Distributed Fault Tolerant Target Tracking Protocol for New Face-based Wireless Sensor Networks

Ammara Razzaq, Ahmed M.Khedr, and Zaher Al Aghbari

***Abstract*—Wireless Sensor Networks (WSNs) consist of sensor nodes that have limited batteries, and are hard to replace due to their deployment in inaccessible areas. Hence, the goal of any WSN protocol is to implement energy efficient mechanisms that prolong the lifetime of WSNs. The existing face topology in WSNs can lead to high energy consumption due to all the nodes in the network being part of topology. In this paper, we propose a new protocol that introduces a new distributed energy-based face structure by putting redundant nodes to a sleep state for reduced energy consumption and prolonged lifetime of network. A distributed target tracking algorithm is designed to run on the new proposed face structure which tracks the target accurately while reducing the energy consumption of network. Additionally, a distributed fault tolerance algorithm is proposed which uses the sleeping redundant nodes in the network as backup nodes. When a node's battery reaches a critical level it selects a nearby sleep node to replace itself to minimize the interruption in tracking of the target. The simulation results show that our proposed scheme achieves better energy efficiency and improves the lifetime of network while maintaining the performance of tracking. At high speeds, our work achieves significant improvement in accuracy of tracking.**

***Index Terms*—Distributed Target Tracking, Face Topology, Fault Tolerance, Wireless Sensor Networks.**

I. Introduction

WIRELESS sensor networks (WSNs) consist of hundreds and thousands of low power and low cost sensor nodes which are deployed in a random or deterministic manner to monitor different environmental attributes. WSNs have many constraints that need to be dealt with while developing an application. Energy is the biggest resource constraint in WSNs [1], [2]. Therefor, the need is to develop energy efficient algorithms that consume minimum energy and perform accurately. Minimum energy consumption will lead to prolonged network lifetime. Energy is consumed during computation at sensor nodes and communication between sensor nodes. However, communication cost is the main reason for energy depletion in sensor nodes. The goal is to reduce these costs and increase the lifetime of network [3], [4], [5].

Sensor nodes have three possible states; active, awake and sleep state. In active state, sensor nodes are receiving and transmitting messages. Hence, this is the most energy consuming state. In awake state, nodes are only listening to messages and will become active if they receive a request message from other nodes to send messages. In sleep state, nodes are neither listening nor sending any messages. Hence, this is the most energy saving state. Therefore, to save energy it is best to keep nodes in sleep state for as long as possible.

A topology in WSNs is the grouping of nodes to collect data cooperatively in an energy conserving manner. Mainly three types of network topologies exist in WSNs, *tree*, *cluster* and *face* [6], [5], [7] topology. In face topology, network is divided into a set of polygons called faces. A face consists of minimum three nodes. A face structure is a planar graph where nodes act as vertices of the graph and edges between nodes represent the communication links. A planar graph is a graph which has no crossing edges. Gabriel Graph (GG) and Relative Neighborhood Graph (RNG) are two well-known planar graphs which are used to build face structure [8]. WSNs are usually densely deployed, which means there are many nodes in each area of the network [9]. Hence it is not always necessary to keep all the nodes in active state. In the existing face topology, all the nodes in the WSN become part of topology which can lead to small face sizes and more energy consumption. A too small face size can cause irregular signal patterns affecting the accuracy of application running on the network [10].

Many distributed tracking protocols have been designed for different network topologies such as, cluster based, tree based and face based target tracking protocols [7], [11], [12], [13], [14], [15]. The goal of these target tracking applications is to gather data about one or multiple targets and estimate their positions in the network [16]. In target tracking applications, a node detects the presence of target by sampling the sensed signals (e.g., light, sound, image or video), then these readings are used to estimate the trajectories of one or more targets and notify the sink node [17], [18]. The main goal of a target tracking protocol is to design an energy efficient mechanism that tracks the target accurately and prolongs the lifetime of WSN. In face-based target tracking protocols, once the target is detected, two or more number of nodes collaborate with each other to track the target. Once the target moves away from these nodes, new set of nodes are selected to track the target. One of the nodes from the set will become beacon node that will be responsible for communicating with the tracker.

In [14], [15], [13], object tracking protocols have been developed in face-based WSNs. However, none of these works have tried to modify the topology of the network to adapt to new situations such as, identification of certain nodes to put to sleep state to minimize energy consumption. In these approaches, all the nodes in network take part in the topology formation, which is energy consuming. Due to the small faces, objects moving at high speed will leave the face quickly increasing the chances of target loss. The main focus of these protocols have been on designing the prediction protocols to predict the future location of target

Manuscript received August 29, 2019; revised December 11, 2019.

Ammara Razzaq did masters in Computer Science from Univesity of Sharjah, Email: ammara_razzaq@outlook.com.

