Boundary Node Selection and Target Detection in Wireless Sensor Network

Prasan Kumar Sahoo
Dept. of Information Management
Vanung University
Chungli, Taiwan, 320, R.O.C.
Email: pksahoo@msa.vnu.edu.tw

Kun-Ying Hsieh
Dept. of Computer Science
and Information Engineering
National Central University
Chungli, Taiwan, 320, R.O.C.
Email: rocky@axp1.csie.ncu.edu.tw

Jang-Ping Sheu
Dept. of Computer Science
and Information Engineering
National Central University
Chungli, Taiwan, 320, R.O.C.
Email: sheujp@csie.ncu.edu.tw

***Abstract*— Recording information of the entry and exit of a target through the boundary of the deployed region is highly essential in wireless sensor network. In this paper, sequential boundary node selection (SBNS) and distributed boundary node selection (DBNS) algorithms are proposed to find out the boundary nodes of the wireless sensor network with or without presence of obstacles over the deployed region. Besides, a target detection protocol is proposed to detect the entry and exit of the single target using those boundary nodes. Simulation results show that the selection of boundary nodes in our protocol is almost close to the optimal one and time of selecting the boundary nodes would not increase rapidly with increase in size of the wireless sensor network.**

Key words: Boundary nodes, target detection, wireless sensor network.

I. Introduction

Wireless sensor network (WSN) is important for a number of strategic applications such as coordinated target detection, surveillance, and localization. Several works on target detection and tracking [2] are found in recent years, which can be classified into four different categories. The first category is to find out the trajectory of the target. In [1, 6], authors focus on finding trajectory of the target via the detected data. They use the time of entering or leaving of a target through the sensing range of the sensors to draw trajectory of the interesting target. The second category, as described in [8] is to wake up the sensors by using predictive strategy in order to keep track with the target, when it moves into their sensing ranges. The third category is to use the predictive strategy to reduce the transmitted data between the sink and each sensor node. The last category is to obtain more accurate information of the target. The authors in [5] have proposed a tree based structure that uses a lot of sensors to collaborate the detection mechanism and to collect precise data. However, above methods always waste the resources, if a target moves to certain area iteratively.

It is to be noticed that some applications may only need to record the information of a target entering or leaving the boundary of the specific regions. For example, zoologists want to know the wildlife migration or habitual behavior such as duration of a target that stays in the monitoring regions or when it enters into or exits from the region. Hence, in this paper, we propose the boundary node selection algorithms, in which we select limited number of nodes out of all the deployed nodes along the border area of the monitoring region. Besides, we propose a target detection protocol that uses those selected boundary nodes to monitor the entry and exit of the target in order to keep surveillance of the monitoring region.

The remainder of this paper is organized as follows. Section II presents our system model such as the research environment and assumptions. Two boundary node selection protocols are proposed in Section III and the target detection protocol is described in Section IV. Evaluation of our algorithms and simulation results are presented in Section V, and concluding remarks are made in Section VI of the paper.

II. System Model

In our work, it is assumed that all sensors are randomly and densely deployed over the monitoring region such that no isolated node exists in the network. The communication range of the sensors is fixed and the monitoring region is fully sensing covered. The sensing range of a node is variable, which may be larger or smaller than its communication range. Each node has a unique ID for its identification and knows its location information via GPS or through some positioning methods [7]. In order to avoid the ambiguity, we differentiate between the *border nodes* and *boundary nodes*. In our work, nodes present along close proximity of the border of the deployed region are termed as the *border nodes* and few selected nodes out of those *border nodes*, based on our algorithms are termed as the *boundary nodes*. These boundary nodes enclose three kinds of regions: the *monitoring region*, *sensing holes*, and *user-defined regions*. The sensing holes are formed due to some obstacles within the monitoring region such as ponds or small hills and user-defined region means that users can define certain area within monitoring region by themselves, as per the required surveillance. The boundary is generated by the selected *boundary nodes (BNs)* and are linked together to form a loop and responsible for detecting the entry or exit of the target to or from the monitoring region.

