.

2. Related Work

Real-time communications in wireless sensor networks are not new. Hu et. al. [4] propose a real-time communication protocol called SPEED, which provides real-time unicast, real-time area-multicast and real-time area-anycast for WSN. It achieves them by using a combination of feedback control and non-deterministic QoS-aware geographic forwarding with a bounded hop count. Lu et. al. [5] present a real-time communication architecture for large-scale wireless sensor networks. It describes a packet scheduling policy called *velocity monotonic scheduling* that inherently accounts for both time and distance constraints. Felemban et. al. [6] propose a novel packet delivery mechanism called Multi-path and Multi-Speed Routing Protocol (MMSPEED) for probabilistic QoS guarantee in wireless sensor networks. Multiple QoS levels are provided in the timeliness domain by guaranteeing multiple packet delivery speed options, while various requirements are supported by probabilistic multipath forwarding in the reliability domain. Although a number of protocols are proposed for WSN, they may not work well when applying directly to WSAN. There are additional considerations on the heterogeneous characteristics, network structure, and different operations among the sensors and actuators. Particularly, the sensor-to-actuator communications and actuator-to-actuator communications are the unique features of WSAN in comparison with WSN. Several investigations have been done on exploring the heterogeneous sensor networks [7, 8], but they do not cope with the special features and ways of operations in WSAN.

For WSAN, Hu et. al [9] propose an anycast communication paradigm. It constructs an anycast tree rooted at each event source and updates the tree dynamically according to the join and leave of the sinks. Their approach discovers the routes by flooding of the interest from the sinks. Also, E. Cayirci et. al [10] propose a power-aware many-to-many routing protocol. Actuators register the types of the data that they are interested by broadcasting a task registration message. Then, the sensors build their routing tables accordingly. In this scheme, the sensed data generated by any sensor node are forwarded to every actuator that is interested in that type of data, which may produce a lot of network traffic. Moreover, both approaches overlook the coordination among the sensors and actuators, which can be improved to increase the efficiency of event reporting and reaction. Furthermore, Melodia et. al. [11] propose a distributed coordination framework for wireless and actuator networks based on an event-driven clustering paradigm. All sensors in the event area forward their readings to the appropriate actors by the data aggregation trees. Their work assumes immobile actuators that can act on a limited area defined by their action range, and provides actuator-actuator coordination to split the event area among different actuators. Our work shares the similar event-driven hypothesis, but we propose an event-reporting algorithm which divides the event area into pieces of maps and transmits the sensed data with special ordering to reduce the response time. Moreover, our actuator coordination algorithm can support mobile actuators under sparse deployment.

Apart from the above, various papers discuss the research challenges and work on diverse topics in WSAN. Dinh et. al. [12] review the recent research achievements and open research issues, and evaluate the performance of three popular ad-hoc network routing protocols in handling actuator-to-actuator communications. Durresi et. al. [13] present a geometric broadcast protocol for WSAN (GBSA). It is a distributed algorithm where nodes make a local decision on whether to transmit based on a geometric approach. M. Coates [14] addresses the evaluation of causal relationships in WSAN, so that the expected marginal response of a system can be estimated. Hu and Cao [15] propose a two-level re-keying/re-routing scheme and a multiple-key management scheme to provide security for WSAN. Ganeriwal et. al. [16] consider a network where nodes have traction ability. They present methods for the network to be aware of its own integrity and use actuators to repair the coverage loss in the area being monitored.

3. Real-time Communications Framework

As mentioned earlier, we consider a network which consists of a large number of sensors and multiple actuators. Sensors are static and resource-limited devices for monitoring the environments and reporting events to the actuators. Actuators are mobile devices with richer resources and longer-range transmission that enable them to communicate with each other directly. We assume an event-driven model in the framework, in which sensors only send application data to the actuators when they discover an event. Upon receiving the events, actuators process the data and decide upon which to perform actions.