Professor Ahmed M. Khedr is with Department of Computer Science, University of Sharjah, UAE, Email: akhedr@sharjah.ac.ae.

Professor Zaher Al Aghbari is with Department of Computer Science, University of Sharjah, UAE, Email: zaher@sharjah.ac.ae.

and only awakening the nodes in predicted region to save the energy. However, in prediction based protocols, a target can be easily lost if it does not move towards the predicted region. In this case, a target recovery mechanism will be required to recover the target by activating as many number of nodes as required which can lead to excessive energy consumption.

Sensor nodes are prone to failure due to harsh environmental conditions, malicious attacks on the network, hardware failure such as battery depletion or software failure such as wireless channel fluctuations [19]. Faults in the network mainly occur due to node, network or sink problems [20]. Faulty nodes cause the WSN to be partitioned, causing holes in the network, and disrupting the performance of the network as a consequence. Faults in sensor nodes can cause error in sensing, processing and communicating data [21]. In target tracking applications, partitioning of the network will result in loss of tracking and the accuracy of the tracking protocol will be reduced. Distributed protocols require self healing to deliver a desired level of functionality in the presence of faults [22]. The ability to deal with faults in the network is an important factor in the performance of WSNs.

In this paper, a new energy-based face structure is proposed in which extra nodes are put in sleep state while building the face structure. The resulting network will have generally bigger faces which will solve the problem of signal irregularities. Due to sleep nodes in the network the overall energy consumption of face structure will be less and the life time of network will be improved. To test the performance of the proposed face structure a new target tracking protocol is designed to compare with existing face based tracking protocols. To further enhance the lifetime of network and ensure uninterrupted tracking in the event of failure of a node due to battery depletion, a new distributed fault tolerant protocol is designed which replaces the faulty node with a nearby sleeping node by waking it up and using it as a replacement node. The simulation results show that the proposed scheme improves the energy efficiency and prolongs the network lifetime, while maintaining the performance of tracking protocol.

The remaining of this paper is organized as follows: In Section II related work is presented. Proposed scheme is presented in Section III and experimental evaluation is presented in section IV. Conclusion is given in Section V. Mathematical notations used in proposed work are mentioned in Table. I.

TABLE I
MATHEMATICAL NOTATIONS

Notation	Meaning
A_N	Active Node
S_N	Sleep Node
B_N	Boundary Node
AE_N	Active node that has made edges
IS_N	Invalid Sleep Node
F	Face
B	beacon node
C	coop node
ctp	coop track packet
bup	beacon update packet
Δt	Time interval of battery monitoring
$TL1$	First battery threshold level
$TL2$	Second battery threshold level
N_s	Nearest sleeping node
R_p	Replacement Node
N_d	Node whose battery is depleting
E_d	List of edge nodes of N_d
$E_d[i]$	A node in E_d

II. RELATED WORK

In Gabriel Graph (GG) based face structure [8], an edge is valid between two nodes if there is no node inside the circle formed by the edge as a diameter, otherwise the edge is invalid. In Relative Neighborhood Graph (RNG) based face structure [23], an edge exists between two nodes if there is no node inside the intersection area of their communication circles. In both structures, all the nodes in network become part of topology. In dense WSNs, where nodes lie very close to each other, the resulting topology will consist of large number of faces which will be very energy consuming. In [24], a redundancy aware face structure is proposed in which redundant nodes are not included while building the topology of the network. Redundant nodes are identified by analyzing the coverage of the network. If the sensing area of a node is already covered by nearby nodes, the node is considered redundant.

In face based tracking, object is tracked through faces in the network. One or more nodes in every face will be responsible for tracking the target. A mobile sink/tracker may be chasing the target in the network by getting information through representative nodes in each face. Many object tracking protocols have been designed using face structure in WSNs such as [11], [12]. These protocols can be divided into prediction based and non-prediction based tracking protocols.

The prediction based protocols such as [25], [26] aim to track the target using as less number of nodes as possible and focus on developing a prediction mechanism to predict the future location of the target. The nodes in the predicted region are awaken to continuously track the target hence they are more energy efficient, but they can lose the target very easily if it doesn't move to the predicted area and a target recovery mechanism will have to be invoked to rediscover the target.

In non-prediction based protocols such as [27], [28], [13], [29], a target is usually trapped by one or more faces, to track the target. The target will be tracked in whichever direction it moves, because it is covered from all sides. Hence no prediction is required to predict the future position of target. Non-prediction mechanisms are more accurate than prediction based mechanisms as they don't lose the target easily. However, they are more energy consuming depending on how many nodes are tracking the target at a time. In [27], a dynamic object tracking (DOT) solution is provided using face architecture. It uses the spatial neighborhood discovery algorithm to track the target. Sensor node that is nearest to the target acts as beacon node and wakes up all its spatial neighbor nodes to cooperatively track the target, in order to avoid losing target tracks. However, waking up all the spatial neighbor nodes consumes a lot of energy. In this work, no prediction mechanism is applied to predict the future position of the target.

