Tracking Fast Moving Targets in Wireless Sensor Networks

Aysegul Alaybeyoglu, Kayhan Erciyes[1], Aylin Kantarci and Orhan Dagdeviren[2]

Department of Computer Engineering, Ege University, Izmir-Bornova, Izmir, Turkey,
[1]Department of Computer Engineering, Izmir University, Izmir, Turkey
[2]Department of Computer Engineering, Izmir Institute of Technology, Izmir, Turkey

Abstract

We propose a dynamic distributed algorithm for tracking objects that move fast in a sensor network. In the earlier efforts in tracking moving targets, the current leader node at time t predicts the location only for time $t + 1$ and if the target moves in high speed, it can pass by a group of nodes very fast without being detected. Therefore, as the target increases its speed, the probability of missing that target also increases. In this study, we propose a target tracking system that predicts future k locations of the target and awakens the corresponding leader nodes so that the nodes along the trajectory self organize to form the clusters to collect data related to the target in advance and thus reduce the target misses. The algorithm first provides detection of the target and forms a cluster with the neighboring nodes around it. After the selection of the cluster leader, the coordinates of the target is estimated using localization methods and cooperation between the cluster nodes under the control of the leader node. The coordinates and the speed of the target are then used to estimate its trajectory. This information in turn provides the location of the nodes along the estimated trajectory which can be awaken, hence providing tracking of the moving object. We describe the algorithm, analyze its efficiency and show by simulations that it performs well to track very fast moving objects with speeds much higher than reported in literature.

Keywords

Clustering, Distributed algorithms, Leader election, Look-ahead cluster formation, Target tracking, Wireless sensor networks.

1. Introduction

Due to the advancements in low cost embedded processors and wireless transmission technologies, wireless sensor networks have been commonly used in many civil and military applications. Target tracking applications, which are designed for transmitting location information and aggregated data of a mobile target to sinks are among the most common applications of wireless sensor networks.

In a typical tracking application, the first step is to detect the target entering the sensor network when its energy exceeds a predefined threshold [1]. To detect the target, all sensors may stay awake continuously [2] or nodes may periodically oscillate between active and sleeping states to conserve energy as in probabilistic sensing coverage approaches [3,4].

After the detection of the target, its location is calculated by using localization techniques, such as *triangulation*, *trilateratio*n or *Voronoi* cells. All these techniques rely on the fact that the target is located in the intersection of the sensing ranges of the nearby sensor nodes that detect the target [5,6].

Tracking methods in literature can be generally classified as *cluster-based*, *tree-based* and *prediction-based* studies. In *cluster-based* methods, a leader node is selected by a group of nodes and all the nodes in the cluster send their RSS (received signal strength) values to the leader node. Based on these RSS values, the leader node calculates the position of the target using one of the localization techniques mentioned above. *Cluster-based* target tracking algorithms can be grouped as dynamic and static algorithms that can also differ from each other in some respects. In [7], cluster leader nodes and its members are defined statically at the beginning of network deployment. Leader nodes have a higher communication range and more computational power. Besides this, leader nodes can send each other target's location information, while regular nodes cannot. In [8], clusters are formed dynamically by the leader nodes with the highest power. Leader node broadcasts a joint message that includes time and the signature of the data to other nodes. Nodes that have matching data with the data included in the message, send reply messages to the cluster leader and only one cluster leader is active at a time. In [9], clusters are called cells and the size of the cells are determined in accordance with the observed speed of the target. Cell size increases as the target moves faster. In [10], leader nodes are the ones with the highest power and they know the location of every node in their cluster. When sensor nodes detect the target, they only send a notification message to the leader node and store the observed data in their local memory. After the leader node collects notification messages, it runs an algorithm that finds three closest sensor nodes to the target. The leader node calculates the location of the target based on these three closest sensor nodes' observed data. In [11], some nodes are deployed on the border of the network area and these nodes sense at all times to detect the target when it enters the network area. Leader nodes know the identification, location and energy level of each of the nodes in their cluster. Leader node selects three nodes closest to the target's predicted location, which is calculated by the previous cluster. These nodes are woken up to be ready to detect the approaching target.

Another group of tracking algorithms is based on *tree* structure [12]. In these studies, a node closest to the target is selected as a root node and nodes send their observed data to the root node via a distributed spanning tree. As the target moves, some nodes become far away from the target. While these nodes are pruned from the tree, new nodes that become closer to the target are added to the tree. When the distance between the current root and the target is above the predefined threshold, a new root node is selected and the tree is reconfigured.

