International Journal on Emerging Technologies (Special Issue on NCRIET-2015) **6**(2): 90-97(2015)

ISSN No. (Print) : 0975-8364
ISSN No. (Online) : 2249-3255

Efficient In-Network Moving Object Tracking in Wireless Sensor Network

D. Ramesh* and G S Biradar**
**Research Scholar, Department of Electronics and Communication Engineering, VTU, INDIA,*
***Professor, Department of Electronics and Communication Engineering, PDAEC, Kalburagi, INDIA,*

(Corresponding author: D. Ramesh)

(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: The main aim of this paper is to implement "An Efficient In-network Moving Object Tracking in wireless sensor network". In wireless sensor networks sampling time interval and the number of nodes involved in each stage of tracking are important factors which have high effect on the efficiency of target tracking applications. In this paper, a new phase-based adaptive target tracking method has also been proposed which at each time step employs two helpful tools. First, an extended Kalman filter (EKF)-based estimation technique to predict the tracking error and second, an energy consumption model to estimate energy consumption based on different number of nodes and sampling time intervals. By using these estimations, this method selects the best number of nodes and sampling time interval according to an objective function which is defined based on tracking accuracy and energy consumption.

Keywords: WSN, Target Tracking, Kalman Filter.

I. INTRODUCTION

Very past progress of wireless communication and embedded micro-sensing MEMS technologies has made *wireless sensor networks* possible. In light of storage in sensors, a sensor network can be considered as a distributed database, in which one can conduct *in-network* data processing. An important issue of wireless sensor networks is object tracking, which typically involves two basic operations:-a) Update b) Query This issue has been intensively studied in other areas, such as cellular networks. However, the in-network processing characteristic of sensor networks has posed new challenges to this issue. Here we develop several tree structures for in-network object tracking which take the physical topology of the sensor network into consideration. The optimization process has two stages. The first stage tries to reduce the location update cost based on a deviation-avoidance principle and a highest-weight first principle. The second stage further adjusts the tree obtained in the first stage to reduce the query cost. The way we model this problem allows us to analytically formulate the cost of object tracking given the update and query rates of objects Extensive simulations are conducted, which show a significant improvement over existing solutions.

To detect a target and trac its motion using wireless sensor network with the help of phase based optical flow algorithm. Object tracking is an important application of wireless sensor networks (e.g., military intrusion detection and habitat monitoring). Existing research efforts on object tracking can be categorized in two ways. In the first category, the problem of accurately estimating the location of an object is addressed. In the second category, in-network data processing and data aggregation for object tracking are discussed. The main theme of this paper is to propose a data aggregation model for object tracking. Object tracking typically involves two basic operations: *update* and *query*.

In general, updates of an object's location are initiated when the object moves from one sensor to another. A query is invoked each time when there is a need to find the location of an interested object. Location updates and queries may be done in various ways. A naive way for delivering a query is to flood the whole network. The sensor whose sensing range contains the queried object will reply to the query. Clearly, this approach is inefficient because a considerable amount of energy will be consumed when the network scale is large or when the query rate is high. Alternatively, if all location information is stored at a specific sensor (e.g., the sink), no flooding is needed.

But whenever a movement is detected, update messages have to be sent. One drawback is that when objects move frequently, abundant update messages will be generated. The cost is not justified when the query rate is low. Clearly, these are tradeoffs.

In, a *Drain-And-Balance* (DAB) tree structure is proposed to address this issue. As far as we know, this is the first in-network object tracking approach in sensor networks where query messages are not required to be flooded and update messages are not always transmitted to the sink. However as two drawbacks. First, a DAB tree is a logical tree not reflecting the physical structure of the sensor network; hence, an edge may consist of multiple communication hops and a high communication cost may be incurred. Second, the construction of the DAB tree does not take the query cost into consideration. Therefore, the result may not be efficient in some cases. To relieve the aforementioned problems, we propose a new tree structure for in-network object tracking in a sensor network. In particular, we take the physical topology of the sensor network into consideration. We take a two-stage approach. The first stage aims at reducing the update cost, while the second stage aims at further reducing the query cost. For the first stage, several principles, namely deviation-avoidance and highest-weight-first ones, are pointed out to construct an object tracking tree to reduce the communication cost of location update. Two solutions are proposed: Deviation-*Avoidance Tree* (DAT) and *Zone-based Deviation-Avoidance Tree* (Z-DAT). The latter approach tries to divide the sensing area into square-like zones, and recursively combine these zones into a tree. Our simulation results indicate that the Z-DAT approach is very suitable for regularly deployed sensor networks.

