Data Management of Mobile Object Tracking Applications in Wireless Sensor Networks

Jin Zheng
School of Information Science and Engineering
Central South University, Changsha 410083, China
Email: zhengjin@mail.csu.edu.cn

Weijia Jia*
Department of Computer Science
City University of Hong Kong, Kowloon, Hong Kong
Email: itjia@cityu.edu.hk

Guojun Wang
School of Information Science and Engineering
Central South University, Changsha 410083, China
Email: csgjwang@mail.csu.edu.cn

***Abstract*—In this paper, we focus on the data management problem of object tracking applications in wireless sensor networks. We propose link-segment storage and query protocol to track mobile object in wireless sensor network dynamically. The main idea of our protocol is to combine advantages of local storage with data centric storage methods to support the query of object movement information efficiently. Object's movement information will be stored in node near detecting sensor and the relation of storage nodes is maintained using multiple access entry linked list along the moving path of the object. Index-store node (a designated node) stores the access entry messages of linked list. Performance analysis and simulation studies show that the proposed protocol is energy efficient, with high probability of successful query and low query latency.**

***Index Terms*—Wireless sensor networks, object tracking, linked list, segment.**

I .INTRODUCTION

Recent advances in Micro-Electro-Mechanical Systems (MEMS) technology, wireless communications, and digital electronics have enabled the development of low-cost, low-power, multifunctional sensor nodes with small size and short communication range [1]. A Wireless Sensor Network (WSN) consists of a large number of tiny sensor nodes deployed in an area of interest. These sensor nodes are equipped with data processing, sensing, and communication capabilities. They are usually powered by battery. However, replacing battery is not only costly but also impossible in many situations. At the same time, many applications of wireless sensor networks require the network to support Quality of Service (QoS), such as low query latency and high probability of successful query. Thus, energy efficiency and QoS are critical considerations in the design of large-scale WSNs.

In WSNs, object tracking is a popular and important application. In this application, users need to know the location of the detected object. Most object tracking applications require real-time location report. However, some applications also need the complete or partial historical movement information of the detected object. For instance, it needs the animal's movement information in the application of wild animals protecting.

The object tracking protocols are classified into cluster-based and non-cluster-based protocols in WSNs. In cluster-based protocols [2-5], when a non-cluster head sensor node detected an object, it forwards information to its cluster head. The cluster head collects and propagates the information to a sink. This approach reduces the required communication bandwidth and energy consumption. In non-cluster-based protocols, there is not any node to serve as cluster head in WSNs. When a sensor detects an object, it records the object's moving information in its local memory. A user issues a request to WSNs when he/she wants to know the location of the tracked object. If a sensor has the information of the tracked object, it replies the information to the user.

Tseng et al. proposed a novel protocol based on the mobile agent [6]. Once a new object is detected, a mobile agent will be initiated to track the roaming path of the object. The mobile agent will choose and stay in a sensor that is the closer to the tracked object. The mobile agent cooperates with its neighbors to accomplish the task of

Manuscript received November 20, 2008; revised December 20, 2008; accepted January 26, 2009. *Corresponding author: Prof. Weijia Jia, Email: itjia@cityu.edu.hk.

object tracking and it propagates object's location to sink node.

In order to reduce energy consumption, we can reduce the number of data traffic [7], or reduce the frequency of communications [8], by reducing the number of sensors involved in object tracking and also by reducing the amount of data being sent to the cluster head in object tracking [9]. Balancing energy usage in the network is another approach and one way of performing this is by using of mobile sinks [10].

For saving energy, the prediction-based methods [11-13] are used to predict the location of mobile object. When a sensor detects an object, it forwards the object's information to its cluster head. The information contains the location, velocity and moving direction of the object. The cluster head calculates and predicts the location of the object and then it multicasts wakeup information to the predicted area (forwarding area). This method can reduce the number of active nodes so it is energy efficient. However, the object tracking may be failed when multicast method meet a hollow grid in sensor network and location update traffic is not reduced. In references [14, 15], prediction-based reporting for object tracking was proposed, which uses prediction to reduce location update traffic. But the efficiency depends on the accuracy of the prediction model, and in real applications, it is not easy to get an efficient prediction model because the object may move in a random way. At the same time, updating the prediction model needs a lot of extra traffic.

Object tracking wireless sensor networks need some storage policies to store data of moving object. According to the reference [16], the data storage scheme was divided into three categories, namely, external storage, local storage, and data-centric storage. In external storage, a node directly sends detected data to a sink. It has to transfer its detected data to the sink node, so that users can access the data later. Query requests do not need to be forwarded to the network, and query response latency is very low. The disadvantage is that each node which detects an object needs to deliver its data to the sink node. But users may only request some parts of the detected data. This method may waste a lot of power of the entire network.