In *prediction-base*d tracking algorithms [13,14], a prediction-based method is used to predict the next position of the target. By this way, only the nodes around the target's next predicted location will be activated to detect the target. Prediction-based algorithms may assume that the target speed and its direction are constant during its movement (*Heuristic INSTANT*). Alternatively, target speed and direction may be predicted by averaging movement history (*Heuristic AVERAGE*) or different weights may be assigned to movement history stages to obtain a weighted average (*Heuristic EXPAVG*) [13,14].

In the existing studies mentioned above, the current leader node at time t predicts the location only for the time $t + 1$, and if the target moves in high speed, it can pass by a group of nodes very fast without being detected. Therefore, as the target increases its speed, the probability of missing that target also increases. As we propose in this study, awakening the nodes in the predicted trajectory and look-ahead cluster formation with these nodes performs well to track very fast moving targets with speeds much higher than reported in literature.

The rest of the paper is organized as follows. Section 2 describes assumptions and introduces the proposed algorithm in details. An analysis of the proposed algorithm is given in Section 3 and simulation results are shown in Section 4. Finally conclusions and future works are presented in Section 5.

2. The Algorithm

The proposed study focuses on tracking the targets moving very fast. Our aim is to decrease the probability of missing the target and increase the tracking accuracy.

In this section, basic steps of the proposed tracking algorithm including leader election, initial cluster formation and localization, pre-forming the clusters along the trajectory of the target and target tracking are illustrated in more details. We assume that all nodes are aware of their own and immediate neighbors' locations and homogenous nodes with the same sensing and communication ranges are randomly distributed across the tracking area. The transmission range is two times of sensing range so that object tracking can be achieved cooperatively. To save power, sensor nodes stay in sleep mode periodically. The states of the proposed tracking algorithm are illustrated in Figure 1 and the algorithmic representation is given in Algorithm 1.

2.1 Leader Election

When a moving target enters the network area, some nodes in active state and closer to the target can detect it and form the initial cluster by first selecting a node to be the leader of the cluster. We use a two-phase timer-based leader election algorithm, which selects the node closest to the target as the leader node. In this algorithm, each node i that detects the target, sets a timer, which vary in accordance with the node's RSS. The higher the RSS, the smaller the timer value is set. The node, whose timer expires first, is the closest node to the target. Each node waits for expiration of its timer and does not send any message to its neighbors. If the node does not receive any *CANDIDAT*E message until the timer expires, it becomes a leader candidate and sends *CANDIDATE*(*RSS*, *id*) message to its neighbors including its *RS*S and its own identification (id). Otherwise, it gives up and selects the node that sends the *CANDIDAT*E message as the leader node. Since the nodes may not be in one hop communication range of each other, two or more leader candidates may exist after the first phase. For that reason, after the first phase, each leader candidate node i, sets a second timer. If until the second timer expires, the leader candidate node i, receives a candidate message with higher *RS*S value, it gives up the candidacy. As a result, all leader candidate nodes give up, but one with the highest *RS*S value becomes the leader node.

2.2 Forming the Initial Cluster and Localization

To form the initial cluster, the selected leader node sends I *AM LEADER(idi*) message to its one hop neighbors. When the node receives I *AM LEADER*(*idi*) message,