II. SIMULATION RESULTS

We have simulated a sensing field of size 256x256. Unless otherwise stated, 4096 sensors are deployed in the sensing field. Two deployment models are considered. In the first one, sensors are regularly deployed as a 64x64 grid-like network. In the second model, sensors are randomly deployed. In both models, the sink may be located near the center of the network or one corner of the network. Event rates are generated based on a model similar to the city mobility model . Assuming the sensing field as a square of size r X r, the model divides the field into 2X2 sub-squares called level-1 sub regions. Each level-1 sub region is further divided into 2 X 2 sub-squares called level-2 sub regions. This process is repeated recursively. Given an object located in any position in the sensing field, it has a probability p1 to leave its current level-1 sub region, and a probability 1- p1 to stay. In the former case, the object will move either horizontally or vertically with a distance of r/2. In the latter case, the object has a probability p2 to leave its current level-2 sub-region, and a probability 1 - p2 to stay. Again, in the former case, the object will move either horizontally or vertically with a distance of $r/2^2$, and in the latter case it may cross level-3 sub-regions. The process repeats recursively. The probability pi is determined by an exponential probability $pi = e^{-C.\ 2d\text{-}i}$, where C is a positive constant and d is the total number of levels.

In fact, the above behavior only formulates how objects move in the sensing field. After sensors are deployed in the network (no matter the sensors are deployed in a regular or random way), the movement patterns of these objects will generate event rates between neighboring sensors.

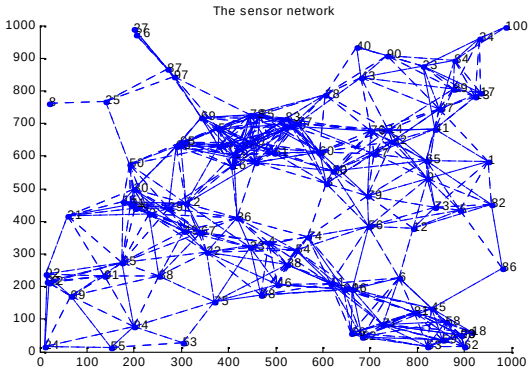

Fig. 1. The tree structure in wireless sensor network when the sensors are randomly deployed.

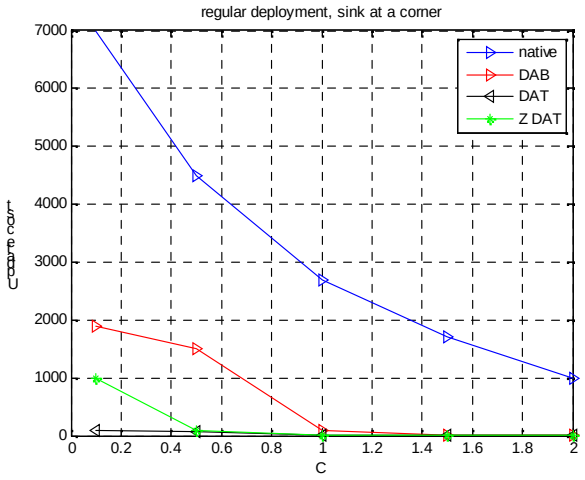

Fig 2.a Comparison of update costs when the sensors are regularly deployed and the sink at corner

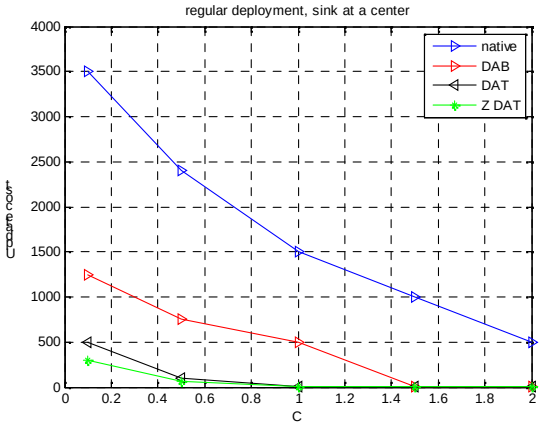

Fig 2.b Comparison of update costs when the sensors are regularly deployed and the sink at center.

