A Distributed Wakening Based Target Tracking Protocol for Wireless Sensor Networks

Aysegul Alaybeyoglu
Celal Bayar University
Computer Engineering Department
Muradiye, Manisa
aysegul.alaybeyoglu@bayar.edu.tr

Orhan Dagdeviren
Izmir Institute of Technology
Computer Engineering Department
Urla, Izmir
orhandagdeviren@iyte.edu.tr

Aylin Kantarci
Ege University
Computer Engineering Department
Bornova, Izmir
aylin.kantarci@ege.edu.tr

Kayhan Erciyes
Izmir University
Computer Engineering Department
Uckuyular, Izmir
kayhan.erciyes@izmir.edu.tr

Abstract

We propose a two layer protocol for tracking fast targets in sensor networks. At the lower layer, the Distributed Spanning Tree Algorithm (DSTA) [12] partitions the network into clusters with controllable diameter and constructs a spanning tree backbone of clusterheads rooted at the sink. At the upper layer, we propose a target tracking algorithm which wakes clusters of nodes by using the estimated trajectory beforehand, which is different from existing studies [3] in which target can be detected only when the nodes close to the target are awake. We provide the simulation results and show the effect of fore-waking operation by comparing error and miss ratios of existing approaches with our proposed target tracking algorithm.

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.

The existing target tracking algorithms in literature can be basically grouped as cluster-based, spanning tree based and prediction based. Clustering is an important approach to manage the scarce resources in sensor networks. By clustering the network, nodes are classified as either clusterheads or members. [1-8] are cluster based tracking algorithms in which member nodes of the cluster detect the object and send information to the clusterhead. When the clusterhead collects all information from its members, it calculates the location of the target and sends this information to the sink node. One of the most important advantage of the cluster based approaches is preservation energy consumption to prolong the network lifetime. Another group of tracking algorithms are based on spanning trees. In [9], nodes that detect the target communicate with each other and select a root node that collects information from the nodes via distributed spanning tree. Besides these, when the root node is far away from the target, tree will be reconfigured. Although the spanning tree based approaches track the moving object more accurately, tree configurations result in high energy consumptions. Finally, [10] and [11] are the prediction based tracking algorithms in which the next position of the target is predicted based on the current moving speed and the direction of the target. In these studies, by using two or more calculated location of the target and the time passed between the locations, the speed and the direction of the target can be calculated.

In the existing studies mentioned above, the current leader/root node at time t predicts the location only for time *t*+1. In case the target moves in high speed, it can pass by a group of nodes very fast without being detected. Therefore, in these studies, as the target increases its speed, the probability of missing that target also increases. Due to the high probability of the missing the target, existing tracking

978-0-7695-4120-4/10 $26.00 © 2010 IEEE
DOI 10.1109/ISPDC.2010.33

studies contain many complicated recovery mechanisms for target missing conditions. These recovery mechanisms consume significant energy to detect the target again. In our study, instead of using these complicated recovery mechanisms, we propose to wake up nodes in the predicted trajectory in advance to decrease the probability of missing the target.

The rest of this paper is organized as follows. In Section 2, the system model is defined. In Section 3 the proposed tracking algorithm as the the upper layer of the protocol is defined, illustrated and analyzed. In Section 4, the simulation results are presented. Finally, conclusions and future works are presented in Section 5.

2. System Model

The protocol for target tracking consists of two layers as seen in Fig. 1. At the upper layer, we propose a wakening based target tracking algorithm (WBTA) that is described in detail in Section 3. Our target tracking algorithm obtains cluster and backbone information from the lower layer. Thus, any algorithm that maintains clusters and a backbone of clusterheads is suitable for this layer. We used a distributed spanning tree based clustering algorithm (DSTA) [12] for the lower layer. DSTA is chosen because it constructs the clusters with controllable diameter and the spanning tree of clusterheads directing to the sink as the backbone at the same time. The depth parameter is provided by the algorithm to adjust the diameter of the clusters.