Many applications in WSAN require real-time response to the physical world. A real-time communication framework, which provides efficient communication and coordination among sensors and actuators, is essential to achieve a timely reaction. Such a framework relies on a low latency communication in the event reporting process from sensors to actuators, and a well-organized coordination algorithm that ensures a quick move to the event area by the actuators.

Indeed, our real-time communication framework focuses on providing a low latency event-reporting algorithm for sensor-to-actuator communication and an effective coordination algorithm among the actuators. Figure 1 shows the workflow of our real-time communication framework. Firstly, a group of sensors detect an event in the area where they are located. The sensors, which sensed the event earlier, start clustering and aggregating the data from their surrounding nodes. The event area is divided into pieces of maps according to the clusters formed. The maps and the corresponding data are reported to their closest actuators separately. Also, the data are transmitted in a special order, such that the most important data are sent first, with the following details later. This ensures the actuators can obtain a rough image on the event within a short time. Without waiting for the arrival of a complete report, the actuators can already start their coordination. They combine the maps they received and determine how many and which actuators should perform the actions. The size of the event area, together with the distance between the event area and the actuators, are significant factors in making a decision. Intuitively, a larger event area requires more number of actuators to perform the actions. Also, actuators located closer to the event are normally a better choice for reaction as they can arrive at the scene of event faster. Finally, the assigned actuators move to the event area and perform appropriate actions with proper location update mechanism. In this paper, we mainly focus on the event-reporting algorithm and the actuator coordination algorithm, but we also provide a comprehensive view from event detection and report regarding actuator coordination and reaction.

We assume that every sensor and actuator in the network knows its location. This is quite natural as nodes should be able to recognize the locations of the events in order to monitor, report, and react. Their locations can be obtained by equipping with a GPS receiver [17] or the position can be determined by some localization techniques [18, 19, 20]. Our framework also adopt the geographical-based protocols for routing as they scale up well and can adapt to the location changes of the actuators easily. In geographical-based routing, locations of nodes are exploited to route data in the network [21, 22]. A routing protocol can control certain system parameters in order to adapt to the current network conditions as well as the available energy levels. For example, the transmission delay can be considered when a node is selecting a neighbor that the message will be forwarded to [4]. Finally, we impose a virtual grid structure on the network, so a simplified coordinate representation can be applied to ease the formation and combination of the pieces of maps during event reporting and actuator coordination. Although the network area can be represented by grids of equal size, the initial *(x,y)* coordinate system is still employed as the basic scheme in our framework.

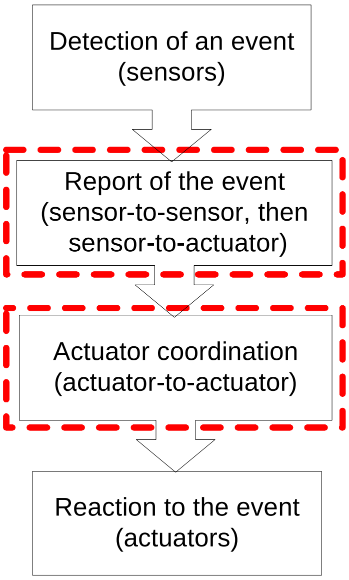

Figure 1 - Workflow of the Framework

For the ease of exposition, in Table 1, we list the major notations used throughout this paper.

Table 1. List of Notation

v	Sensor node
r	Sensor that builds and reports a piece of map in event area
a_i	Actuator
e	Event *ID*
h	Hops to the r in its cluster (depth)
v_h	Sensor node with depth h from r
n_v	Neighbors of node v
(x_v, y_v)	Coordinates of node v
$data_v$	Data collected by node v
S_r	Nodes on the map being reported by r
B_r	Boundary nodes on the map being reported by r
$meanv_h$	Mean from v_h and its descendents with depth $h+1$
$MEAN_{Sr}$	Overall mean among S_r
C	Centre of the map S_r
l_v	Location of node v
d_v	Distance from node v to r
$R(a_i)$	Voronoi cell associated with a_i