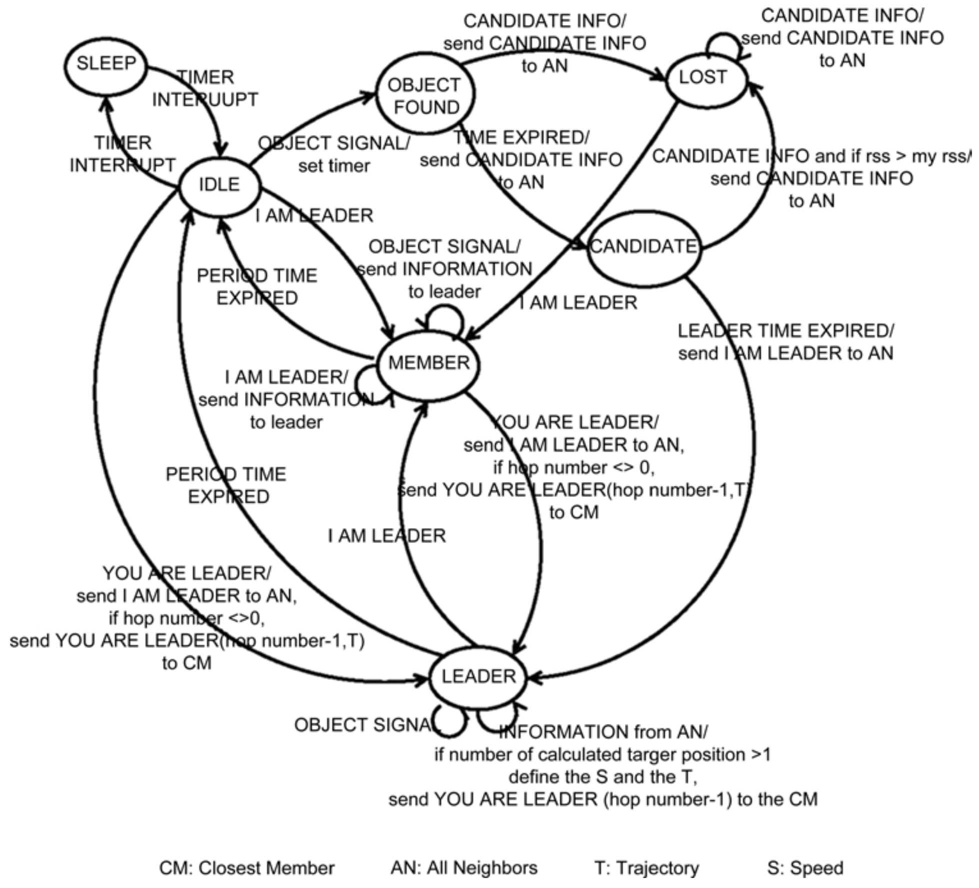
SLEEP
TIMER INTERUUPT
TIMER INTERRUPT
IDLE
OBJECT FOUND
OBJECT SIGNAL/ set timer
CANDIDATE INFO/ send CANDIDATE INFO to AN
LOST
CANDIDATE INFO/ send CANDIDATE INFO to AN
TIME EXPIRED/ send CANDIDATE INFO to AN
CANDIDATE INFO and if rss > my rss/ send CANDIDATE INFO to AN
CANDIDATE
I AM LEADER
OBJECT SIGNAL/ send INFORMATION to leader
PERIOD TIME EXPIRED
I AM LEADER
MEMBER
I AM LEADER/ send INFORMATION to leader
LEADER TIME EXPIRED/ send I AM LEADER to AN
YOU ARE LEADER/ send I AM LEADER to AN, if hop number <> 0, send YOU ARE LEADER(hop number-1,T) to CM
PERIOD TIME EXPIRED
I AM LEADER
YOU ARE LEADER/ send I AM LEADER to AN, if hop number <>0, send YOU ARE LEADER(hop number-1,T) to CM
LEADER
OBJECT SIGNAL
INFORMATION from AN/ if number of calculated targer position >1 define the S and the T, send YOU ARE LEADER (hop number-1) to the CM
CM: Closest Member
AN: All Neighbors
T: Trajectory
S: Speed

Figure 1: Finite state machine of the proposed tracking algorithm.

it becomes a member of the cluster. Every certain time interval, each of these member nodes *i*, sends *INFORMATION*(*RSSi,idi*) message to the leader node. After the leader node has received all *INFORMATIO*N messages from its members, in order to calculate target's location, it selects three member nodes that are closer to the target and calculates the targets current location by *trilateratio*n [15]. After that, the leader node sends this information to the sink node via a distributed spanning tree, which is created at the beginning of the tracking algorithm.

2.3 Look-ahead Cluster Formation

Every certain time interval, member nodes send *INFORMATION*(*RSSi,idi*) message to the leader node and leader node calculates the current location of target, so between sequential time intervals, leader node can also calculate the speed and the trajectory of the target. Due to the target's speed, leader node sets a hop count value, which represents number of look-ahead clusters. In cases where the speed is low, hop count will also be set to the low value so that a few number of clusters will be formed along the trajectory of the target. Therefore, energy consumption due to awakening many nodes in advance will decrease.

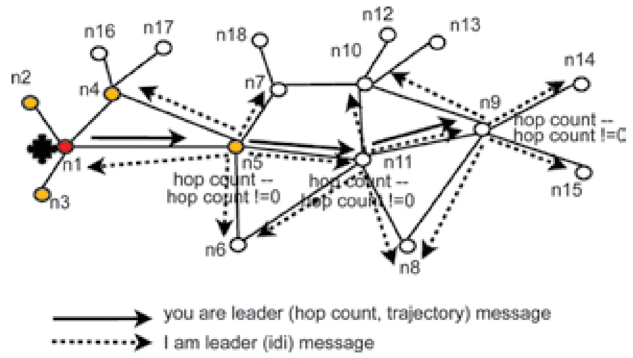

Figure 2: Look-ahead cluster formation.

The nodes in look-ahead clusters will be active and will not sleep as long as they sense the moving target. Figure 2 illustrates how to form look-ahead clusters along the trajectory of the target.