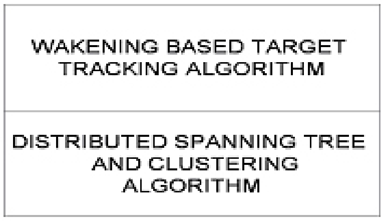

WAKENING BASED TARGET TRACKING ALGORITHM
DISTRIBUTED SPANNING TREE AND CLUSTERING ALGORITHM

Figure 1. Our Protocol Stack

The sink periodically sends *PARENT(nhops)* message to its neighbors to reinitiate the operation. Each node sends the *PARENT((nhops + 1) mod depth)* message to its neighbors upon first reception of the *PARENT(nhops)* message. The recipient of the message with *nhops* = 0 are the *SUBROOTS*, *nhops* < *depth* are the *INTERMEDIATE* nodes, *nhops* = *depth* are the LEAF nodes. An example network clustered with DSTA and *depth* = 1 is shown in Fig. 2. The dotted lines are the borders of the clusters and the bold lines indicate the spanning tree backbone rooted at the sink. As shown in Fig. 2, *SUBROOTS* are the clusterheads, *LEAF* nodes are the cluster members and there is no *INTERMEDIATE* nodes since depth is equal to 1.

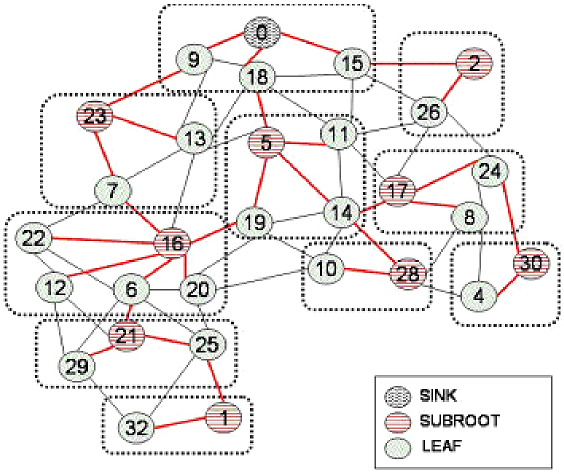


Figure 2. An Example Network with Clusters and Backbone

3. Wakening Based Target Tracking Algorithm

In this section the proposed tracking algorithm is illustrated in detail. Our algorithm aims to decrease the probability of missing the target and to increase the tracking accuracy. In order to do this, clusters along the trajectory of the target are waken up before the target arrives at those locations.

3.1. Assumptions

We have some assumptions about the sensor network as follows :

- Each node has an unique node_id.
- Each node knows its location and the location of its immediate neighbors.
- All nodes have equal sensing and communication ranges.
- Nodes are randomly distributed in the network on a two-dimensional plane with uniform density.
- All nodes are equal in terms of processing capabilities, radio, battery and memory.
- The nodes are stationary and time synchronized.
- Links between the nodes are symmetric. Thus if there is a link from *u* to *v*, there exists a reverse link from *v* to *u*.

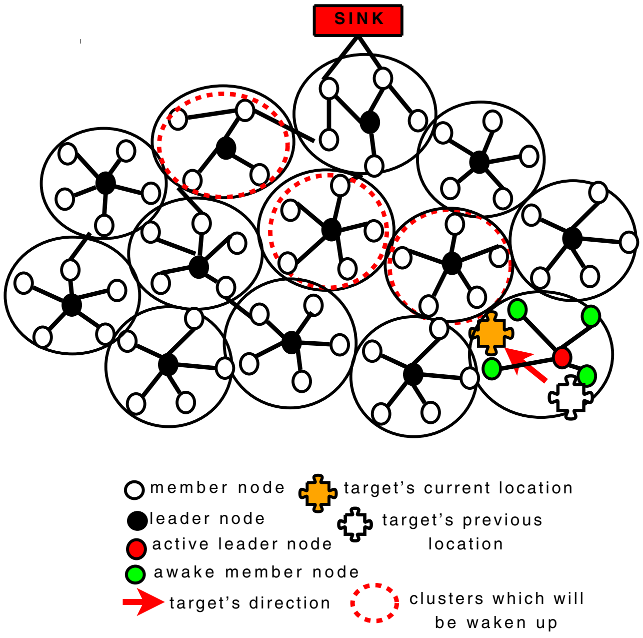

SINK
member node
target's current location
leader node
target's previous location
active leader node
awake member node
target's direction
clusters which will be waken up

Figure 3. General Idea of WBTA.