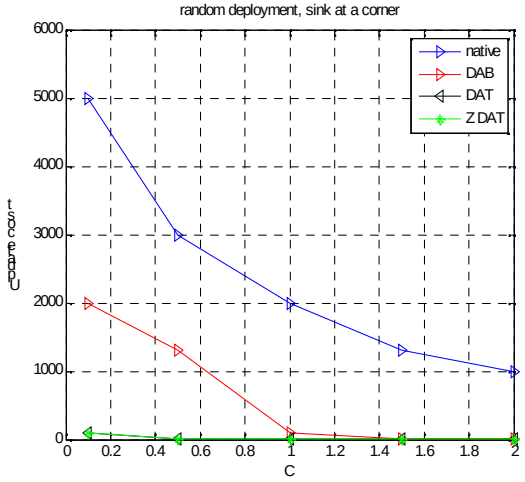

Fig 2.c Comparison of update costs when the sensors are randomly deployed and the sink at corner.

Also, objects are queried by the sink with the same probability. Since objects may be located at different sensors with different probabilities, the query rates may vary in different sensors. We compare our schemes with a naive scheme and the DAB scheme. In the naive scheme, any update is sent to the sink (i.e., there is no in-network processing capability.) In this case, the query cost is always zero, so it is preferable when the query rates are relatively high. For the DAB scheme, all sensors are considered leaf nodes, and a logical structure is used to connect these leaf nodes. When two sub-trees are merged into one, the root of the sub-tree which is closer to the sink will become the root of the merged tree (note that this may still cause deviation). First, we observe the advantage of using in-network processing to reduce update cost.

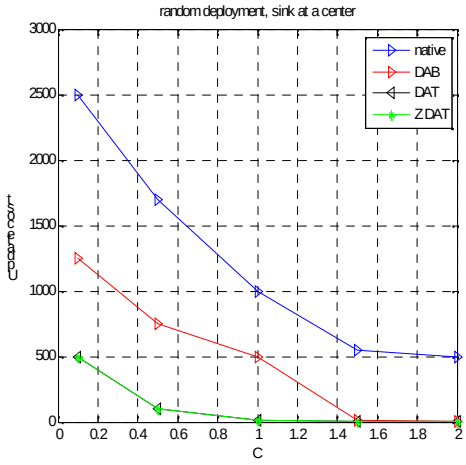

Fig. 2.d Comparison of update costs when the sensors are randomly deployed and the sink at center.

Figure 2.(a, b, c, d) below shows the result under different values of C for regular and random sensor deployment. As can be seen, a larger C implies a higher moving locality, thus leading to a lower update cost. The naïve scheme has the highest update cost, which is reasonable. By exploiting the concept of deviation avoidance and taking the physical topology into account, DAT and Z-DAT further outperform DAB. Next, we investigate the effect of deployment models. By comparing, the graphs in Fig. 2(a, b, c, d), we see that Z-DAT outperforms DAT under regular deployment, but the advantage is almost negligible under random deployment. This is because maintaining the shapes of sub-trees in Z-DAT is difficult. For example, Fig. shows snapshots of DAT trees and Z-DAT trees under regular and random deployments. As can be seen, Z-DAT does exploit the locality of sensors by partitioning sensors into zones under regular deployment. However, this is not true for the random case.