When the current leader node n_1 defines the hop count and the trajectory of the target, it sends *YOU ARE LEADER*(*hop count,trajectory*) message to n_5, which is the closest neighbor to the target's calculated trajectory. When n_5 receives this message, it changes its state to *LEADER* and sends *I AM LEADER(idi)* message to its one hop neighbors(n_1, n_4, n_7, n_6, n_{11}). When the neighbor nodes receive this message, they become members of the newly created cluster and at every

Algorithm 1: Tracking algorithm for node$_j$

```
Algorithm 1 Tracking algorithm for node_j
initially current_state_j = IDLE
    Legend: □ STATE Λ input_message → actions
    V: Target's speed, P_t: Target's position at time t, T: Target's trajectory
    n_T: Neighbor nearest to T, Γ_j: Neighbors of node_j
loop
    □ IDLE    Λ   I-Am_Leader(i) → leader_j ← id_i current_state_j ← MEMBER, set Periodic_Timer
              Λ   Object_Signal → current_state_j ← OBJECT_FOUND, set Object_Timer
              Λ   Timer-Interrupt → current_state_j ← SLEEP, set Timer
              Λ   You_Are_Leader (hop) → send I_Am_Leader to Γ_j, send You_Are_Leader (hop − 1) to nT
                                   send current_state_j ← LEADER, set Periodic_Timer
∴ SLEEP       Λ   Timer_Interrupt → leader_j ← id_i, current_state_j ← IDLE, set Timer
□ MEMBER      Λ   I_Am_Leader (i) → leader_j ← id_i, set Periodic_Timer
        Λ   Object_Signal → send INFORMATION to leader_j, set Periodic_Timer
        Λ   Periodic_Timer_Interrupt → current_state_j ← IDLE, set Timer
        Λ   You_Are_Leader (hop) → send I_Am_Leader to Γ_j, send You_Are_Leader (hop − 1) to n_T
                               current_state_j ← LEADER, set Periodic_Timer
□ LEADER      Λ   Object_Signal → record Object_Signal, set Periodic_Timer
        Λ   INFORMATION   →
                                calculate P_t from all INFORMATION (i) and Object_Signal(j) at time t
                                calculate T and V from P_t and P_{t−1}, send You_Are_Leader (hop) to n_T
        Λ   Periodic_Timer_Interrupt → current_state_j ← IDLE, set Timer
        Λ   I_Am_Leader (i) → current_state_j MEMBER, set Periodic_Timer, leader_j ← id_i
□ OBJECT_FOUND Λ   Candidate_Info(i) → forward Candidate_Info(i) to Γ_j, current_state_j ← LOST
        Λ   Object_Timer_Expired → send Candidate_Info to Γ_j, set Leader_Timer
                               current_state_j ← LEADER
□ LOST        Λ   Candidate_Info(i) → forward Candidate_Info(i) to Γ_j
              Λ   I_Am_Leader (i) → current_state_j ← MEMBER, leader_j ← id_i, set Periodic_Timer
□ CANDIDATE     Λ   Candidate_Info(i, RSS) →
                                if RSS > my_RSS then
                                forward Candidate_Info(i, RSS) to Γ_j, current_state_j ← LOST
                                end if
end loop
```

certain time interval, they send *INFORMATION*(*RSSi,idi*) message to their leader node n_5 as long as they receive signal from the moving target. The new leader node n_5 decreases the hop count value, included in *YOU ARE LEADER* message, by one. If this value is not equal to zero, n_5 sends *YOU ARE LEADER*(*hop count,trajectory*) message to n_{11}, which is the closest neighbor to the target's trajectory. When n_{11} receives this message, it changes its state to *LEADE*R and sends *I AM LEADER*(*idi)* message to its one hop neighbors (n_5, n_6, n_8, n_9, n_{10}). When the neighbor nodes receive this message, they become members of the newly created cluster. This process is repeated until hop count reaches 0. At this time; n_1 clusters have been created while the target is still in the initial cluster. When the target moves in high speed, it will be detected by one of the preformed clusters, so the probability of missing the target will decrease.

2.4 Target Tracking

As stated in the previous section, when the target moves fast through the calculated trajectory, its location is calculated by the look-ahead cluster leader nodes. There can be some cases in which target changes its direction during its movement. The proposed algorithm is designed in such a way to adapt to trajectory changes during the target's movement.

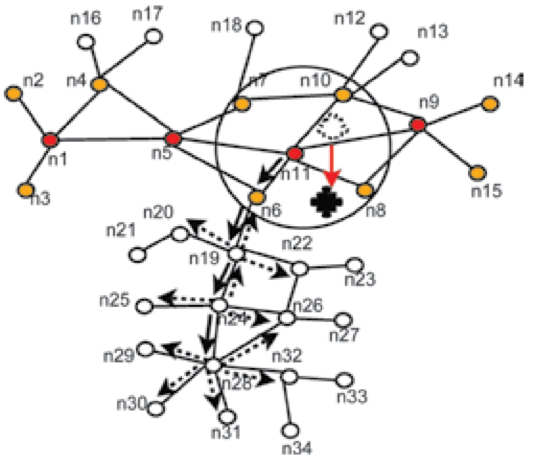

Figure 3: Target changes its trajectory.