3.2. Description of The Algorithm

The proposed algorithm is completely distributed and does not require any central point of control point. To save power, sensor nodes stay in sleep mode periodically. Our algorithm is based on a clustered structure and clusters are formed statically at the time of network deployment. Therefore, the proposed algorithm comes into play after the sensors are deployed and the clusters are formed. Fig 3. illustrates the general idea in the proposed tracking algorithm. As the target enters the network area, some nodes in active state detect the target. These nodes are included in a cluster and have a leader node. By using the sensing reports sent by member nodes in the cluster, the leader node calculates the location, speed and the trajectory of the target. Due to the average speed and the trajectory of the target, clusters that are in the same direction with the calculated trajectory are waken up. When the target changes its direction, new clusters are woken up along the new trajectory of the target. Number of clusters that will be woken up, will vary in accordance with the target's speed. In cases where the speed is low, few number of clusters will be waken up, whereas as the speed increases the number of clusters also increases. Therefore, energy consumption due to wakening many nodes in advance will decrease. The states of the proposed tracking algorithm are illustrated in Fig. 4. The algorithmic representation of the algorithm is given in Alg. 1. The list of the states and their description are given below:

- *SLEEP*: When a node is in *SLEEP* state, it stops sensing object, receiving and transmitting data. Therefore, the most energy savings are gained while the nodes are in this state. Thus, to gain from energy preservations, *SLEEP_INTERRUPT* occurs periodically for the the nodes in *MEMBER* and *LEADER* state causing them to make a state transition to *SLEEP* state after handling this interrupt.

- *MEMBER*: If a node which is in *MEMBER* state senses the moving target, with the aim of warning the leader node about the target's getting closer, it sends a *WAKE* message to its leader node. Also when a node in this state receives a *WAKE (hop_count, trajectory)* message, it sends a *WAKE (hop_count-1, trajectory)* message to the neighbor node that is closest to the trajectory of the target, and a $WAKE$ message to the leader node. The nodes periodically make state transition to *SLEEP* state to preserve energy while they are in MEMBER state.

- *LEADER*: A node that is in *LEADER* state, is the leader of a cluster which is responsible for sensing the target as well as collecting and processing its members's sensing information. In this state, when a node senses the moving target, it makes a state transition to *LDR_OBJECT_FOUND* state and sends a $WAKE$ message to its neighbors. Also in this state, when a node receives a *WAKE (hop_count, trajectory)* message, it sends a *WAKE (hop_count-1, trajectory)* message to the neighbor node that is closest to the trajectory of the target, and $WAKE$ messages to the other neighbors.

- *LDR_OBJECT_FOUND*: In this state, when a node receives *OBJECT_RSS (id, RSS)* messages from its members, it calculates the location of the moving target. By using two or more location information of the target, it calculates the trajectory and the speed of the target. After that, it sends *WAKE (hop_count, trajectory)* message to the neighbor node that is closest to the trajectory of the target, If a node receives *WAKE (hop_count, trajectory)* message, it sends *WAKE(hop_count-1, trajectory)* message to the neighbor node that is closest to the trajectory of the target. When the nodes in this state do not sense the target for a certain time duration, a $TIMER\_INTERRUPT$ occurs in each node and they will make a state transition to *LEADER* state. By this way, we will be able to gain from energy consumption.

- *MBR_OBJECT_FOUND*: When a node in this state senses the target, it sends $OBJECT\_RSS$ message to its leader node. If a node receives *WAKE (hop_count,*