III. Boundary Node Selection Protocol

In this section we propose the sequential (SBNS) and distributed (DBNS) algorithms to select the boundary nodes out of all deployed nodes along the border of the network. It is to be noted that our SBNS algorithm can find boundary nodes along the border area of the monitoring region, and it cannot find boundary nodes around the sensing holes and user-defined regions. Hence, we propose the DBNS algorithm to find out boundary nodes along the border of the monitoring region.

A. Sequential Boundary Node Selection (SBNS) Algorithm

Initially, we assume that the sink node is assigned as a starting node in the border area of the monitoring region, which will be the first node to execute the SBNS algorithm. Hence, the sink sets itself as a *BN*, and broadcasts the *Request_Info* packet to its one-hop neighbors, which contains the location information, sensing range and unique ID of the sink and waits for the response. Upon receiving that packet, each of its one-hop neighbors responds with a *Reply_Info* packet, which includes their location information, sensing range and unique ID. Upon receiving the neighbor's information, the sink compares its location with location of its one-hop neighbors. Since, sink is present along the border area of the monitoring region, it has either maximum or minimum value in its x or y coordinates. We assume that each node has a turning line to find a *BN* among its neighbors. The turning line is assumed to be horizontal on the right or left side of the sink, if it has maximum (x_0=X_{max}) or minimum (x_0=X_{min}) value in its *x*-coordinate, respectively or vertical on its top or bottom side of the sink, if it has maximum (y_0=Y_{max}) or minimum (y_0=Y_{min}) value in its *y*-coordinate, respectively.

The sink uses the right-hand rule [3] and rotates the turning line along clockwise direction. The first node whose sensing range intersects with the rotating turning line is selected as the next *BN*. The selected *BN* is included in the *Ack_Info* packet and is broadcast by the sink. Besides, each *BN* that wants to select a new *BN* assumes a turning line through the line connecting to the previous *BN* and itself. This procedure is repeated until the starting node is revisited. Thus, the nodes those are initially selected as *BNs* change their role to *Non-BNs*, thereby reducing the number of *BNs*. To maintain the integrity among the *BNs*, each *BN* periodically sends a beacon packet to its previous *BN*. If any *BN* cannot receive the beacon packet from its related *BNs*, it executes the SBNS algorithm to select a new *BN* among the existing neighbors.

B. Distributed Boundary Nodes Selection (DBNS) Algorithm

Normally, the DBNS algorithm is used to select the *BNs* along the border of the sensing holes. It has three phases as described below.

1) Initial Phase: In this phase, the sink broadcasts a *BN_Start* packet to its one hop neighbors with its location information, sensing range and unique ID. Upon receiving this packet, each node includes the same information and rebroadcasts it to their one-hop neighbors. Then, each node determines whether it is an *extreme node* or not based on its maximum or minimum value in its *x* or *y* or both coordinates as compared to its neighboring nodes. Those extreme nodes declare themselves as *BNs*. Thus, each sensor node in the monitoring region could be classified as *BNs* or *Non-BNs* after the *initial phase* is executed.

2) Selection Phase: In this phase, each *BN* collaborates with its neighbors to check the presence of a *BN* on its left, another *BN* on its right side and selects some more *BNs* out of the *Non-BNs*. To achieve this purpose, each *BN* broadcasts a *BN_Msg* packet to its one-hop neighbors. However, there is possibility that the *Non-BNs* might have received zero to multiple number of *BN_Msg* packets from the *BNs*. Accordingly, we classify our protocol into several cases, as discussed below.

Case 1: If a *Non-BN* cannot receive any *BN_Msg* packet from the *BNs* within certain predefined time, the *Non-BN* goes to power saving mode. This type of situation may happen for the nodes located within the central region of the monitoring area.