4. Event Detection and Report

Formation of Maps

Sensor actuator networks can be applied in event-oriented applications such as fire detection, gas leakage detection, intruder detection, etc. In these systems, the sensors collect data from the environment and report special events to the actuators. As we know, sensor networks contain a lot of redundant information. To reduce the network traffic, the sensor will aggregate event reports from the neighboring nodes. The sensors *r*, which detected an event the earliest, start the formation of maps as shown in algorithm 1:

Algorithm 1 Formation of Maps

```
for nodes r detected an event
    if (data aggregation not yet started)
        Broadcast DetectEvt (r, 0, e) msg. to n_r
    end if
end for

for nodes v receive DetectEvt msg. from v'
    if (h<max_hop && (v.event && ! v.reported))
        forward DetectEvt(v, h+1, e) msg. to n_v
    else
        reply ReplyEvt (meets boundary) msg. to v'
    end if
end for

for nodes v receive ReplyEvt msg.
    if (msg.==meets boundary)
        reply ReplyEvt(x_v, y_v, data_v, e) msg. to parent
    else
        concat own data and reply ReplyEvt msg. to
        parent

    end if
end for
```

For building a map, node *v* floods the event detection messages to nodes less than *max_hop* from *r*. Only those nodes which have detected an event and not yet reported forward the message to the next hop. We represent the nodes belonging to the cluster formed by node *r* as S_r. Multiple *r* can exist for the same event. They are usually the nodes that detected the event earlier than the others, which start constructing their own clusters. These clusters divide the event area into pieces of maps, as shown in Figure 2. Each map will be reported by one sensor *r* to one actuator. *Br* is the boundary nodes on the map of S_r, *where* $B_r \subseteq S_r$. Nodes in B_r are either *max_hop* hops from node *r*, or located on the boundary between two maps. Nodes in B_r stop forwarding the *DetectEvt* message and reply to the previous node with their coordinates, data value, and the event *ID*. The event *ID* may include the type of the event and the event discovery time, which is determined by *r*.

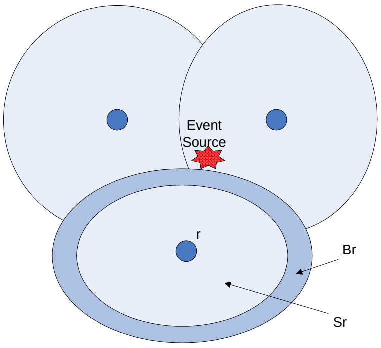

Figure 2 – Pieces of Maps Formed in an Event

Data Aggregation

Each piece of map can be represented by a tree structure with the sensor *r* as the root (Figure 3). When a node receives the replies from its descendent nodes, it concatenates its own reply and forwards them to the previous hop. Nodes with even number of depth *h* concatenate the reply with its own coordinates and sensed data, while nodes with odd number of depth *h* aggregate the data from their immediate descendents before forwarding them. Nodes with odd number of depth calculate the mean from the data values sensed by themselves and their descendents with the depth *h+1* (Algorithm 2). The following equation shows how $meanv_h$ can be calculated by node v_h.

Let h be no. of hops from sensor *r*,
v_h be the node in depth *h*,
$data_{v_h}$ be the data collected by node v_h,
$data_{v_{hj}}$ be the data collected by the j^{th} descendent of $v_{h,}$

$$mean_{v_h} = \left(\sum_{j=1}^{c_{v_h}} data_{v_{hj}} + data_{v_h} \right) / \left(c_{v_h} + 1 \right)$$, where

cv_h is the no. of immediate descendents of v_h.