In local storage, when a sensor detects an event, it stores the data in its local storage. But query requests need to be flooded to all nodes in order to get the tracking data.

In data-centric storage, the data is stored according to the data type. The same type of data will be stored in a node determined by the name associated with the sensed data. Users can send query requests to the node according to data type and then retrieve data. However, due to the fact that the object is moving, a lot of update traffics are required. In references [17, 18], hierarchical distributed location database is proposed where each database site covers a specific geographical grid and contains location information about all objects residing in it. The distributed location database forms a location tree structure. In reference [19], deviation-avoidance and highest-weight-first scheme is proposed to construct an object tracking tree to reduce the communication cost of location update. But maintaining location tree requires extra energy cost and the tracking information is not reported to users. In addition, complete or partial trajectory tracking and query can not be supported in this scheme. In reference [20], EASE is proposed to maintain two versions of object location data in the network. High-precision data is kept at a local storage node which closes to a moving object in order to reduce long-distance traffic resulting from remote updates. Meanwhile, the same data with a lower precision is replicated at some designated storage nodes which are known to users in order to reduce the querying traffic. Accordingly, a query is answered by the designated storage node if its precision constraint is weaker than that specified by the approximation precision. Otherwise, the query is forwarded to the local storage node for resolution. EASE optimizes the network performance by reducing both the updating and querying traffic. But the query latency is high for high-precision query because query message must be forwarded to the designated storage node and then local storage node. At the same time it can't support tracking query efficiently.

All above methods focus on how to track current location of the object and they do not consider the query for trajectory. Hua-Wen Tsai, Chih-Ping Chu and Tzung-Shi Chen proposed a dynamical object tracking (DOT) protocol for sensor network [21]. A group of sensors forms an envelopment-net to besiege and to detect the object. When the object moves, the envelopment-net follows the object and a set of ingress nodes which record object's movement information is kept. The sequence of ingress nodes is the object traces (face-track). The ingress node can pilot the query to the object position. The query can obtain the object information from the ingress node directly so this protocol can decrease the frequency of querying. DOT protocol has better performance than the other flooding-based query methods. But if a user wants to query object's movement information, query requests need to be flooded to ingress node. When the query frequency is high, more energy will be wasted because of flooding. On the other hand, DOT utilizes face structure to object tracking, thus, face discovery and face maintaining process are complex and large energy consumption.

Some object tracking applications can tolerate delays in data collection and processing. Taking advantage of the delay tolerance, a delay-tolerant trajectory compression (DTTC) was proposed [22]. In DTTC, a cluster-based infrastructure is built within the network. Each cluster head compresses an object's movement trajectory detected within its cluster by a compression function. The cluster head communicates only the compression parameters, which not only provide the sink node with expressive yet traceable models about the object's movements, but also significantly reduce the total amount of data communication required for tracking operations.

Shin-Wei Ho and Gwo-Jong Yu proposed an energy

efficient method that can be applied to applications of WSNs, which need to track moving object's historical data [23]. The main idea is to create a linked list along the moving path of an object, and to query through the linked list to collect entire data of a moving object with small control overhead. This method uses local storage and it doesn't need to flood any query request. However, due to the fact that the linked list has only one access entry (head of a linked list), all query requests must access through the head to the tail along the linked list in order to get a response. So it can't support real-time location query and partial historical trajectory query efficiently because of latency and a lot of energy waste, especially when the linked list length is very long.

In this paper, we focus on data management problem to support users to query object tracking and obtain the object location effectively. We proposed an efficient data storage and query protocol which is improved from EEDS [23].

The rest of this paper is organized as follows: Section II introduces the system model. In Section III, data storage and query protocol is proposed. Sections IV and V describe performance analysis and simulation studies. Finally, Section VI concludes this paper.

II. The System Model

The application scenario of this paper is that users need to get the information of moving object in a period (complete or partial historical data of moving objects) in low latency. Assume that the sensor network is composed of n homogeneous sensors. We denote the *ith* sensor by s_i and the whole node set $S = \{s_1, s_2, \ldots, s_n\}$ where $|S|=n$. The nodes use multi-hop routing to send data to destination node. We assume that the nodes are aware of both their locations and neighboring nodes within their radio ranges. The nodes are either static or move with slow speed, so we can assume they are in the same small area within some time span. Due to energy depletion and faults, some nodes may fail and each sensor has a unique id.