Case 2: If a *Non-BN* receives *BN_Msg* packets from two different *BNs*, the *Non-BN* chooses itself either to be a new *BN* or still remains as a *Non-BN*, or becomes a forwarding node between those two *BNs*. If those two *BNs* can communicate and their sensing range overlaps with each other, the *Non-BN* will not change it's role.

If two *BNs* cannot communicate and their sensing range overlaps with each other, the *Non-BN* directly sets itself as a new *BN*. It is to be noted that if more than one *Non-BN* satisfy this condition, those *Non-BNs* will exchange their location information with their one-hop neighbors and the *Non-BN* having shortest vertical distance between its position and the line connecting to both *BNs* will set itself as the new *BN*. If those two *BNs* can communicate with each other without overlapping their sensing range, and that *Non-BN*'s sensing range overlaps with either of those two *BNs*, then that *Non-BN* sets itself as a new *BN*. If sensing range of those two *BNs* overlaps, but they cannot communicate with each other, the *Non-BN* having shortest vertical distance from the line joining those two *BNs*, becomes the forwarding node between them.

Case 3: If a *Non-BN* receives more than two *BN_Msg* packets from the *BNs*, the *Non-BN* considers each pair of its nearby *BNs* to decide its own role.

Case 4: If any of the *Non-BNs* receives the *BN_Msg* packet from one of the *BNs*, the *Non-BN* responds to it with a *Non_BN_ID* packet that contains its *ID*, sensing range and location information. Upon receiving one to multiple *Non_BN_ID* packets from the *Non-BNs*, the *BN* checks the current number of *BNs* with whom it's sensing range overlaps. If the *BN* has already two *BNs* in its left and right hand side, whose sensing range overlaps with it and both are connected to it, then it ignores that *Non_BN_ID* packet and the corresponding sender. If the *BN* finds only one *BN* with whom it's sensing range overlaps, it has to find out another one *BN*, whose sensing range overlaps with it in its left or right side. Hence, it selects one *Non-BN* sender as a new *BN* out of those *Non-BN* senders.

It is to be noted that if the *BN* has maximum or minimum value in its *x* or *y*-coordinate, the selected *Non-BN* along each of its side must have sensing overlapping with that *BN* and maximum or minimum value in the *x* or *y*-coordinate among the *Non-BNs*. If the *BN* has no sensing range overlapping with other *BNs*, the *BN* selects a neighboring *Non-BN* as the new *BNs* from its right side and another one from its left side.

3) Pruning Phase: This phase is meant to reset the redundant *BNs* to *Non-BNs*, if two *BNs'* sensing ranges in their right and left side overlap with another same *BNs*. This implies that only one *BN* is required to have sensing overlap with these two *BNs*. Therefore, we propose that each redundant *BN* should first set itself as a *Mark_BN* before returning to a *Non-BN* and broadcasts *Mark_BN ID* packets to all of its neighboring *BNs* to check if it can return to a *Non-BN*. The *Mark_BN ID* packet contains a *Mark_BN*'s *ID*. For example, as shown in Fig. 1, the *BN B* sets itself as a *Mark_BN* and broadcasts the *Mark_BN*(*B*) packet to the neighboring *BNs A* and *C*. Once the *BN A* and *C* receive the *Mark_BN*(*B*) packet, each of them checks if it is a *Mark_BN* or not. The *BN C* is also a *Mark_BN*, but its sensing range is larger than the *Mark_BN B*. So, the *BN C* sends the *Go_NonBN*(*C*) packet back to the *Mark_BN B* to allow *Mark_BN B* to return as a *Non-BN* node. Since *BN A* is not a *Mark_BN*, it directly sends the *Go_NonBN*(*A*) packet back to the *Mark_BN B*. When the *Mark_BN B* receives the *Go_NonBN*(*ID*) packets from all of its neighboring *BN A* and *C*, the *Mark_BN B* returns to the *Non-BN* and broadcasts a *Re_NonBN*(*B*) packet to declare its return as the *Non-BN*. Eventually, our *DBNS* algorithm can form a complete boundary, dynamically.