Algorithm 2 Data Aggregation

```
for nodes $v_h \in S_r$ receive ReplyEvt msg.
    if (h==odd)        //node in odd no. of depth
        gather all data from its descendents $v_{hj}$ in h+1
        $mean_{v_h} = \left( \sum_{j=1}^{c_{v_h}} data_{v_{hj}} + data_{v_h} \right) / (c_{v_h} + 1)$
        remove $data_{v_{hj}}$ from ReplyEvt msg.
        concat {$mean_{v_h}$, $x_{v_h}$, $y_{v_h}$, e} to ReplyEvt msg.
        forward ReplyEvt msg. to parent in depth h-1
    else
        concat {$x_{v_h}$, $y_{v_h}$, $data_{v_h}$, e} to ReplyEvt msg.
        forward ReplyEvt msg. to parent in depth h-1
    end if
end for
```

Finally, the root sensor *r* collects all the coordinates and sensed data from the nodes in S_r. It is responsible for reporting this event to its closest actuator. The actuator, which receives this event report, is not necessary the one to perform the actions. At the same time, other actuators may receive information for the same event from other pieces of maps as well. All the informed actuators then coordinate and decide the reaction together. To speed up the coordination, sensor *r* divides the data on its map into different layers according to its importance. For example, the event type, location, and time are the basic information for starting actuator coordination, so they should be transmitted first.

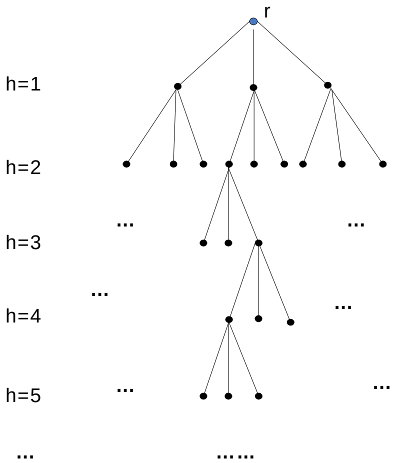

Figure 3 - Tree Representation of Nodes on the Map S_r

Layered Data Transmission

In our work, the data are divided into the base layer and the refinement layer. The base layer contains the type of event, the time when the event is first detected, the location of the map, and the mean value of the collected data. The mean gives the actuator a general idea on the condition of the map. It can be calculated by *r* with the following equation.

$$MEAN_{S_r} = \left(\sum_{i=0}^{N_{h=odd}} mean_i * (c_i + 1) \right) / \sum_{i=0}^{N_{h=odd}} (c_i + 1) \text{, where}$$

i refer to all nodes with odd no. of depth in the S_r.

As mentioned before, virtual grids are imposed on the network area, so the location of map can be simplified as follows:

Let $a \times b$ be the size of the virtual grid, locations (x_i, y_i) of each node *i* in B_r can be represented by (x_i', y_i') in a grid coordinate, where $(x_i', y_i') = (x_i/a, y_i/b)$. After collecting all (x', y') of B_r and removing the redundant coordinates, a set of grid coordinates representing the location of the map S_r is obtained.

The refinement layer contains all the means calculated by nodes with odd number of depth and their corresponding locations. These values are transmitted in a special order based on its distance from the centre *C* on the map *Sr*. $<mean_{di}, x_{di}, y_{di}>$ represents the mean and coordinates of a node *i* with distance d_i from *C*, where $\|l_i - l_C\| = d_i$.

The sensor *r* first forwards the data in the base layer, and then the refinement layer, as shown in Figure 4. It sends refinement layer with the following sequence, given $d_{\max} = Max\|l_j - l_C\| for \forall j \in S_r$:

$<mean_0, x_0, x_0>$: data from the node located at *C*
$<mean_{dmax}, x_{dmax}, y_{dmax}>$: data from nodes with distance d_{max} from *C*
$<mean_{(dmax/2)}, x_{(dmax/2)}, y_{(dmax/2)}>$: data from nodes with distance $d_{max/2}$ from *C*
$<mean_{(dmax/4)}, x_{(dmax/4)}, y_{(dmax/4)}>$: …
$<mean_{(dmax*3/4)}, x_{(dmax*3/4)}, y_{(dmax*3/4)}>$: …
……