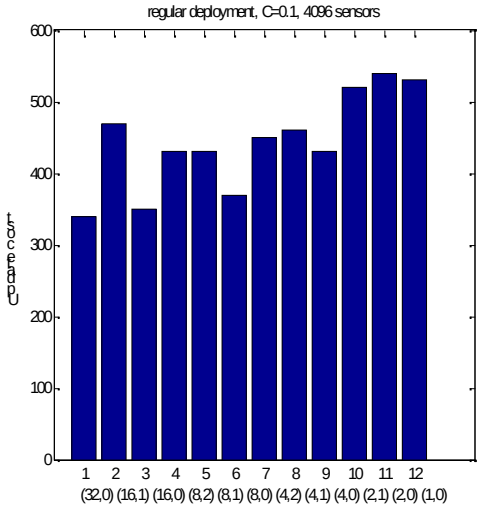

Fig. 3.a Comparison of update costs under different (α, δ) for for Z-DAT when regularly deployed at C= 0.1.

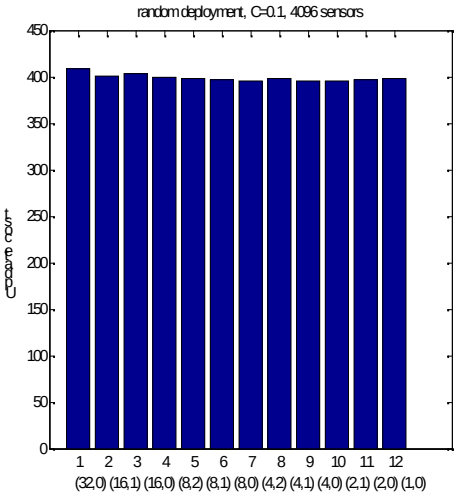

Fig. 3.b Comparison of update costs under different (α, δ) for for Z-DAT when randomly deployed at C= 0.1.

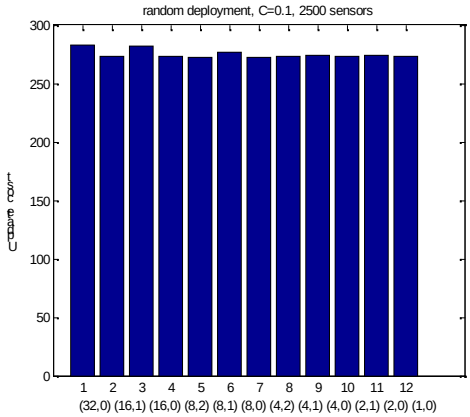

Fig. 3.c Comparison of update costs under different (α, δ) for for Z-DAT when randomly deployed at C= 0.1.

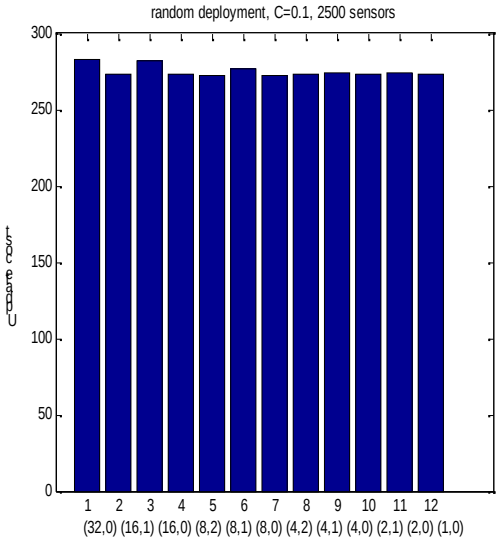

Fig. 3.d Comparison of update costs under different (α, δ) for for Z-DAT when randomly deployed at C= 0.1.

To get further insight into the performance of Z-DAT, we vary α and δ, and show the results in Fig., 3. (a, b, c, d)where a 4096- and a 2500-node sensor networks are simulated. Note that when = 1 and $\delta = 0$, Z-DAT is equivalent to DAT. For regular deployment, Z-DAT performs well when **'α'** is larger than 4. However, for random deployment, the Z-DAT does not perform well, because maintaining the shapes of sub-trees in Z-DAT is difficult. Furthermore, it can be seen that when δ=0, Z-DAT has better performance. This means that a square-like zone is better than a rectangle like zone. Also, note that the trend in both 4096- and 2500-node sensors networks (the latter has a non-power-of-2 number of nodes) are quite similar.