III. Link-Segment Storage And Query

A. Initialization

When the network is deployed, nodes communicate with each other, and the global information such as boundary of the object monitoring field and grid size is broadcasted to the network. Each node, e.g. s_i, then assigns itself a grid id *GID* according to its coordinates:

$$GID\ (s_i) = \left(\left\lceil \frac{s_i.y}{gl} \right\rceil - 1\right) \times \left\lceil \frac{length}{gl} \right\rceil + \left\lceil \frac{s_i.x}{gl} \right\rceil \quad (1)$$

where $(s_i.x, s_i.y)$ denotes the relative x and y coordinates of s_i in the sensor field, respectively, gl is the length of the square grid, and *length* is the width of the field. The network is then organized into partitioned grids, and each grid contains a Cluster Head (CH), which is responsible for data fusion and coordination of the storage within its grid. The CH selection problem in a grid can be solved by simply using clustering algorithm [24]. A CH can adjust its communication range in its CH duty time to enable it communicates with its neighbor CH nodes directly. The network is logically partitioned here, so CHs do not need to maintain the topology of the grid (and so avoids the cost). When a node moves to a new grid, it just assigns itself a new *GID* by equation (1) according to its new coordinates. When a CH moves to a new grid, the neighboring nodes could notice its movement and reselect a node to be the new CH. Some CHs which detect event data send index messages which include object ID, timestamp and their grid ID to the index-store node (the designated storage node). The index-store node can forward the query message to the most suitable grid and from there visiting sequence nodes to get current position information of the object or historical tracking data according to the query.

B. Description of Virtual Linked List

Because an object may move arbitrarily in the sensing field, and in some object tracking applications, users may only need to get tracking information of partial duration, for example, users need to get part of tracking information or only need to get current position of the object. The event data must be stored in the grid which the object is located in. In the proposed protocol, a virtual link-segment list is constructed to help users retrieve historical tracking data or get the position information in a low latency according to query request. When a moving object enters the monitoring field, the sensors that detect object will execute object tracking algorithm by data exchange [9]. When a CH detects an event, the CH will broadcast gridID and objectID of the detected object. The previous CH can construct a link through overhearing. The link points to the next grid of the moving object at next time. When a CH lacks cache space to store new event data, it will select a neighbor sensor called Storage Node (SN) to store event data. Because the CH's ID is gridID, we call the link as a virtual link. The SN election is based on residual storage space.

For example, as shown in Fig.1, three CHs (grids) detect events of a moving object at time T1, T2, and T3, respectively. The link field stores ID of the next grid. The CH which detects the object will record the next gridID and select a SN to store the event data. At time T3, the SN is itself a CH node, because its residual cache space is bigger than its neighbors.

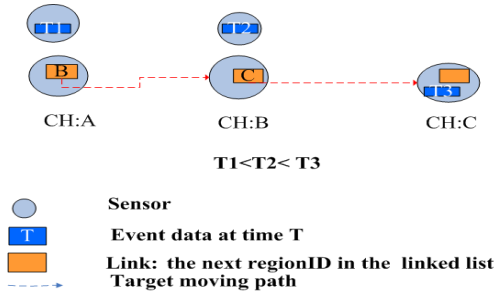

Figure 1. Illustration of linked list storage

When a CH detects an event, the CH will transfer the event data to a SN in its grid and record the SNID about the object at its index table. This protocol stores the event data in SN instead of in CH, and the CH only needs to record the index to help locate the event data when a query needs it. When a grid elects a new CH, the old CH transfers its index table to the new CH. The principle of cooperative storage is that no single node has enough resources to store information about all the event data.

C. Link-Segment Storage

The main objectives of our protocol are fault tolerance, energy efficiency and QoS.

One of the characteristics of sensor nodes is easy to be failed because of limited power or other external reasons. If a sensor node or all the sensor nodes of a grid are failed, the linked list will be broken, and the next node can not be visited without flooding. At the same time, in real tracking applications, there are many kinds of query requests, for example, some queries only need to get the current location of the moving object, and some queries only need to get partial historical data such as event data during a period of time. If the linked list is very long, the query latency is high as the linked list is visited from its head to tail.

In order to enhance the probability of successful query and support the query requests listed above efficiently, we propose link-segment storage and query protocol which is improved on EEDS. The main idea of our protocol is that not only the CH, which is the head of the virtual linked list, that first detected, needs to send a short index message including the objectID, timestamp and its gridID to the index-store node, but also those CHs which are at *Ksth* time in the linked list will send an index message to the index-store node. Here, *Ks* is the length of a segment in a linked list and it is a predefined value. So the protocol makes a linked list which has multiple access entries. At the same time, the event data is stored in a SN which is a neighbor of a CH, and this cooperative storage scheme can solve the limited storage space problem of sensor nodes.