<event type>	*<event time>*	*<location of map>*	*<* $MEAN_{Sr}$*>*

Base Layer

$< mean_0, x_0, y_0>$	$<mean_{dmax}, x_{dmax}, y_{dmax}>$	$<mean_{(dmax/2)}, x_{(dmax/2)}, y_{(dmax/2)}>$	$<mean_{(dmax/4)}, x_{(dmax/4)}, y_{(dmax/4)}>$	$<mean_{(dmax*3/4)}, x_{(dmax*3/4)}, y_{(dmax*3/4)}>$	…	…

Refinement Layer

Figure 4 - Base Layer and Refinement Layer

By the above sequence, the actuators achieve the event reporting gradually with more important information first. This allows them to start the actuator coordination much faster. They can then determine how many and which actuator(s) should perform the action as soon as possible, while finer information arriving later can help the planning of detailed action strategy. We adopt a location-based routing protocol with the consideration of transmission delay [4] for event reporting. More research work can be done in the future for finding the optimal actuator and path with minimum delay.

5. Actuator Coordination and Reaction

We present the algorithms for the combination of maps on the event area, actuator coordination, and location update for the actuators in this Section.

Combination of Maps

After an actuator receives the data in the base layer from the sensor *r*, it gets one piece of map in the event area. It then combines multiple maps if it receives more than one report on same type of event happening in the same area within time period t_e. Moreover, it will start communicating with other actuators located closely to the event area as well. They exchange information for combining their maps and approximating the size of the event as shown in Figure 5 and Algorithm 3. The coordination among the actuators starts before the arrival of the data in the refinement layer; therefore, it brings a quicker response from actuators.

The actuators get a general idea on the event location by finding the x_{max}, x_{min}, y_{max}, and y_{min} of the maps. They can also re-organize the event area by dividing the combined map into different rectangles. They can represent the event area by <x_{ℓ}, y_{ℓ}, x_r, y_{r}>, where x_{ℓ}, y_{ℓ}, x_r, y_r represent the x- and y- coordinates of the grids in lower-left and upper-right corners of a piece of rectangular map.

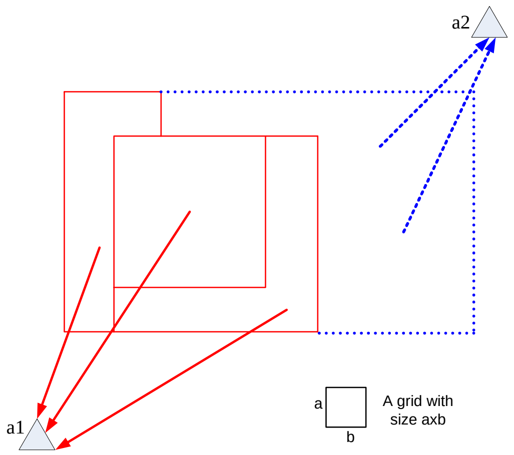

Figure 5 - Combinations of Maps

Algorithm 3 Combination of Maps

```
for each actuator $a$ on event $e$,
    if (received multiple $S_r$)
        Gather the $B_r$ in grid coordinates from all $S_r$
        Remove the redundant $B_r$
        Remove the connected $B_r$
        Store the remaining $B_r$ in $B_a$
    end if
    Exchange the $B_a$ with other actuators
    Remove the redundant $B_a$
    Remove the connected $B_a$
    Estimate the $B_a$ by finding lower-left and upper- right
      grids <$x_{min}$, $y_{min}$> and <$x_{max}$, $y_{max}$>
end for
```

Next, the actuators involved can determine how many and which of them will perform the appropriate actions. For example, actuators located closer to the event should have higher priority to react. We assume the actuators possess the same speed for moving and reacting to the same event for simple analysis. Since a larger event should be assigned with more actuators to response, we estimate the number of actuators *N* as *A/s*, where *s* is the approximate area size to be handled by one actuator.

Let *Area* be the size of the event area, m_i be the time actuator *i* takes to arrive at the attack area, and *w* be the rate of performing appropriate actions by actuators, then the total time *T* required for accomplishing the appropriate actions is calculated by:

$$w * \sum_{i=1}^{N} (T - m_i) = Area \text{ , where } N=Area/s.$$

During the coordination, the actuators can obtain more details of the event when the data in the refinement layer arrives. The refinement layer contains additional information for analysis, such as where the event source locates and the seriousness of the event. This may help the actuator plan an appropriate action sequence or the proper strategy in responding to the event.