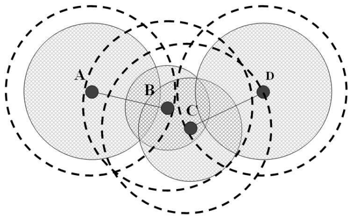

Fig. 1. Pruning the redundant *BN B*

IV. Target Detection Protocol

In this section, we propose the target detection protocol, which detects the entry or exit time of a target over the monitoring region by using the selected boundary nodes, as described in Section III. As shown in Fig. 3, let the bold curved line represents the outer boundary of all *BN*'s sensing range. As soon as the target passes through the outer boundary, the time of entry or exit, as detected by the *BN* is transmitted to the sink. Accordingly, there are two kinds of time stamps. One is the target entering to the network, called T_e; and the other one is the target leaving the network, called T_l.

A. Sensing Overlapping Algorithm

According to the previous assumptions, the monitoring region is fully sensing covered. That is, the circumference of each *BN*'s sensing range within the monitoring region must be covered by other neighboring *BNs*' and *Non-BNs*' sensing range. For example, as shown in Fig. 2, the *BN A*'s circumference is covered by two *BNs*' (the dark gray curves) and one *Non-BN*'s (the light gray curves) sensing range. Therefore, we always can find the nearby *Non-BNs* of each *BN*. Besides, the *BN* has sensing overlapping with the *Non-BN* does not mean that it can communicate with the *Non-BNs*. Hence, we propose the sensing overlapping algorithm to find those *Non-BNs* and ensure that the *BN* can communicate with them. First, the *BN A* broadcasts a *Find_Overlap* packet containing its location information and sensing range S_A to the neighbors, as shown in Fig. 2. The *Non-BNs D*, *E* and *F*, whose distance from the *BN A* is less than or equal to (S_A + S_{MAX}) can ensure that its sensing range overlaps with the *BN A*, where S_{MAX} means the maximum sensing range among all the deployed nodes.

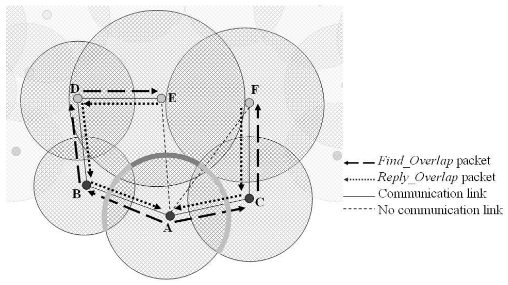

Fig. 2. An example of finding sensing overlap

Upon receiving the *Find_Overlap* packet, only *Non-BNs E* and *F*, who satisfy the above conditions rebroadcast the packet and transmit the *Reply_Overlap* packet to the *BN A* with its location and sensing range. The *BN A* selects the closest *Non-BN E* and repeats the procedure until its circumference of sensing range within the monitoring region is fully sensing covered. If the distance between the *BN* and some *Non-BNs* are same, the *Non-BN* having largest sensing range is selected. Eventually, the *BN A* will select *Non-BN E* to exchange the time stamp information when the target is detected through the routing path *A*, *B*, *D* and *E*.

B. Target Detection Algorithm

This algorithm is meant to collaborate the *BNs* and *Non-BNs*, whose sensing range overlap with each other and to determine the entry or exit of the target through the monitoring region. When a target enters to the monitoring region, the *BNs* and the *Non-BNs*, having sensing overlapping with the *BNs*, can detect the target and then broadcasts the *Detect*(*ID*) packet to their neighbors. Similarly, when they detect the exit of the target, they broadcast the *Leave*(*ID*) packet to their neighbors. Each sensor node maintains a table to record the received *Detect*(*ID*) or *Leave*(*ID*) packets. If a sensor node *X* receives the *Detect*(*Y*) and *Leave*(*Y*) packets from the same node *Y*, it removes the data of those two packets from its recoding table as the target is no longer within the sensor node *Y*'s sensing range. If the *BN* detects that the target is entering to the monitoring region, it transmits the *Entering_Time*(T_e, *ID*) packet to the sink.