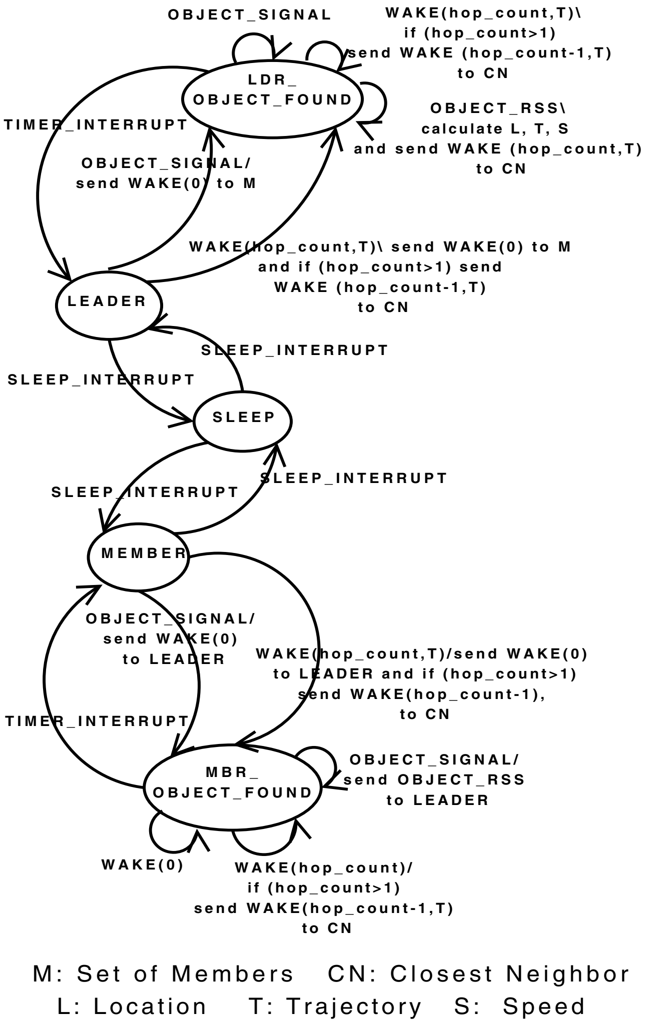

OBJECT_SIGNAL
WAKE(hop_count,T)\ if (hop_count>1) send WAKE (hop_count-1,T) to CN
LDR_ OBJECT_FOUND
OBJECT_RSS\ calculate L, T, S and send WAKE (hop_count,T) to CN
TIMER_INTERRUPT
OBJECT_SIGNAL/ send WAKE(0) to M
WAKE(hop_count,T)\ send WAKE(0) to M and if (hop_count>1) send WAKE (hop_count-1,T) to CN
LEADER
SLEEP_INTERRUPT
SLEEP_INTERRUPT
SLEEP
SLEEP_INTERRUPT
SLEEP_INTERRUPT
MEMBER
OBJECT_SIGNAL/ send WAKE(0) to LEADER
WAKE(hop_count,T)/send WAKE(0) to LEADER and if (hop_count>1) send WAKE(hop_count-1), to CN
TIMER_INTERRUPT
MBR_ OBJECT_FOUND
OBJECT_SIGNAL/ send OBJECT_RSS to LEADER
WAKE(0)
WAKE(hop_count)/ if (hop_count>1) send WAKE(hop_count-1,T) to CN
M: Set of Members CN: Closest Neighbor
L: Location T: Trajectory S: Speed

Figure 4. The Finite State Machine of WBTA.

trajectory) message, it sends *WAKE (hop_count-1, trajectory)* message to the neighbor node that is closest to the trajectory of the target. To preserve energy, nodes in this state which do not sense the target for an amount time will make a state transition to *MEMBER* state similar to the state transition done in *LDR_OBJECT_FOUND* state.

3.3. An Example Operation

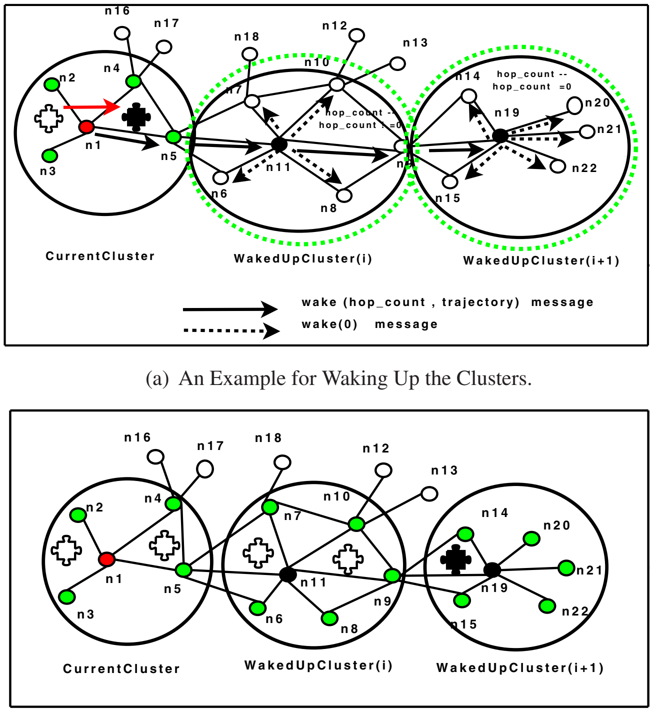
Fig.5 depicts an example for waking up the clusters along the trajectory of the target. In this example, if the cluster in which the target exists currently contains active nodes at that time, the target will be detected by these active nodes. As shown in Fig.5, target is detected by the member nodes n_2, n_3, n_4, n_5 and the leader node n_1. Every certain time interval, these member nodes, send their sensing report to the leader node n_1. After receiving all reports, the leader node n_1 calculates the current location of target and sends this data to the sink node. Between sequential time intervals, leader node can also calculate the average speed and the trajectory of the target. Due to the average speed and the trajectory of the target, clusters that are in the same direction with the calculated trajectory will be waken up. Number of clusters will vary in accordance with the target speed. In cases where the speed is low, few number of clusters will be waken up, whereas as the speed increases the number of clusters also increases.