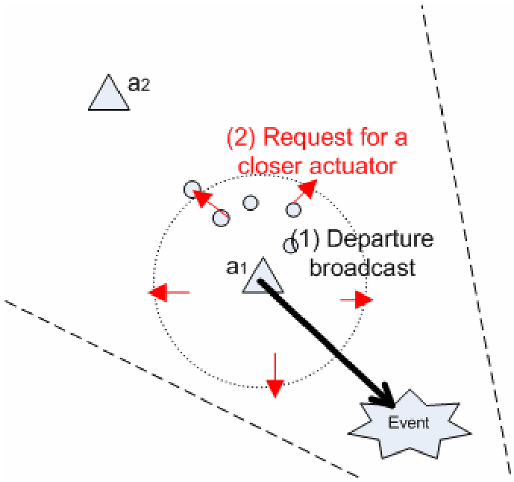

Figure 6 - Leave of Actuator a_1

Location Update

Since the actuators will move when they carry out appropriate actions, their locations should be updated for the sensors in the corresponding areas. An actuator will broadcast messages about its departure and arrival to the surrounding sensors. The sensors will then determine their potential actuator again. The process is shown in Figure 6.

We assume that the sensors always report the event to their closest actuators. Let $A = \{a_1, a_2, \ldots, a_n\}$ be the set of actuators in the network with their various location vectors, i.e. $l_i \neq l_j, \forall i \neq j$. The region $R(a_i)$ is called the Voronoi cell associated with a_i, where

$$R(a_i) = \{l \mid \|l - l_i\| \leq \|l - l_j\|, \forall i \neq j\}$$

Nodes in $R(a_1)$ should be informed for the departure of a_1, so they will look for another actuator. Transmission range R_t is required for broadcasting the departure message to all nodes being affected.

Let N_{a_1} be set of neighboring actuators of a_1,

$$R_t = \max\{(l_{a_i} - l_{a_1})/2\}, \forall a_i \in N_{a_1}$$

Nodes which are located in $R(a_1)$ and received a_1's departure message will look for another actuator nearby. They will broadcast messages to neighbors for requesting the locations of the nearby actuators. Nodes located on the boundary of $R(a_1)$ will reply with the locations of their closest actuators. This information will be propagated hop-by-hop to the nodes in $R(a_1)$. Each of them can then choose the closest actuator as its new actuator for event reporting. Similarly, the actuator will broadcast its arrival and its new location to the surrounding sensors when it arrives at another place after event reaction. The affected sensors will then update their records of actuators.

6. CONCLUSION

In this paper, we present a real-time communication framework for wireless sensor-actuator networks. It provides an efficient event-reporting algorithm, which reduces the network traffic and minimizes the transmission delay by dividing the event area into smaller pieces of maps. The data are aggregated and further divided into different layers according to their importance. It is then transmitted to the closest actuator in the order of significance. This approach enables the actuators to start coordination without waiting for the arrival of the complete event information. Multiple actuators can combine their pieces of maps and decide on the appropriate actuator(s) to perform the actions as soon as possible. The assigned actuators will broadcast their move to the surrounding nodes, so the affected sensors can update the actuator information dynamically for future reporting. We also consider the heterogeneous characteristics and functionalities of sensors and actuators, and offer a distributed, self-organized, and comprehensive solution for real-time communications in WSAN. Our future work will focus on formalizing the current approach, providing performance analysis, and evaluating the solution by experiments. Moreover, we are interested in enhancing the efficiency and reliability of the current approach.

7. ACKNOWLEDGEMENTS

The work described in this paper was fully supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK4205/04E).