Figure 3 illustrates the case when the target changes its direction. In this case, n_{11} notices that the target changes its trajectory. In accordance with the target's new calculated speed, leader node n_{11} sets a hop count value and sends *YOU ARE LEADER(hop count,trajectory*) message to n_6, which is the closest neighbor to the target's new calculated trajectory. According to the value in the hop count parameter, a number of clusters are pre-formed along the target's new trajectory as shown in Figure 3. It should be noted that the nodes in the clusters along the old trajectory are still active. To reduce overall power consumption, if nodes do not sense the target for a certain duration, they make a state transition to the *SLEEP* state periodically as the other nodes in the network.

3. Analysis

Assume a network with N sensor nodes located on a $B \times B$ m^2 area in which a target is moving in +X direction with velocity V and speed s. The sensing range of the nodes are a, and the nodes are sleeping for t_1 s and awake for t_2 s periodically.

If the nodes are uniformly distributed across the network, the number of expected nodes in the area of a^2 and the probability p of the missing target by all the nodes in the area of a^2 at any moment is:

Assume that $a = (ts)/2$, then the probability (p) of the missing target in a period of t time is:

$$p = \int_0^t \frac{t_1}{t_1 + t_2}^{\frac{\pi t^2 s^2 N}{4B^2}} dt \tag{1}$$

Let us now compare the missing rates of generic tracking algorithm and our proposed approach. Assume the target moved in the area with constant speed s until it passed away from the area. Then the expected value of missing the target (M_G) is:

$$M_G = p(B/2s) \tag{2}$$

Assume that our proposed approach constructs k clusters with diameter a. Then the expected value of missing (M_P) is:

$$M_P = p(B/2a - k) \tag{3}$$

Clearly, $M_G > M_P$ for $k >$ and when $k = B/4a$, the expected missing rate reduces to 50%.

4. Simulation Results

We implemented our tracking algorithm in *ns*2 simulator [16] version 2.31. We generated randomly connected networks with 50, 100 and 150 nodes that are uniformly distributed. To measure the effect of network density, we categorized the networks due to node degree as 4, 5 or 6. Our mobility pattern of the target is random waypoint model, which is supported by *ns*2. The speed of the target is varied to measure the detection capability of our algorithm under different mobility conditions. For each scenario, a lower and an upper bound speed is determined. The speeds are respectively chosen from 30 to 40 m/s, 50 to 60 m/s, 70 to 80 m/s and 90 to 100 m/s. We simulated our system up to 100 m/s to see the performance of the system in extreme cases.

The nodes are configured with IEEE 802.11 radio and MAC standards readily available in *ns*2 simulator. Run time of a simulation is 200 s. Three different algorithms are implemented: The Generic Dynamic Tracking Approach (GDTA), which constructs a cluster dynamically upon detection of the target, the Generic Static Tracking Approach (GSTA), which constructs the clusters at the time of network deployment and awakens the cluster along the trajectory of the target and finally our Cluster Ready Tracking Approach (CRTA), which pre-constructs clusters by using the target's calculated future trajectory and speed.

We plotted the target's movements against speed to illustrate the tracking accuracy of our algorithm. As shown in Figure 4a and b, calculated coordinates are very approximate to the real coordinates when the speed is varied between 30 to 40 m/s and 50 to 60 m/s. When the speed is varied between 70 to 80 m/s and 90 to 100 m/s, the difference between real positions and calculated positions is greater; however, it is very important that the trajectory is preserved under very high mobile condi-

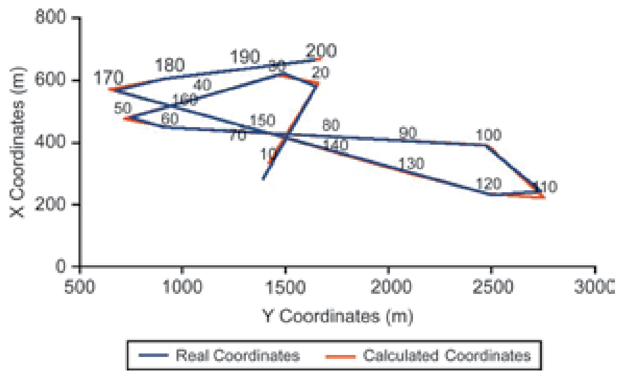

Figure 4a: Mobile scenario.