If a Clusterhead detects that an event happens
{Store the event data in an SN and record the *SNID*;
Broadcast *objectID*, *timestamp* and *gridID*;
If it doesn't receive *k* //have no previous node
// *k* is a counter in the process of constructing a linked-segment
{It sends an index message that includes *objectID*, *timestamp* and *gridID* to index-store node;
k=1;}
Else
{*k*=*k*+1;
if *k*>=***Ks***
{It sends an index message that includes *objectID*, *timestamp* and *gridID* to index-store node;
k=1;} }
}

Figure 2. Construction of link-segment algorithm

If the previous node receives broadcast message from neighbor grid about *objectID* it has recorded at last time
{It stores the *gridID* in its link field of *objectID*;
Send *k* to the grid *gridID;* }

Figure 3. Link generating algorithm in previous node

Fig.2 and Fig.3 show the algorithm of constructing a linked-segment list. The parameter *k* used in Fig.2 and Fig.3 is a counter in the process of constructing a linked-segment list. The properties of link-segment are fault tolerant, even if there are failed nodes in the linked list, and the subsequent nodes can be visited without flooding; the latency associated with querying object's current position or the event data in a time span of a moving object is low from suitable access entries to the node in the linked list; and energy efficient because it takes advantages of local storage and data centric storage, and it doesn't transfer event data not be queried. The link-segment storage is demonstrated in Fig. 4.

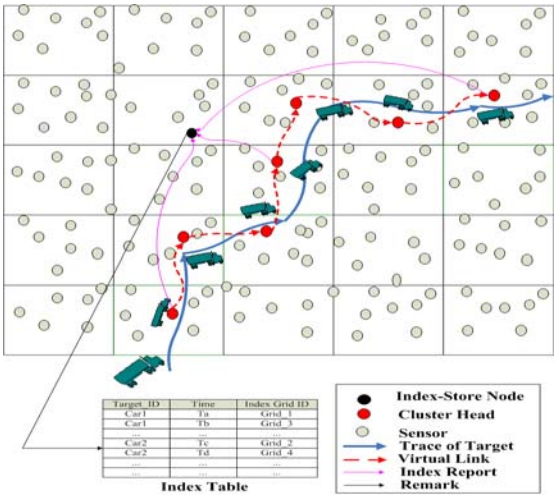

Figure 4. Illustration of link-segment storage

D. Index Table

In order to retrieve event data or position of tracking object without flooding, an index-store node has an index table that contains timestamp, objectID and gridID. The index table is shown in Table 1 ($T_a<T_b<T_c<T_d$). If a query needs to get the current location of Object1, the index-store node will forward the query packet to Grid4 instead of Grid1 according to the index table, and if a query needs to get the tracking information of Object1 at the duration of (t_1,t_2) ($T_b<t_1<T_c$, $T_c<t_2<T_d$), the index-store node will forward the query packet to Grid2. For example, Table 1 shows that Object1's location at T_a, T_b, T_c and T_d is Grid1, Grid2, Grid3 and Grid4, respectively.

TABLE I. INDEX TABLE IN AN INDEX-STORE NODE

Timestamp	ObjectID	GridID
T_a	Object1	Grid1
T_b	Object1	Grid2
T_c	Object1	Grid3
T_d	Object1	Grid4

Because the cache resource of a sensor is limited, our protocol requires a CH to select an SN in its neighbors to store event data, it takes advantage of cooperative resources within a grid to store event data. In order to support users to retrieve event data, a CH has an index table which contains Timestamp, ObjectID, SNID and Next GridID. The SNID field is the ID of a SN which stores the event data about an object, and Next GridID is the gridID of object' position in next time. The index table at CH is shown in Table 2. Table 2 shows which object enters and leaves a grid at what time. For example, at time T_1, Object2 enters the grid and it moves into Grid1 at next time, and the event data is stored at SN1. At time T_4, Object4 and Object5 enters the grid and Object4 moves into Grid4 at next time, Object5 moves into Grid5 and the two object's movement data is stored at SN3.

TABLE II. INDEX TABLE IN CH

Timestamp	ObjectID	Storage Node	Next GridID
T_1	Object2	SN1	Grid1
T_2	Object3	SN1	Grid2
T_3	Object1	SN2	Grid3
T_4	Object4	SN3	Grid4
T_4	Object5	SN3	Grid5

E. Query

When an index-store node receives a query request, it will select one or more suitable access entries (gridIDs) from the index table of an object according to the query request and forward query to the grid in the linked list, then visit and get data beginning from that grid's CH to the next CH depending on virtual link field. If the link is broken, the query packet will be forwarded to the next access entry. The query algorithm is shown in Fig.5.

```
Send query to index-store node;
Select one or more access entries from index table
    according to query request;
Forward the query to the access entry and visit nodes
   along the linked list from there;
 If the linked list is broken then
      Forward query to the next access entry of that
   linked list;
```

Figure 5. Query algorithm

F. Mobility

In many applications of WSNs, some or all sensor nodes may be mobile. The grid structure is applied to resolve the node mobility problem. If a storage node moves out of its grid boundary, it passes its data to a node in the grid and it sends a message to its CH, then the CH updates its index table. In this way, the query packet can get the event data easily even if sensor nodes are mobile.