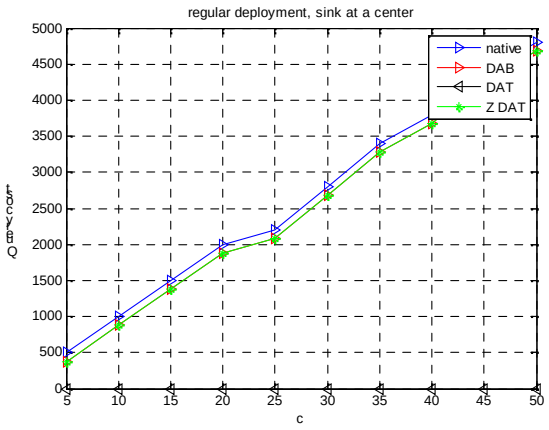

Fig. 4.a Comparison of query costs when sensors are regularly deployed with the sink at center.

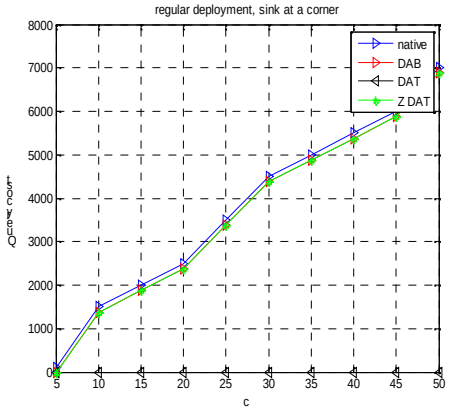

Fig. 4.b Comparison of query costs when sensors are regularly deployed with the sink at corner.

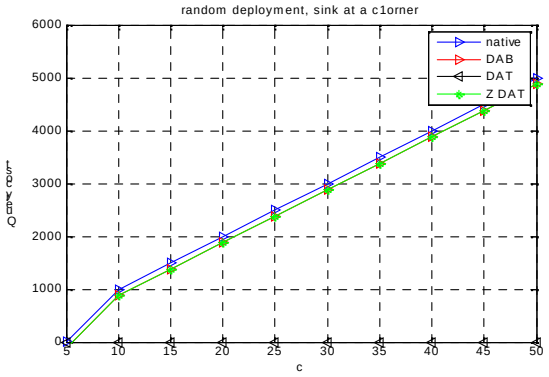

Fig. 4.c Comparison of query costs when sensors are randomly deployed with the sink at corner.

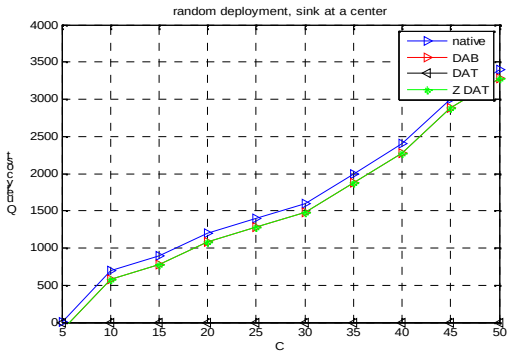

Fig. 4.d Comparison of query costs when sensors are randomly deployed with the sink at center.

Next, we examine the query cost. The result is shown in Fig. 4(a, b, c, d). In general, the query cost increases linearly with the aggregate query rate. As mentioned earlier, the query cost of the naïve scheme is always zero. Both query costs for DAT and Z-DAT are lower than that of DAB. This is attributed to the fact that query messages are always transmitted along the shortest paths between the sink and sensors in DAT and Z-DAT. Also due to the similar reason, the query cost is independent of the shape of *T*; thus, DAT and Z-DAT perform similarly despite the deployment models

CONCLUSION

In this paper, we have developed several efficient ways to construct a logical object tracking tree in a sensor network. We have shown how to organize sensor nodes as a logical tree so as to facilitate in-network data processing and to reduce the total communication cost incurred by object tracking. For the location update part, our work can be viewed as the extension of the work in, and we enhance the work by exploiting the physical structure of the sensor network and the concept of deviation avoidance. In addition, we also consider the query operation and formulate the query cost of an object tracking tree given the query rates of sensors. In particular, our approach tries to strike a balance between the update cost and query cost. Performance analyses are presented with respect to factors such as moving rates and query rates. Simulation results show that by exploiting the deviation-avoidance trees, algorithms DAT and Z-DAT are able to reduce the update cost. By adjusting the deviation-avoidance trees, algorithm QCR is able to significantly reduce the total cost when the aggregate query rates is high, thus leading to efficient object tracking solutions.