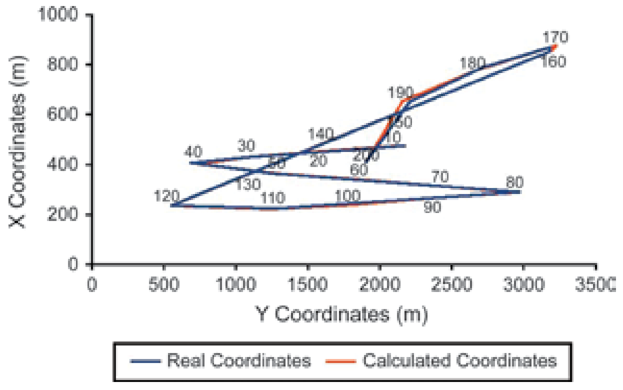

Figure 4b: High mobile scenario.

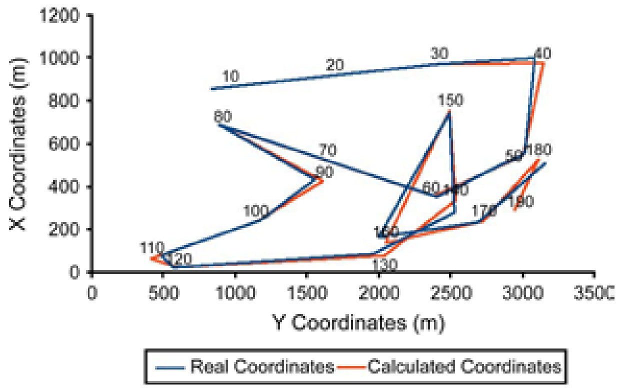

Figure 4c: Very high mobile scenario.

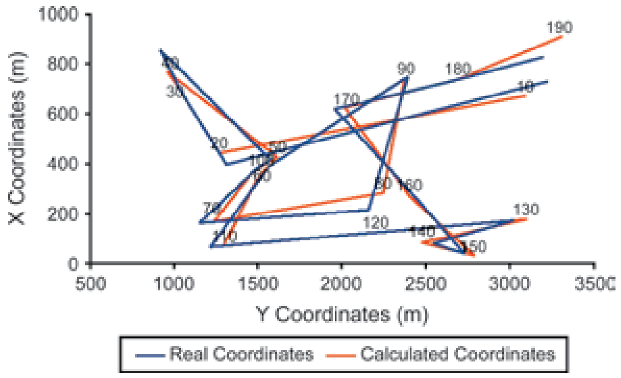

Figure 4d: Extremely high mobile scenario.

tions as shown in Figure 4c and d.

Average distance error between predicted and actual track is 1.44% for 30-40 m/s, 2.56% for 50-60 m/s, 3.58% for 70-80 m/s, 4.29% for 90-100 m/s.

We measured the missing rates of GDTA, GSTA and CRTA against speed. As the lower bound speed increases from 30 to 90 m/s, GDTA and GSTA perform dramatically badly such that missing rates reach up to 45%. The reason of the high missing rate is that the nodes are sleeping while fast target is moving nearby them. On the other hand, CRTA performs very well such that the missing rate is 10% at the worst case as shown in Figure 5. In Figure 6, error ratios of GDTA, GSTA and CRTA are measured against the speed. As the speed increases, the error ratio also increases for each algorithm, but at every speed, the proposed CRTA algorithm performs better than the GDTA and GSTA algorithms.

We also measured the effects of average network density on missing rate. In Figure 7, the missing rate difference between GDTA and CRTA is measured against average node degrees 4, 5 and 6. It can be seen that CRTA performs better than GDTA also in dense networks. When a node's degree increases, the number of nodes in a cluster also increases; this in advance decreases the missing rate. We can state

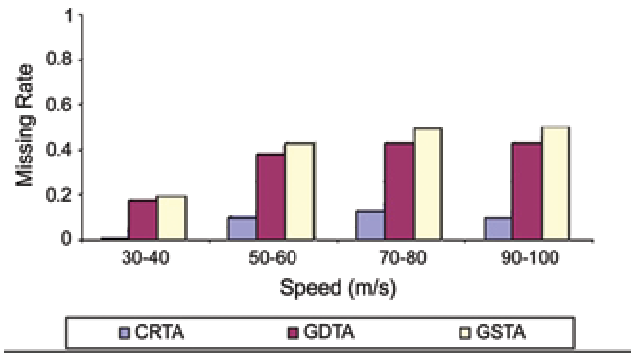

Figure 5: Missing ratio for cluster ready tracking approach, generic dynamic tracking approach, generic static tracking approach.