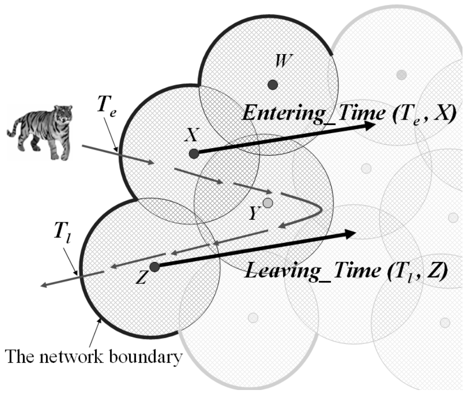

Fig. 3. Illustration of the target detection algorithm

Similarly, if *BN* detects the target leaving the monitoring region, it will send *Leaving_Time*(T_l, *ID*) packet to the sink. Since, each sensor node has the location information of the sink, it can utilize some geographic routing protocols [4] to transmit the *Entering_Time* and *Leaving_Time* packets to the sink. As shown in Fig. 3, when the tiger enters into the monitoring region, the *BN X,* first detects the target at time T_e, and it broadcasts the *Detect*(*X*) message to its neighbors. Besides, it checks and finds its recording table is empty, and then sends the *Entering_Time*(T_e, *X*) to the sink. After the target leaves the *BN X*'s sensing range, it broadcasts the *Leave*(*X*) packet and checks its recording table again. The time during *BN X* broadcasts the *Leave*(*X*) packet to its neighbors; *Non-BN Y* has already sent the *Detect*(*Y*) packet to the *BN X*. So, *BN X* finds a non-empty field in its recording table and therefore does not transmit the *Leaving_Time*(T_l, *X*) to the sink. Later, when the target turns back and leaves the monitoring region, *BN Z* receives the *Detect*(*Y*) packet from the *Non-BN Y,* and finds its recording table is non-empty. Hence, when the *BN Z* detects the target, it does not transmit the *Entering_Time*(T_e, *Z*) to the sink. Prior to leaving the *BN Z*'s sensing range, the target must have received the *Leave*(*Y*) packet from the *Non-BN Y*. Since, the *Detect*(*Y*) packet and *Leave*(*Y*) packet coexist in the *BN Z*'s recording table, it removes them form its recording table. After the target leaves *BN Z*'s sensing range, it broadcasts the *Leave*(*Z*) packet to its neighbors, and finds its recording table empty. Therefore, the *BN Z* transmits the *Leaving_Time*(T_l, *Z*) to the sink, as it is the last sensor that detects the target leaving.

V. Performance Analysis

To evaluate the performance of the proposed boundary node selection algorithms, the simulation is implemented in JAVA programming language. Nodes are randomly deployed over a monitoring region of different area, such as 100m×100m, 150m×150m, 200m×200m, and 250m×250m. The number of deployed nodes varies from 1000 to 2500 nodes. Communication range for every sensor nodes is fixed at 8m, though the sensing range of each node varies from 4m to 12m. Then, our DBNS and SBNS algorithms are implemented for different node numbers, area of the deployed region and compared with the centralized algorithm, considering the control packets overhead of our algorithms. In the centralized algorithm, we assume that the sink has the locations of all nodes in the network. It can find the nodes on the border area of the monitoring region and chooses the nodes with larger sensing range as the *BNs*. The performance metrics such as the number of *BNs* is defined as the number of selected *BNs* to enclose the monitoring region, the control packets overhead is defined as the number of control packets to find and select the *BNs* and the *BN* selection time is defined as the total time of finding the entire *BNs*.