Ⅳ. PERFORMANCE ANALYSIS

Let n denote the total number of nodes in a sensor network. $Event_{path}$ denote the length of a linked list, $Query_{num}$ denote the number of query requests of an object, the parameter Ks is the length of a segment link, and $Event_{query}$ is the average length of being queried partly in a linked list.

A. Energy Cost Analysis

According to the analysis described in EEDS [24], EEDS is more energy efficient than DCS, ES and LS when n is large, so the proposed protocol is only compared with EEDS. The total amount of control messages about query that EEDS generates can be summarized as the following Formula (2).

$$(Query_{num}+1)*O(\sqrt{n})+Event_{path}*O(1) + 1/2*Query_{num}*Event_{path}*O(1) \quad (2)$$

The first item of Formula (2) is the number of query packets sent to the head of a linked list and the index message of the head of a linked list to an index-store node, the second item is the cost of constructing a linked list and the third item is the cost of visiting a linked list. Because every query must visit linked list beginning from the head of a linked list, so the average cost of queries is 1/2 times of a linked list length.

EEDS combines the advantages of local storage and data centric storage, and it is energy efficient. However, due to the fact that EEDS has only one access entry of a linked list, if any node in a linked list fails, the subsequent nodes can not be easily visited without flooding, and it doesn't support other kinds of query efficiently. In order to enhance the probability of successful query and support more kinds of query efficiently in low latency, the proposed scheme increases access entry of the linked list. The total control messages (excluding the event data in a query) of our scheme can be formulated as Formula (3).

$$(Query_{num}+Event_{path}/K_s)*O(\sqrt{n})+Event_{path}*O(1) + Query_{num}*Event_{query}*O(1) \quad (3)$$

From Formula (3), we can see that our protocol needs to transfer a small number of index messages to an index-store node. However, due to the fact that the index packet is very small, the cost of energy for index message is not high, and the query can visit a linked list from any suitable access entry instead of its head, and the query forwarding cost has no relation to the length of the linked list. When the total length of a linked list is large and the length being queried part is small, the total cost of energy is decreased drastically.

B. Query Success Analysis

In EEDS, there is only one access entry of a linked list, and it can be broken for any node failure. This property reduces query success ratio, and EEDS applies backup technology to reduce the effect of faulty sensor. It adopts multiple sensors to store event data and link. The probability of successful query is

$$P_{query}=(1-p_{nfailure}{}^{k})^{Len} \quad (4)$$

When the length of a linked list Len is large, k (the number of backup sensors) must be large enough to guarantee the probability of successful query. For

example, when the probability of sensor node failure ($P_{nfailure}$) is 0.1, *Len* (the length of a link) is 200, and it needs at least five nodes to act as backup nodes for every node in a linked list to keep 90% query success. More backup nodes are needed for larger *Len*. This scheme needs a large part of cache capacity which is quite limited in sensor nodes. In the linked-segment protocol, a link is stored at CHs, and a CH has the most residual energy in a grid, so the probability of cluster head failure ($P_{CHfailure}$) is very low. The probability of successful query of the linked-segment scheme is

$$P_{query} = (1 - p_{CHfailure})^{Ks} \quad (5)$$

The *Ks* is the length of a segment and the probability of query success has no relation with the length of a linked list (*Len*), and the probability of successful query can be guaranteed with a suitable parameter *Ks*. It doesn't need backup nodes to enhance probability of successful query. It is cache capacity efficient.

C. Latency Analysis

In EEDS, all query requests must be forwarded to the head of a linked list, however, in our proposed protocol, a query can begin from any suitable access entry instead of a linked list's head. So the query latency is lower than in EEDS.

From the analysis listed above, the proposed protocol can support various queries efficiently in energy cost, low latency and high probability of successful query.

Ⅴ. Simulation Studies

In this section, we use OMNET++ to validate the proposed data storage and query protocol.

EEDS is more energy efficient than ES, LS and DCS for large scale networks, so we only compare the proposed protocol with EEDS. The simulation focuses on the energy consumption (total message), probability of successful query and query latency. Table 3 shows the parameters used in OMNET++ simulations. Fig.6 shows the simulation results which show the relation between the total message which represents energy cost and segment length. Here, it shows that the total message of the proposed protocol is lower than EEDS. Fig.7 shows the relation of query latency with segment length. We can see that the query latency in the proposed protocol is lower than that in EEDS, especially when *Ks* is small. Fig.8 shows the relation of segment length with probability of successful query. We observe that the proposed protocol is more energy efficient and it can provide better performance in probability of successful query and query latency than EEDS which has only one access entry.