REFERENCES

[1] C.H. Ngai, M.R. Lyu, and R.T. Chin, “An authentication service against dishonest users in mobile ad hoc networks,” in *Proc. 2004 IEEE Aerospace Conference*, Big Sky, Montana, U.S.A., March 6-13 2004.

[2] J. N. Al-Karaki and A. E. Kamal, “Routing techniques in wireless sensor networks: a survey,” *Ad Hoc Networks Journal(Elsvier)*, issue 3, 2005, pp. 325-349.

[3] I. F. Akyldiz and I. Kasimoglu, “Wireless sensor and actor networks: research challenges,” *Ad Hoc Networks Journal(Elsevier)*, Oct 2004.

[4] T. He, J. Stankovic, C. Lu, and T. Abdelzaher, “SPEED: a real-time routing protocol for sensor networks,” in *Proc. of IEEE ICDCS, Providence*, RI, USA, May 2003, pp. 46-55.

[5] C. Lu, B. M. Blum, T. F. Abdelzaher, J. A. Stankovic, and T. He, “RAP: a real-time communication architecture for large-scale wireless sensor networks,” in *Proc. of IEEE RTAS*, San Jose, CA, USA, Sep 2002.

[6] E. Felemban, C.-G. Lee, E. Ekici, R. Boder, and S. Vural, “Probabilistic QoS guarantee in reliability and timeliness domains in wireless sensor networks,” in *Proc. of IEEE Infocom 2005*, Miami, FL, USA, Mar 2005.

[7] V. P. Mhatre, C. Rosenberg, D. Kofman, R. Mazumdar, N. Shroff, “A minimum cost heterogeneous sensor network with a lifetime constraint,” *IEEE Transaction on Mobile Computing*, vol. 4, no. 1, Jan/Feb 2005.

[8] M. Yarvis, N. Kushalnagar, H. Singh, A. Rangarajan, Y. Liu, and S. Singh, “Exploiting Heterogeneity in Sensor Networks,” in *Proc. of IEEE Infocom 2005*, Miami, FL, USA, Mar 2005.

[9] W. Hu, S. Jha, and N. Bulusu, “A communication paradigm for hybrid sensor/actuator networks,” in *Proc. of the 15th IEEE Int. Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, Bacelona, Spain, 5-8 Sep 2004.

[10] E. Cayirci, T. Coplu, and O. Emiroglu, “Power aware many to many routing in wireless sensor and actuator networks,” in *Proc. of the Second European Workshop*, 31 Jan-2 Feb 2005, pp. 236 – 245.

[11] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz, “A distributed coordination framework for wireless sensor and actor networks,” in *Proc. of ACM Mobihoc*, Urbana-Champaign, IL, USA, 2005, pp.99-110.

[12] D. V. Dinh, M. D. Vuong, H. P. Nguyen, H. X. Nguyen, “Wireless sensor actor networks and routing performance analysis,” *International Workshop on Wireless Ad-hoc Network*, London, UK, 23-26 May 2005.

[13] A. Durresi and V. Paruchuri, “Geometric broadcast protocol for sensor and actor networks,” *Int. Conference on Advanced Information Networking and Applications*, 28-30 Mar 2005, pp. 343 – 348.

[14] M. Coates, “Evaluating causal relationships in wireless sensor/actuator networks*,” IEEE Int. Conference on Acoustics, Speech, and Signal Processing*, 18-23 Mar 2005, pp.937 – 940.

[15] F. Hu and X. Cao, “Security in wireless actor & sensor networks (WASN): towards a hierarchical re-keying design,” *Int. Conference on Information Technology Coding and Computing (ITCC'05)*, Las Vegas, NV, Apr 2005, pp. 528-533.

[16] S. Ganeriwal, A. Kansal, and M. B. Srivastava, “Self aware actuation for fault repair in sensor networks*,” IEEE Int. Conference on Robotics and Automation*, 26 Apr-1 May 2004, pp. 5244 – 5249.

[17] J.G. McNeff, “The global positioning system,” *IEEE Transactions on Microwave Theory and Techniques*, vol. 50, issue 3, Mar 2002, pp. 645 - 652.