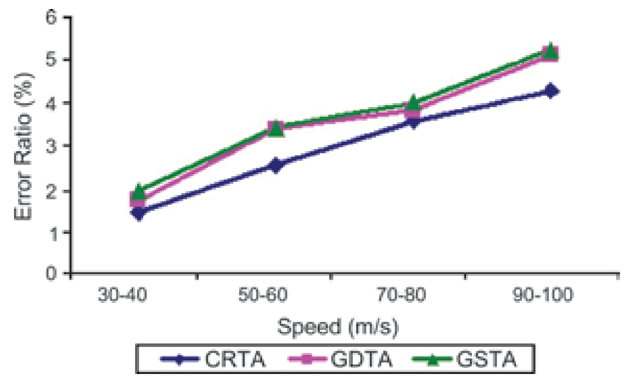

Figure 6: Error ratio for cluster ready tracking approach, generic dynamic tracking approach, generic static tracking approach.

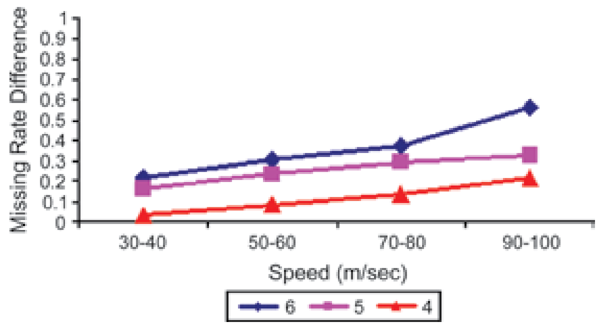

Figure 7: Missing ratio difference between cluster ready tracking approach and generic dynamic tracking approach for node degrees 4, 5 and 6.

that the pre-constructed clusters of awaken nodes can detect the very fast moving target in CRTA. On the other hand, the sleeping nodes miss the target in GDTA, thus CRTA greatly reduces the missing rate. As the node degree increases, CRTA greatly reduces the missing rate, so the missing rate difference between the CRTA and GDTA increases.

The localization error ratios are calculated against the network size and speed for CRTA and GDTA.

As shown in Figure 8, for both algorithms, when the

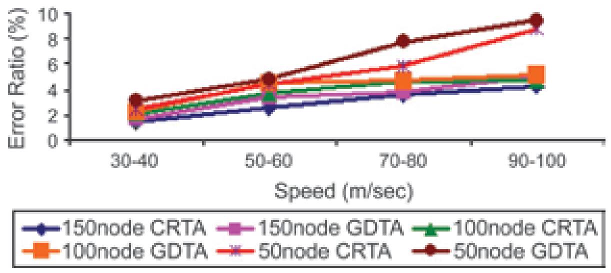

Figure 8: Error ratio in cluster ready tracking approach and generic dynamic tracking approach for different node numbers.

network size increases, the localization error decreases.

Also it can be seen from the Figure that the error ratio of CRTA is approximately 1% smaller than GDTA. It is expected that the error ratios of both approaches are approximate because they use the same localization technique. However missing rate values of GDTA are very high, thus a small number of accurately calculated positions do not give good trajectory information. As the network size is increased, the error ratios decrease approximately by 5% in both approaches.

5. Conclusion

Tracking of fast moving targets is of paramount importance for many civil as well as military applications. In this work, we have provided an algorithm that looks ahead and guesses the possible track of a fast moving target. The algorithm consists of detecting the target, forming a cluster around it with a chosen leader. This cluster estimates the coordinates of the target using localization techniques. The main contribution we hope to provide is estimating the future movement of the target and wakening of the many clusters in this route as in a sliding window. This method provided the tracking of very fast moving targets up to 100 m/s. We showed that both theoretically and experimentally, the algorithm provided outperforms existing approaches reported in literature in terms of accuracy and provides tracking objects moving at multiple of the speeds than various other studies. Our work is ongoing and we are currently investigating on formal methods to more accurately estimate the routes of the target.