TABLE III. Simulation Parameters

Probability of sensor failure	0.02
MAC protocol	802.11
Routing protocol	GPSR
Number of objects	10
Average length of linked list	60 hops
Average length of query linked list	20 hops
Segment length *Ks*	5,10,15

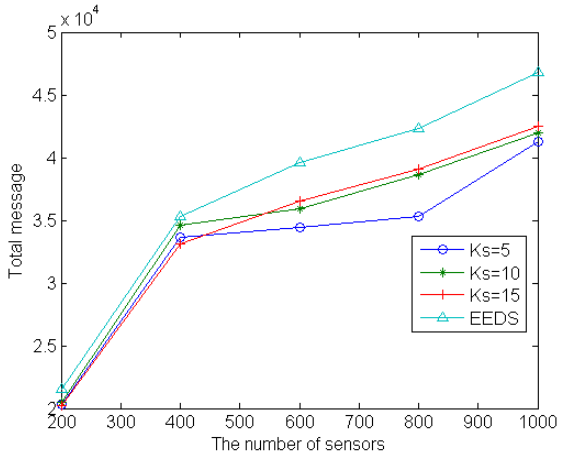

Figure 6. Comparison of total message

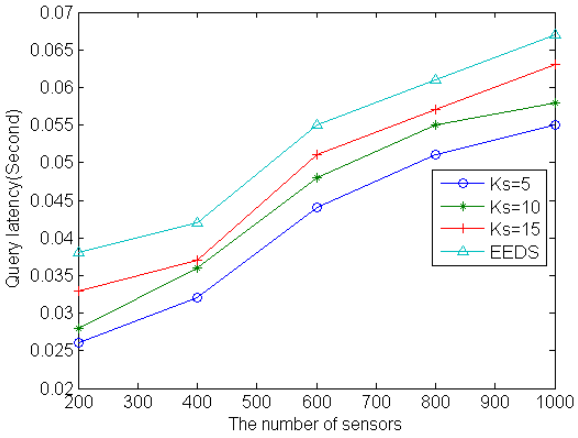

Figure 7. Query latency vs. number of sensors and segment length

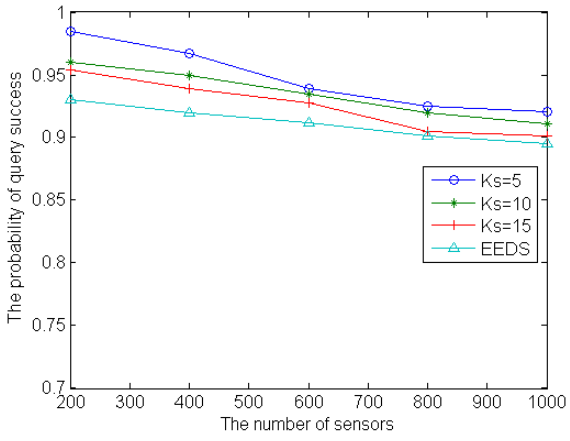

Figure 8. Probability of successful query vs. number of sensors and segment length

Ⅵ. Conclusion

This work proposes an object tracking protocol for sensor networks, which focuses on users how to query target tracks and obtain the target position effectively. The proposed protocol combines advantages of local storage and data centric storage methods to support query efficiently. Object's moving data will be stored in node near detecting sensor and the relation of storage nodes is maintained using multiple access entry linked list along the moving path of the object. Access entry

messages which are called index will be stored in a designated node. Users can obtain tracking information of object from a part of linked list according to query request. Performance analysis and simulation studies show that the proposed protocol is energy efficient and it supports multiple kinds of query in better performance in terms of probability of successful query and query latency than EEDS. In some applications, because of limited power and other external reasons, such as a group of sensor nodes being disabled or because of frequent sensors movement, some grids may become holes (void grids). How to efficiently manage tracking data in such circumstance is our future work.

ACKNOWLEDGMENT

This work is supported by the Hunan Provincial Natural Science Foundation of China for Distinguished Young Scholars under Grant No. 07JJ1010, the National Basic Research and Development Plan (973) of the Chinese Ministry of Science and Technology under Grant No. 2003CB317003, the Research Grants Council of the Hong Kong SAR, China No. (CityU 114908) and CityU Applied R & D Funding (ARD-(Ctr)) Nos. 9681001 and 9678002.

REFERENCES

[1] I. Akyildiz, W. Su, Y. Sankarasubramaniam, et al. "A Survey on Sensor Networks," *IEEE Communications Magazine*, 2002, 40 (8), pp. 102-114.

[2] W. P. Chen, J.C. Hou, and L. Sha. "Dynamic Clustering for Acoustic Object Tracking in Wireless Sensor Networks," *Proceeding of 11th IEEE International Conference on Network Protocols (ICNP'03)*, Atlanta, Georgia, USA, November 2003, pp. 284–294.