As shown in Fig.5, with the aim of waking up the clusters along the trajectory, n_1 sends a $WAKE(hop\_count, trajectory)$ message to n_5 which is the closest neighbor node to the target's trajectory. The member node n_5 forwards this message directly to the n_{11} which is the closest neighbor node to the target's trajectory. To wake up the nodes in its cluster, the leader node n_{11} sends $WAKE(0)$ message to all its neighbors (n_7,n_{10}, n_9, n_8, n_6). When the nodes receive $WAKE(0)$ message, they stay awake to be ready to detect the target. After that, the leader node n_{11} decreases the *hop_count* value contained in the message by one and it notices that the *hop_count* value is still greater than zero so it continues to wake up the clusters. 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. Fig. 5 depicts an example for tracking the moving target. As shown in Fig. 5, as long as the target moves, its location will be calculated by the previously woken up clusters. Above all, when the target moves in high speed, it will be detected by one of the previously woken up clusters. Therefore, the probability of missing the target decreases.

All leader nodes send their calculated location information with time stamps to the sink node. With the convergecast method, sink node collects the location information

Algorithm 1 Tracking Algorithm for node$_j$

```
initially $current\_state_j$=IDLE
    $Legend$ : □ STATE ∧ $input\_message \longrightarrow$ actions
    $M_j$: Set of Members, $S$: Speed
    $T$: Target's trajectory
    $n_T$: Neighbor nearest to $T$.
loop
  □ SLEEP ∧ $Sleep\_Interrupt \longrightarrow$
    if $id_j$=$leader_j$ then
      $current\_state_j \leftarrow LEADER$
    else $current\_state_j \leftarrow MEMBER$
    set $Sleep\_Timer$
  □ LEADER ∧ $Sleep\_Interrupt \longrightarrow$
    $current\_state_j \leftarrow LDR\_SLEEP$
    set $Sleep\_Timer$
  □ LEADER ∧ $Wake(hop\_count) \longrightarrow$
    $current\_state_j \leftarrow LDR\_OBJECT\_FOUND$
    send $Wake(0)$ to $M_j$
    if hop_count $> 0$ then
      send $Wake(hop\_count - 1)$ to $n_T$
    set $Timer$
  □ LEADER ∧ $Object\_Signal \longrightarrow$
    $current\_state_j \leftarrow LDR\_OBJECT\_FOUND$
    send $Wake(0)$ to $M_j$
    set $Timer$
  □ LDR_OBJECT_FOUND ∧ $Timer\_Interrupt$ $\longrightarrow$
    $current\_state_j \leftarrow LEADER$
    set $Sleep\_Timer$
  □ LDR_OBJECT_FOUND ∧ $Object\_Signal \longrightarrow$
    record $Object\_Signal$
  □ LDR_OBJECT_FOUND ∧ $Object\_RSS \longrightarrow$
    calculate $L$, $T$ and $S$
    send $Wake(hop\_count)$ to $n_T$
  □ MEMBER ∧ $Wake(hop\_count) \longrightarrow$
    $current\_state_j \leftarrow MBR\_OBJECT\_FOUND$
    send $Wake(0)$ to $leader_j$
    if hop_count $> 0$ then
      send $Wake(hop\_count - 1)$ to $n_T$
    set $Timer$
  □ MEMBER ∧ $Object\_Signal \longrightarrow$
    $current\_state_j \leftarrow MBR\_OBJECT\_FOUND$
    send $Wake(0)$ to $leader_j$
    set $Timer$
  □ MBR_OBJECT_FOUND ∧ $Timer\_Interrupt$ $\longrightarrow$
    $current\_state_j \leftarrow MEMBER$
    set $Sleep\_Timer$
  □ MBR_OBJECT_FOUND ∧ $Object\_Signal \longrightarrow$
    send $Object\_RSS$ to $leader_j$
end loop
```



(a) An Example for Waking Up the Clusters.

(b) An Example for Tracking the Target.

Figure 5. Example Operations of WBTA

from its leaf nodes and store this data with times tamps in a table. If there are more than one leader node that send the location info at the same time stamp, it averages the location values that the leader nodes send.