[18] L. Hu and D. Evans, “Localization for mobile senosr networks,” in *Proc. of ACM Mobicom*, Philadelphia, PA, USA, 26 Sept - 1 Oct 2004.

[19] A. Savvides, C.C. Han, and M.B. Srivastava, “Dynamic fine-grained location in ad hoc networks of sensors,” in *Proc. of ACM Mobicom*, Rome, 2001, pp. 166-179.

[20] T. He, C. Huang, B.M. Blum, J.A. Stankovic, and T. Abdelzaher, “Range-free localization schemes for large scale sensor networks,” in *Proc. of ACM Mobicom*, San Diego, CA, 2003, pp. 81-95.

[21] J. Li, J. Jannotti, D. D. Couto, D. Karger, and R. Morris, “A scalable location service for geographic ad hoc routing,” in *Proc. of ACM MobiCom*, Boston, Massachussets, 2000, pp. 120-30.

[22] Y. Xu, J. Heidemann, D. Estrin, “Geography-informed energy conservation for ad-hoc routing,” in *Proc. of ACM Mobicom*, Rome, 2001, pp. 70-84.

BIOGRAPHY

Edith C. H. Ngai *received the B.E. degree in Computer Engineering, in 2002, and the M.Phil. degree in Computer Science and Engineering, in 2004, from The Chinese University of Hong Kong, Shatin, Hong Kong. She currently is the Ph.D student in the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong. Her research interests include mobile ad hoc networks, sensor networks, network security, and video information processing.*

Michael R. Lyu *received the B.S. (1981) in electrical engineering from National Taiwan University, the M.S. (1985) in computer engineering from University of California, Santa Barbara, and the Ph.D. (1988) in computer science from University of California, Los Angeles. He is a Professor in the Computer Science and Engineering Department of the Chinese University of Hong Kong. He worked at the Jet Propulsion Laboratory, Bellcore, and Bell Labs; and taught at the University of Iowa. His research interests include software reliability engineering, software fault tolerance, distributed systems, image and video processing, multimedia technologies, and mobile networks. He has published over 200 papers in these areas. He has participated in more than 30 industrial projects, and helped to develop many commercial systems and software tools. Professor Lyu was frequently invited as a keynote or tutorial speaker to conferences and workshops in U.S., Europe, and Asia. He initiated the International Symposium on Software Reliability Engineering (ISSRE), and was Program Chair for ISSRE'1996, Program Co-Chair for WWW10 and SRDS'2005, and General Chair for ISSRE'2001 and PRDC'2005. He also received Best Paper Awards in ISSRE'98 and in ISSRE'2003. He is the editor-in-chief for two book volumes: Software Fault Tolerance (Wiley, 1995), and the Handbook of Software Reliability*

Engineering (IEEE and McGraw-Hill, 1996). He has been an Associate Editor of IEEE Transactions on Reliability, IEEE Transactions on Knowledge and Data Engineering, and Journal of Information Science and Engineering. Professor Lyu is an IEEE Fellow.

Jiangchuan Liu *(S'01-M'03) received the B.Eng degree (cum laude) from Tsinghua University, Beijing, China, in 1999, and the Ph.D. degree from The Hong Kong University of Science and Technology in 2003, both in computer science. He is currently an assistant professor in the School of Computing Science, Simon Fraser University, BC, Canada, and was an assistant professor at The Chinese University of Hong Kong from 2003 to 2004. His research interests include Internet architecture and protocols, media streaming, wireless ad hoc networks, and service overlay networks. He serves as TPC member for various international conferences, including IEEE INFOCOM and IWQoS. He was Information System Co-Chair for IEEE INFOCOM'04, and a guest-editor for ACM/Kluwer Journal of Mobile Networks and Applications (MONET), Special Issue on Energy Constraints and Lifetime Performance in Wireless Sensor Networks. He is an editor of IEEE Communications Surveys and Tutorials. He is a member of IEEE and ACM, and an elected member of Sigma Xi.*