[3] C.-Y. Chong, F. Zhao, S. Mori, and S. Kumar. "Distributed Tracking in Wireless Ad Hoc Sensor Networks," *Proceedings of the Sixth International Conference on Information Fusion (FUSION 2003)*, Cairns, Australia, July 2003, pp. 431–438.

[4] F. Mondinelli and Z. M. Kovacs-Vajna. "Self-localizing Sensor Network Architectures," *IEEE Transactions on Instrumentation and Measurement*, 2004, 53 (2), pp. 277–283.

[5] H. Yang and B. Sikdor. "A Protocol for Tracking Mobile Objects Using Sensor Network, Sensor Network Protocols and Applications," *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, Anchorage, Alaska, May 2003, pp. 71–81.

[6] Y.-C. Tseng, S.-P. Kuo, H.-W. Lee, and C.-F. Huang. "Location Tracking in A Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies," *Computer Journal*, 2004, 47 (4), pp. 448–460.

[7] G. Wang, J. Cao, H. Wang, and M. Guo. "Polynomial Regression for Data Gathering in Environmental Monitoring Applications," *Proceedings of the 2007 IEEE Global Telecommunications Conference (GLOBECOM 2007)*, November 2007, Washington DC, USA, pp. 1307-1311.

[8] S. Goel and T. Imielinski. "Prediction-Based Monitoring in Sensor Networks: Taking Lessons from MPEG," ACM SIGCOMM Computer Communications Review, October 2001, Vol. 31, Issue 5, pp. 82-98.

[9] E. Olule, G. Wang, M. Guo, and M. Dong. "RARE: An Energy-Efficient Object Tracking Protocol for Wireless Sensor Networks," *Proceedings of the 2007 IEEE International Conference on Parallel Processing Workshops (ICPPW 2007)*, September 2007, Xi'an, China, pp. 76-81.

[10] G. Wang, T. Wang, W. Jia, M. Guo, and J. Li. "Adaptive Location Updates for Mobile Sinks in Wireless Sensor Networks," *The Journal of Supercomputing (Springer)*, 47(2): 127-145, February 2009.

[11] S. Goel and T. Imielinski. "Prediction-Based Monitoring in Sensor Networks: Taking Lessons from MPEG," *ACM SIGCOMM Computer Communication Review*, 2001, 31 (5), pp. 82-98.

[12] Y. Xu, J. Winter and W.-C. Lee. "Prediction-Based Strategies for Energy Saving in Object Tracking Sensor Networks," *Proceedings of the 2004 IEEE International Conference on Mobile Data Management (MDM'04)*, Berkeley, California, January 2004, pp. 346-357.

[13] H. Yang and B. Sikdor. "A Protocol for Tracking Mobile Objects Using Sensor Network," *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications*, Anchorage, Alaska, May 2003, pp. 71–81.

[14] Y. Xu, J. Winter, and W. Lee. "Dual Prediction-Based Reporting for Object Tracking Sensor Networks," *Proceedings of the First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous 2004)*, Boston, USA, 2004, pp. 154-163.

[15] Y. Xu and W.-C. Lee. "On Localized Prediction for Power Efficient Object Tracking in Sensor Networks," *Proceedings of 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW 2003)*, May 2003, pp. 434-439.

[16] S. Ratnasamy, B. Karp, Y. Li, F. Yu, R. Govindan, S. Shenker, and D. Estrin. "A Geographic Hash Table for Data-Centric Storage," *Proceedings of the First ACM International Workshop on Wireless Sensor Networks and Applications (WSNA 2002)*, 2002, pp. 78-87.

[17] E. Pitoura and I. Fudos. "Distributed Location Databases for Tracking Highly Mobile Objects," *The Computer Journal*, 2001, 44(2), pp. 75-91.

[18] W. Zhang and G. Cao. "DCTC: Dynamic Convoy Tree-Based Collaboration for Object Tracking in Sensor Networks," *IEEE Transactions on Wireless Communications*, 2004, 3(5), pp. 1689-1701.

[19] O. Nakov and D. Petrova. "Setting of Moving Object Location with Optimized Tree Structure," *Proceedings of the 11th WSEAS International Conference on Computers*, Agios Nikolaos, Crete Island, Greece, July 2007, pp. 67-71.

[20] J. Xu, X. Tang, and W-C. Lee. "EASE: An Energy-Efficient In-Network Storage Scheme for Object Tracking in Sensor Networks," *Proceedings of the Second Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON 2005)*, pp. 396-405.

[21] H.-W. Tsai, C.-P. Chu, and T.-S. Chen. "Mobile Object Tracking in Wireless Sensor Networks," *Computer Communications 30 (2007)*, pp. 1811-1825.