4. Results

The proposed protocol is implemented in *ns2* simulator version 2.31. We generated randomly connected networks with 50, 100 and 150 uniformly distributed nodes. Random waypoint model that comes with *ns2* simulator is chosen as the mobility pattern. Two way ground was used as the propagation model. We used linear prediction technique to estimate the target's direction. The transmission power is 0.660 mW, the received power is 0.395 mW, and the communication range of a sensor node is 250 m. Nodes periodically sense the environment and when the signal strength they receive from the target exceeds the predetermined threshold, we assume that the node detects the target and the target stays in its sensing range. We measured the performance of the algorithms for average node degree 4 and the size of the surface area is X:2700 m Y:1200 m. The speed of the target is varied to measure the detection capability of our algorithm. For each scenario, a lower and an upper bound speed is used. The speeds are respectively chosen from 30 m/s to 40 m/s (mobile scenario), 50 m/s to 60 m/s (high mobile scenario), 70 m/s to 80 m/s (very high mobile scenario), and 90 m/s to 100 m/s (extremely high mobile scenario). Number of 2, 3, 4 and 5 clusters are awaken for 30-40 m/s, 50-60 m/s, 70-80 m/s, 90-100 m/s speeds respectively.

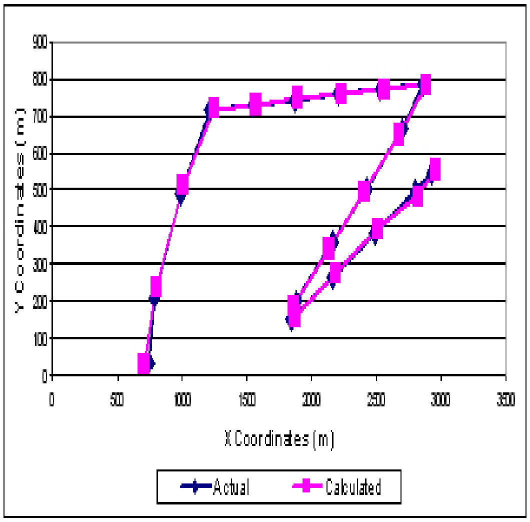


Figure 6. 30-40 m/s

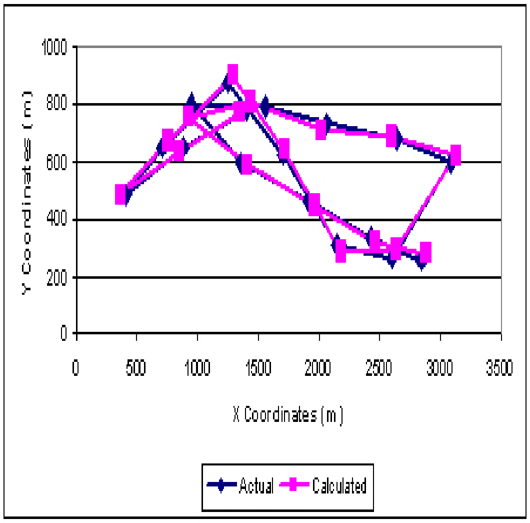


Figure 7. 50-60 m/s

IEEE 802.11 radio and MAC standards readily available in ns2 simulator are chosen for lower layer protocols. DSTA with *depth* = 1 is implemented on top of these standards and under our proposed protocol. The periodic sleep and awake time of the nodes is set to 1 s. Trilateration is used as the localization technique.

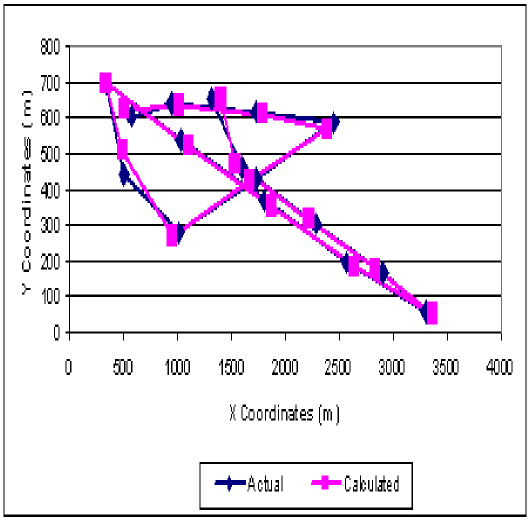


Figure 8. 70-80 m/s

The number of clusters to be waken up (c), is determined by the heuristic: c=$\lfloor(calculated\_speed$ / 10)$\rfloor$ for $calculated\_speed \leq$ 50 and c=6 for $calculated\_speed >$ 50. For example, when the calculated speed is 33 m/s, 3 clusters on the way of estimated trajectory will be waken up and when the calculated speed is 75 m/s, 6 clusters along the trajectory will be waken up.