References

1. D. Li, K.D. Wong, Y.H. Hu, and A.M. Sayeed. "Detection, classification and tracking of targets in distributed sensor networks", IEEE Signal Processing Magazine, pp. 17-29, Mar. 2002.
2. S. Ren, Q. Li, H. Wang, and X. Zhang. "Design and analysis of wave sensing scheduling protocols for object tracking applications", in International Conference on Distributed Computing in Sensor Systems, California, USA, pp. 228-43, June 2005.
3. C. Gui, and P. Mohapatra. "Power conservation and quality of surveillance in target tracking sensor networks", in ACM Annual International Conference on Mobile Computing and Networking, Pennsylvania, USA, pp. 129-43, Sep. 2004.
4. S. Ren, Q. Li, H. Wang, X. Chen, and X. Zhang. "Analyzing object detection quality under probabilistic coverage in sensor networks", in Thirteenth International Workshop on Quality of Service, Passau, Germany, pp. 107-22, Jun. 2005.
5. R. Gupta. "Tracking moving targets in a smart sensor network", in IEEE Vehicular Technology Conference, Florida, USA, pp. 3035-9. Oct. 2003.
6. Y. Tseng, S. Kuo, H. Lee, and C. Huang. "Location tracking in a wireless sensor network by mobile agents and its data fusion strategies", in Information Processing in Sensor Networks, California, USA, pp. 625-41. Apr. 2003.
7. S. Oh, and S. Sastry. "A hierarchical multiple-target tracking algorithm for sensor networks", in IEEE International Conference on Robotics and Automation, Kobe, Japan, pp. 2197-202, May 2009.
8. W. Chen, and J. Hou. "Dynamic clustering for acoustic target tracking in wireless sensor networks", IEEE Trans. on Mobile Comp., vol. 20, pp. 258-71, 2004.
9. R.R. Brooks, P. Ramanathan, and A.M. Sayeed. "Distributed target classification and tracking in sensor networks", IEEE Signal Processing Magazine, vol. 91, pp. 1163-71, 2002.
10. Y. Zou, and K. Chakrabarty. "Target localization based on energy considerations in distributed sensor networks", in First IEEE Workshop on Sensor Network Protocols and Applications, May 2003, pp. 51-8.
11. H. Yang, and B. Sikdar. "A protocol for tracking mobile targets using sensor networks", in First IEEE Workshop on Sensor Network Protocols and Applications, pp. 71-81. May 2003.
12. W. Zhang, and G. Cao. "DCTC: Dynamic convoy tree-based collaboration for target tracking in sensor networks", IEEE Transactions on Wireless Communications, pp. 1689-701, 2004.
13. Y. Xu, and W.C. Lee. "Prediction based strategies for energy saving in object tracking sensor networks", in 5th IEEE International Conference on Mobile Data Management, Berkeley, USA, pp. 346. Jan. 2004.
14. Y. Xu, J. Winter, and W.C. Lee. "Dual prediction based reporting for object tracking sensor networks", in The First Annual International Conference on Mobile and Ubiquitous Systems, Massachusetts, USA, pp. 154-63, Aug. 2004.
15. K. Bhaskar, B.W. Stephen, and B. Ramon. "Phase transition phenomena in wireless ad-hoc networks", in IEEE Global Telecommunications Conference, San Antonio, Texas, pp. 2921-5, Mar. 2001.
16. VINT Project. "Network Simulator version 2 (NS-2)", in Technical report, Available from: http://www.isi.edu/nsnam/ns. [last cited on 2001 Jun].

AUTHORS

Aysegul Alaybeyoglu received the BSc. degree in Computer Eng. from the University of Sakarya. She is a PhD candidate in Computer Eng. at Ege (Aegean) University, working under the supervision of Assoc. Dr. Aylin Kantarci and Prof. Dr. Kayhan Erciyes. She is a research assistant in University of Ege and currently working on target tracking algorithms in wireless sensor networks.

E-mail: aysegul.alaybeyoglu@ege.edu.tr

Kayhan Erciyes received a BSc. degree in Electrical Eng. and Electronics from the University of Manchester Institute of Science and Technology, England, MSc. degree in Electronic Control Eng. from the University of Salford, England and a Ph.D. degree in Computer Engineering from Ege (Aegean) University. He was a visiting scholar at Edinburgh University Computer Science Dept. during his Ph.D. studies. Dr. Erciyes worked as visiting and tenure track faculty at Oregon State University, University of California Davis and California State University San Marcos, all in the U.S.A. He also worked in the research and development departments of Alcatel Turkey, Alcatel Portugal and Alcatel SEL of Germany. His research interests are broadly in parallel and distributed systems and computer networks. More precisely, he works on distributed algorithms for middleware functions in mobile ad hoc networks, wireless sensor networks and the Grid. Dr. Erciyes is a faculty member at Ege University International Computer Institute,Izmir, Turkey.

E-mail: kayhan.erciyes@izmir.edu.tr

Aylin Kantarci received the B.Sc., M.Sc. and Ph.D. degrees from Ege University, Izmir, Turkey, in 1992, 1994 and 2000, respectively. She is an associate professor at the Department of Computer Engineering at Ege University. Her current research issues include wireless sensor networks, distributed systems, operating systems and video streaming.

E-mail: aylin.kantarci@ege.edu.tr

Orhan Dagdeviren received the BSc. degree in Computer Eng. and MSc. degree in Computer Eng. from Izmir Institute of Technology. He is a PhD candidate in Computer Eng. at Ege University, working under the supervision of Professor Kayhan Erciyes. He is also a research assistant in Izmir Institute of Technology. His interests lie in the computer networking and distributed systems areas. His recent focus is on graph theoric middleware protocol design for wireless sensor and mobile ad hoc networks.

E-mail: orhandagdeviren@iyte.edu.tr

DOI: 10.4103/0256-4602.58973; Paper No TR_163_09; Copyright © 2009 by the IETE