Fig. 4 represents a simulated final constructed boundary using DBNS algorithm, when 2500 nodes are deployed randomly over the monitoring region of size 100m×100m. The gray dot represents the finally selected boundary nodes and the black circle represents their corresponding sensing range. Fig. 5 shows the possible number of boundary nodes those can be selected by the centralized, SBNS and DBNS algorithms. Here, the monitoring region is assumed to be 100m×100m, and the communication range is fixed at 8m. The simulation result demonstrates that when we increase the number of nodes in the network, the number of selected *BNs* is decreased. This is because increasing the number of nodes will increase the probability of selecting the nodes with higher sensing range and thereby decreasing the number of *BNs*. Therefore, the monitoring region can be enclosed by fewer *BNs* with larger sensing ranges.

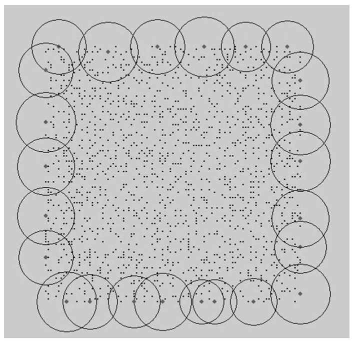

Fig. 4. Simulated boundary nodes using DBNS algorithm

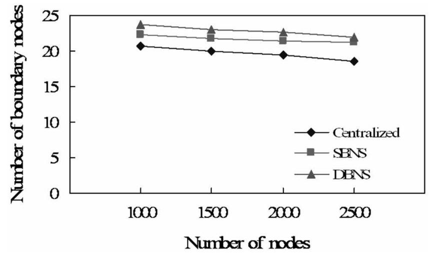

Fig. 5. Number of *BNs* using SBNS, DBNS, and centralized algorithm

Comparing the SBNS and DBNS algorithms, it is found that SBNS has better performance than DBNS, as the SBNS usually selects the *BNs* with larger sensing range. Contrary to this, in DBNS, initial phase selects the *BNs* according to the node's location information without taking node's sensing range into account. However, the DBNS also considers the sensing range in selection and pruning phases. Therefore, these two algorithms have similar performance in selecting *BNs*. The simulation also shows that the number of selected *BNs* is very close to the optimal value, if DBNS is used.

Although the centralized algorithm has fewest number of *BNs*, its control packet overhead are quite high, as each node in centralized algorithm has to transmit its own information to the centralized node such as the sink. This needs to transmit lots of control packets to the sink. Obviously, as shown in Fig. 6, our two algorithms have fewer control packets overhead than the centralized one. The control packets overhead in our DBNS algorithm is due to the flooding of start message through the entire monitoring region by the sink in the initial phase, exchange of *Non-BNs* and *BNs* local information in the selection phase, and the exchange of redundant *BNs* and its nearby *BN*s messages in the pruning phase. DBNS has the lowest control packets overhead in selecting the *BNs*. Because, in SBNS, the sink needs to flood the Starting packet to notify the starting node, and each selecting procedure need to exchange the information among each *BNs* and its neighbors. However, in DBNS, not all of *BNs* need to exchange information with neighbors. Therefore, the control packet overhead of DBNS is lower than the SBNS and Optimal algorithm. For this, we simulate different numbers of nodes from 1000 to 2500 and compare the DBNS with SBNS. It is found that the control packet overhead of SBNS is more than the DBNS, if node numbers are increased. Hence, the percentage of average control packets overhead of SBNS is more than 39.89% of DBNS and with the centralized algorithm is 3.29 times more than the DBNS algorithm.