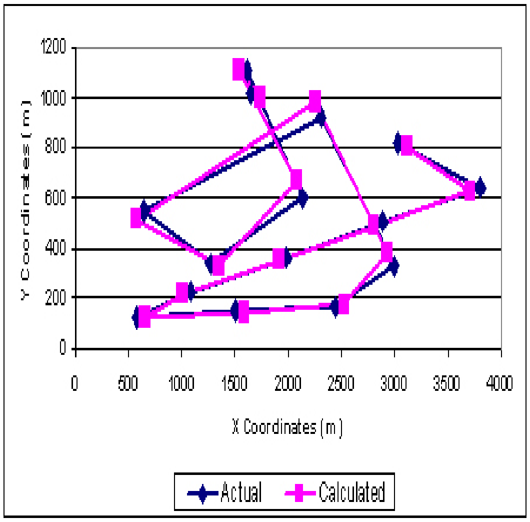


Figure 9. 90-100 m/s

Firstly, we plot the actual and calculated trajectories against the various speeds for a sensor network with 100 nodes. Fig.6, Fig.7, Fig.8, and Fig. 9 respectively shows the trajectories in a sample of mobile, high mobile, very high mobile, extremely high mobile scenarios. For the upper bound speed is equal to 40 m/s, the calculated trajectory is very close to the actual trajectory as shown in Fig.6. As we increase the upper bound on the speed step by step up

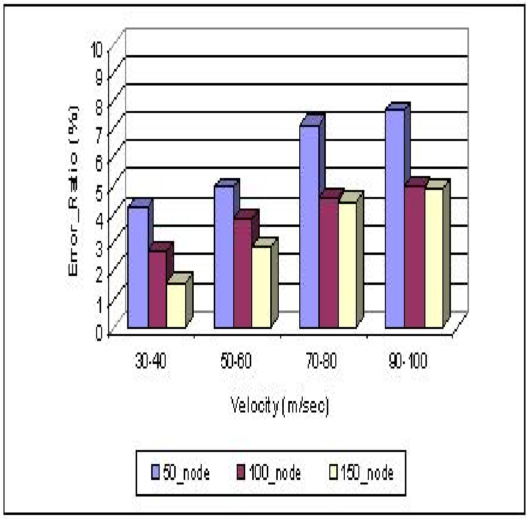


Figure 10. Error Ratio against Speed

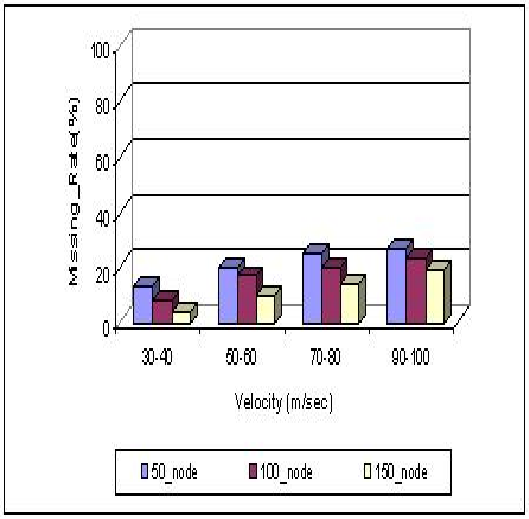


Figure 11. Missing Ratio against Speeds

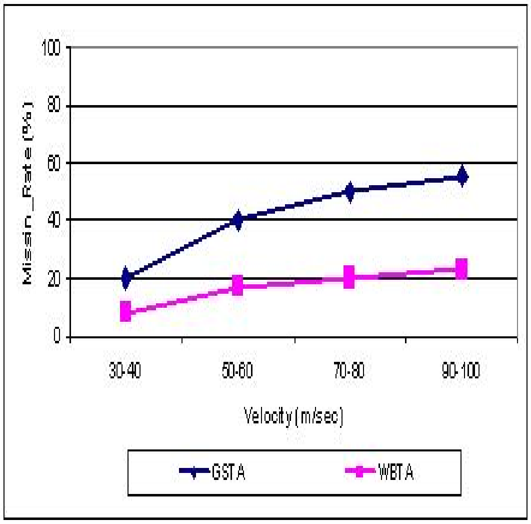


Figure 12. Comparison of Miss Ratios

to 90 m/s, the calculated trajectories are close to actual trajectories meaning that the algorithm is capable of catching the actual trajectory of targets moving at very high speeds. Our system also gives good performance for slowly mobile systems at 2m/s - 30m/s speeds.