REFERANCES

[1]. J. Aslam, Z. Butler, F. Constantin, V. Crespi, G. Cybenko, and D. Rus, "Tracking a MovingObject with Binary Sensors," in *Proc. of ACM SenSys*. ACM Press, November 2003.

[2]. F. Aurenhammer, "Voronoi Diagrams - A Survey of a Fundamental Geometric Data Struc- ture," *ACM Computing Surveys*, vol. 23, no. 3, pp. 345–405, September 1991.

[3]. W.-P. Chen, J. C. Hou, and L. Sha, "Dynamic Clustering for Acoustic Target Tracking in Wireless Sensor Networks," in *Proc. of IEEE International Conference on Network Protocols (ICNP)*,November 2000**.**

[4]. D. Ganesan, R. Cristescu, and B. Beferull-Lozano, "Power-Efficient Sensor Placement and Transmission Structure for Data Gathering under Distortion Constraints," in *Proc. of Int'l Workshop on Information Processing in Sensor Networks (IPSN)*, 2004.

[5]. C.-F. Huang and Y.-C. Tseng, "The Coverage Problem in a Wireless Sensor Network," in *Proc. of ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)*, September 2003.

[6]. C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed Diffusion: A Scalable and Ro- bust Communication Paradigm for Sensor Networks," in *Proc. of 6th Annual International Conference on Mobile Computing and Networking (MobiCOM)*, August 2000.

[7]. B. Krishnamachari, D. Estrin, and S. Wicker, "Modelling Data-Centric Routing in WirelessSensor Networks," in *Proc. of IEEE Infocom*, 2002.

[8]. H. T. Kung and D. Vlah, "Efficient Location Tracking Using Sensor Networks," in *Proc. Of IEEE Wireless Communications and Networking Conference (WCNC)*, March 2003.

[9]. S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, "Supporting Aggregate Queries over Ad-Hoc Wireless Sensor Networks," in *Proc. of 4th IEEE Workshop on Mobile Computing and Systems Application*, 2002.

[10]. K. Mechitov, S. Sundresh, and Y. Kwon, "Cooperative Tracking with Binary-Detection Sen- sor Networks," *University of Illinois at Urbana-Champaign, Technical Report UIUCDCS-R-2003-2379*, 2003.

[11]. G. J. Pottie and W. J. Kaiser, "Wireless Integrated Network Sensors," *Communications of the ACM*, vol. **43**, no. 5, pp. 51–58, 2000

[12]. K. Sohrabi, J. Gao, V. Ailawadhi, and G. J.Pottie, "Protocols for Self-organization of a Wire- less Sensor Network," *IEEE Personal Communications*, vol. **7**, no. 5, pp. 16–27, October200.

[13]. Y.-C. Tseng, S.-P. Kuo, H.-W. Lee, and C.-F. Huang, "Location Tracking in Wireless Sensor Network by Mobile Agents and Its Data Fusion Strategies," in *Proc. of Int'l Workshop on Information Processing in Sensor Networks (IPSN)*, 2003

[14]. Y. Xu and W.-C. Lee, "On Localized Prediction for Power Efficient Object Tracking in SensorNetworks," in *Proc. of Int'l Workshop on Mobile Distributed Computing (MDC)*, May 2003.

[15]. W. Zhang and G. Cao, "DCTC: Dynamic Convoy Tree-Based Collaboration for Target Track- ing in Sensor Networks," *IEEE Transactions on Wireless Communication*, vol. **3**, no. 5, pp. 1689–1701, September 2004.