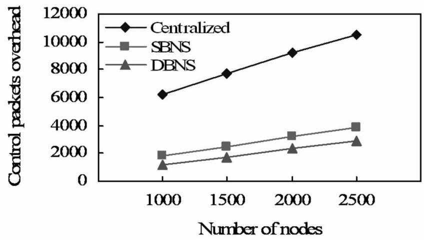

Fig. 6. The control packets overhead to select the *BNs*

We simulated the transmission time by deploying the sensor nodes over the monitoring regions of size 100m×100m, 150m×150m, 200m×200m, and 250m×250m with density of 0.25 nodes/m^2 and the result is shown in Fig. 7. The time unit (t) represents the transmission time plus the calculation time. It is observed that the calculation time is much less then the transmission time and we can ignore it. In SBNS, it sequentially selects the *BNs* one by one and only one selected *BN* can select another one. However, in DBNS, the selected *BNs* can select other *BNs* simultaneously. Therefore, DBNS

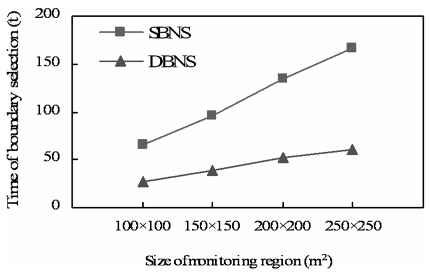

Fig. 7. Boundary selection time in various sizes of regions

algorithm takes less time to find the boundary than the SBNS. Besides, the size of the monitoring region does not affect a lot on the selection time of DBNS algorithm. Our DBNS is more time efficient than the SBNS for a large-scale monitoring region.

VI. CONCLUSION

In this paper, a sequential boundary node selection algorithm (SBNS) and distributed boundary node selection algorithm (DBNS) for the WSN are proposed to find the boundary nodes of a monitoring region. Besides, a target detection protocol is proposed to know the entry and exit of single target through the monitoring region using those boundary nodes. The simulation results show that the DBNS algorithm has similar performance with the SBNS, in selecting the boundary nodes. However, the communication overhead of DBNS is lower than SBNS due to the information exchange among fewer neighboring nodes in DBNS. In addition, with increase in network size, the boundary node selection procedure in DBNS is much faster than the SBNS.

REFERENCES

[1] J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a Moving Object with a Binary Sensor Network", in *Proc. 1st International Conference on Embedded Networked Sensor Systems*, pp. 150-161, Los Angeles, California, USA, Nov. 2003.

[2] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui and B. Krogh, "VigilNet: An Integrated Sensor Network System for Energy-efficient Surveillance", *ACM Trans. on Sensor Networks*, Vol. 2, Issue 1, pp. 1-38, Feb. 2006.

[3] B. Karp, and H. T. Kung, "GPSR:Greedy Perimeter Stateless Routing for Wireless Networks", in *Proc. 6th Annual International Conference on Mobile Computing and Networking* , pp. 243-254, Boston, Massachusetts, USA, Aug. 2000.

[4] W. H. Liao, Y. C. Tseng, and J. P. Sheu, "GRID: A Fully Location-Aware Routing Protocol for Mobile Ad Hoc Networks", *Journal of Telecommunication Systems*, Vol. 18, No. 1, pp. 37-60, Sep. 2001.

[5] C. Y. Lin, W. C. Peng and Y. C. Tseng, "Efficient in-Network Moving Object Tracking in Wireless Sensor Networks", *IEEE Trans. on Mobile Computing*, Vol. 5, Issue 8, pp. 1044-1056, Aug. 2006.

[6] K. Mechitov, S. Sundresh, Y. Kwon, and G. Agha, "Cooperative Tracking with Binary-Detection Sensor Networks", in *Proc. 1st International Conference on Embedded Networked Sensor Systems*, pp. 332-333, Nov. 2003.

[7] J. P. Sheu, C. S. Hsu, and J. M. Li, "A Distributed Location Estimating Algorithm for Wireless Sensor Networks", *Wireless Networks*, Vol. 1, pp. 218-225, Jun. 2006.

[8] S. P. M. Tran and T. A. Yang, "A Novel Target Movement Model and Energy Efficient Target Tracking in Sensor Networks", in *Proc. 37th SIGCSE Technical Symposium on Computer Science Education*, Vol. 38, Issue 1, pp. 97-101, Mar. 2006.