The error ratio between actual and calculated positions are measured against number of nodes and target mobility. In the average case, as we increase the network size from 50 nodes to 150 nodes, the error ratio decreases to approximately its half as shown in Fig.10. In the worst case, the error ratio is 7.5 %. We also measured the miss ratio of our algorithm. As the network size is increased, since more number of nodes are ready for sensing both of the error and miss ratio decrease as shown in Fig.10 and Fig.11. In the average case, the miss ratio is approximately 10 % and in the worst case the miss ratio reaches to 22 % in extremely mobile scenarios. To compare our algorithm with existing approaches, we implemented the generic static tracking approach (GSTA) which only detects target when the cluster of nodes nearby the target are awake. On the other hand, our proposed WBTA wakes clusters by using the target's calculated future trajectory and speed. Both of GSTA and WBTA are located at the upper layer of DSTA in the protocol stack. As shown in Fig.12, in the average, the miss ratio of WBTA is 50 % less than GSTA. This result shows that the selection of initially constructed clusters along the target's estimated trajectory decreases the target miss ratio considerably.

5. Conclusion

We proposed a protocol stack for tracking fast targets in sensor networks. At the lower layer, both the clustering and backbone formation services are maintained by DSTA that is suitable for a target tracking application. At the upper layer, WBTA obtains the clustering and backbone information and finds the positions of the target from the lower layer. The cluster of nodes along the target's estimated trajectory is waken beforehand to reduce the miss ratio. The simulation results showed that WBTA calculates close to the real trajectory even when the average speed of the target reaches to 95 m/s. In the worst case, the measured error ratio of the WBTA is 7.5 %. The miss ratio of the WBTA is about half of the miss ratio of GSTA in the average case. Generally, the results show that wakening nodes on the predicated way of a fast target is useful for target detection. Our work is ongoing and we are extending our algorithm for multitarget tracking.

5.1. References

[1]Yang, W., Fu, Z., Kim, J., Park, M., ”An Adaptive Dynamic Cluster Based Protocol for Target Tracking in Wireless Sensor Networks”, Proc. of. WAIM 07, 157-167 (2007).

[2]Chen, W., Hou, J., ”Dynamic Clustering for Acoustic Target Tracking in Wireless Sensor Networks”, IEEE Trans. on Mob. Comp. , **20**, 258–271 (2004).

[3] Brooks, R., Ramanathan, P., Sayeed, A., ”Distributed Target Classification and Tracking in Sensor Networks”, IEEE Signal Proc. Magazine, **91**, 1163-1171 (2002).

[4]Suganya, S., ”A Cluster Based Approach for Collaborative Target Tracking in Wireless Sensor Networks”, Proc. of ICETET 08, 276-281 (2008).

[5]Yang, H., Sikdar, B., ”A Protocol for Tracking Mobile Targets Using Sensor Networks”, Proc. of SNPA 03,7181 (2003).

[6]Olule, E., Wang, G., Guo, M., Dong, M., ”RARE: An Energy-Efficient Target Tracking Protocol for Wireless Sensor Networks”, Proc. of ICPPW 07, 76 (2007).

[7]Xu, Y., Lee, W., ”Compressing Moving Object Trajectory in Wireless Sensor Networks”, Int. Journal of Dist. Sensor Networks, **3**, 151-174 (2007).

[8] Goshorn, R., Goshorn, J., Goshorn, D., Aghajan, H., ”Architecture for Cluster Based Automated Surveillance Network for Detecting and Tracking Multiple Persons”, Proc. of ICDSC 07, 219-226 (2007).

[9] Zhang, W., Cao, G., ”DCTC:Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks”, IEEE Trans. on Wireless Com., 1689-1701 (2004).

[10] Wang, Z., Li, H., Shen, X., Sun, X., ”Tracking and Predicting Moving Targets in Hierarchical Sensor Networks”, Proc. of ICNSC 08, 1169-1173 (2008).

[11] Mechitov, K., Sundresh, S., Kwon, Y., Agha, G., ”Cooperative Tracking with Binary-Detection Sensor Networks”, Proc. of ACM Sensys 03, 332-333 (2003).

[12]Erciyes, K., Ozsoyeller, D., Dagdeviren, O., ”Distributed Algorithms to Form Cluster based Spanning Trees in Wireless Sensor Networks”, Proc. of ICCS 08 (2008).