[22] Y. Xu and W.-C. Lee. "DTTC: Delay-Tolerant Trajectory Compression for Object Tracking Sensor Networks," Sensor Networks, Ubiquitous, and Trustworthy Computing, 2006. *IEEE International Conference on Volume 1, Issue*, June 2006, pp. 436-445.

[23] S. W. Ho and G. J. Yu. "An Energy Efficient Data Storage

Policy for Object Tracking Wireless Sensor Network," *Proceedings of the IEEE International Conference on Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC 2006)*, pp. 20-25.

[24] N. Lin, C. Chang, and C. Pan. "An Adaptive Deflect and Conquer Clustering algorithm," *Proceedings of the 7th WSEAS International Conference on Simulation, Modelling and Optimization*, Beijing, China, September 2007, pp. 156-160.

Jin Zheng was born in November 1970 in Bazhong, China. She received the BSc degree in applied mathematics from Jilin University and the MSc degree in computer science from Central South University, in 1993 and 2003 respectively.

Currently, she is an Associate Professor and a Ph.D. candidate in the Department of Computer Science and Technology, Central South University. Her research interests include computer networks, data management and software Engineering.

Weijia Jia received BSc, MSc, from Center South University, China and M. Applied Sci. and Ph.D. from Polytechnic Faculty of Mons, Belgium, all in computer science.

He is a Professor of Computer Science in the City University of Hong Kong (CityU). He joined German National Research Center for Information Science (GMD) in Bonn (St. Augustine) from 1993 to 1995 as post-doc. In Aug. 1995, he joined Department of Computer Science, CityU, as an assistant professor. In 2006, he was awarded HK$11 millions from the Innovation & Technology Fund of the HKSAR Government for a project entitled "Digital Network Platform Technology and Equipment for Ubiquitous Communications". The project is the No. 1 in size at CityU in 2006 with intention of design and implementation of ubiquitous communication platform and soft switch to harness Internet with 3G, WiFi,WiMAX, ad-hoc and PSTN networks.

His research interests include wireless communication and networks, distributed systems, multicast and (pioneer) anycast QoS routing protocols for Internet. In these fields, he has about 300 publications in international journals, books/chapters and refereed international conference proceedings. He (with Wanlei Zhou) has published a book "Distributed Network Systems" (Springer, 2005) where the book contains extensive research materials and implementation examples. He has received the best paper award in a prestige (IEEE) conference. He (with J. Chen et al.) has proposed an improved algorithm for well-known Vertex Cover and set-packing NP-hard problems with time bounds of O(kn + 1.2852k) and O((5.7k)kn) respectively. The both results stand on the current best time-bound (as at Oct. 2006) for the fixed-parameterized intractable problems.

Prof. Weijia has served as the editor and guest editor for international journals and PC chairs and PC members/keynote speakers for various IEEE international conferences. He is the member of IEEE and has been listed in Marquis Who's Who (VIP) in the World (2000-2008).

Guojun Wang was born in February 1970 in Changsha, China. He received B.Sc. in Geophysics, M.Sc. in Computer Science, and Ph.D. in Computer Science, from the Central South University, in 1992, 1996, 2002, respectively.

He is currently a Professor at Central South University, China. He is the Director of the Trusted Computing Institute, formerly the Mobile Computing Institute, in the University. He is also a Vice Head of the Department of Computer Science and Technology in the University. He is currently a Visiting Scholar at Florida Atlantic University in USA during 2009–2010. He was a Research Fellow at the Hong Kong Polytechnic University, Hong Kong, during 2003–2005, and a Visiting Researcher in the University of Aizu, Japan, during 2006–2007. His research interests include trusted computing, pervasive computing, mobile computing, and software engineering. He has published more than 140 technical papers and books/chapters in the above areas.

He is an associate editor or on the editorial board of some international journals including Security and Communication Networks, Journal of Computer Systems, Networking, and Communications, and International Journal of Multimedia and Ubiquitous Engineering. He has also served as guest editor-in-chief or guest editor for some international journals including IEICE Transactions on Information and Systems, The Journal of Supercomputing (Springer), Security and Communication Networks (Wiley), and Journal of Computers (Academy Publisher). He has also served as a general chair, program chair, program vice-chair, track chair, workshop chair, publication chair, publicity chair, and program committee member for more than 90 international conferences and workshops such as GLOBECOM, ICC, WCNC, AINA, ICPADS, HPCC, IWCMC, EUC, ISPA, ICYCS, CSA, ATC, TrustCom, TSP, ScalCom, and ISSR. He is a senior member of the China Computer Federation, the Academic Committee member of the YOCSEF of the China Computer Federation, the chair of the YOCSEF Changsha of the China Computer Federation (2007–2008), and a member of several Technical Committees of China Computer Federation, including Fault Tolerant Computing, Pervasive Computing, Software Engineering and E-Government, and Office Automation.