In [28], only the beacon node and one of its immediate

neighbor nodes cooperatively track the target. Beacon node is responsible for communicating with the tracker. A beacon node determines the next beacon node based on a prediction mechanism. In Face-based Object Tracking Protocol (FOTP) [13], a hexagon based prediction mechanism is designed to reduce the number of awaking nodes. Instead of waking up all the spatial neighbors, this algorithm proposes a hexagon based prediction mechanism to predict the future location of target. One drawback of this approach is that, target can easily be lost since prediction generates future location based on only the current and prediction location. If the target is moving fast, it will be lost by the time spatial neighbors are activated, and then all the sensor nodes will need to wake up to recover the target. Secondly, random soldier nodes are used to detect the entrance of target. The target may not be detected at all until it comes in the vicinity of one of the soldier nodes. In [29], a Prediction Based Object Tracking approach (POOT) along with two target recovery schemes are presented. To detect the presence of target in network, the source uses the flooding mechanism. The target can be detected by more than one nodes, and the nearest node to the target takes the responsibility of tracking the target.

All the above face based tracking protocols have not focused on building a new planar topology for better energy efficiency and prolonging the lifetime of WSNs. In our proposed approach, we design a new face structure that has the main goal of reducing the energy consumption and prolonging the network lifetime. In the new proposed non-prediction based target tracking algorithm, only the face in which target is present is used to track the target. The target crossing a face is detected using an edge detection algorithm. Upon edge detection, the forward polygon is informed that the target is approaching and the nodes in forward polygon wake up and wait for the target to enter the face.

In the literature, fault tolerance has been studied in various aspects categorization of fault tolerance techniques is centralized vs. distributed [30]. In a centralized fault tolerance approach, the nodes will have to send their data to base station and the base station node will implement the fault tolerance mechanism. In a distributed fault tolerant protocol, nodes can detect faults and recover from faults locally without any communication with the base station.

Although fault tolerance has been extensively studied in WSNs, not much work has been done in face based WSNs. In [31], a fault tolerance protocol is presented which suggests merging two or more faces into one single face. Merging of faces is done once the fault is detected in the network.

Some other face based target tracking protocols presented target recovery protocols in case of target loss. Target loss can happen as a result of faults in the network like node failures, link failures etc. A target recovery mechanism tries to find the target by gradually increasing the number of active nodes where the target was last found. Most papers such as [32], [26] that work with face based target tracking protocols, focus on target recovery mechanism in case of loss of tracking of target. In [32] a target recovery mechanism is given in which the neighbor sensors close to the monitor node try to relocate the target. If the target is not found all the nodes in the face cooperate to find the target. If the target is still not detected, all the neighbor sensors in close vicinity of the face try to recover the target. If all tries fail, the network returns to the initial state where every node in the network tries to relocate the target. In [26] in case of target loss. If the target does not move to the predicted face, the target is assumed to be missing and the target recovery mechanism is triggered. Firstly, the previous face is awaken to check if the target is still there. If the target is not found, the vicinity faces are awaken one by one until the target is found. Lastly, if the target is not in any of the vicinity faces, the entire network will be awaken to find the target.

In this paper, we propose a new fault tolerance technique which replaces the faulty node with the nearest redundant node in a distributed manner, that prolongs the network lifetime. This approach tries to avoid the loss of target by immediately trying to replace the faulty node as soon as the fault is detected.

III. Distributed Fault Tolerant Target Tracking Protocol for new Face-based WSNs

In this section, we describe the proposed scheme. The proposed scheme includes three new algorithms: energy-based face structure algorithm, target tracking algorithm which uses the proposed face structure to track an object and finally fault tolerance algorithm that replaces the faulty node with the nearest redundant node to avoid network partitioning and to prolong the network lifetime.

A. Energy-based Face Structure

The proposed energy-based face structure algorithm requires that each node makes edges at individual time, hence a distributed scheduling algorithm is used for this purpose. At first, the nodes will gather the information of their communication neighbors using neighborhood discovery protocol. Then each node will make edges, at its assigned time slot. A node will make edges with nodes that are farthest in sensing range and put the nearby nodes to sleep state by running the proposed edge making procedure . Once all the nodes have made edges, the result is the proposed face structure where nodes that made edges have become part of topology and remaining nodes have gone to sleep state. Next every node runs a face discovery procedure to discover the spatial neighborhood nodes. The details of the face structure algorithm is described in the following procedures:

1) Neighborhood Discovery Procedure: In this procedure, the information of the neighboring nodes is gathered by sending broadcast packets by each node. A broadcast packet is a packet which is transmitted without any recipient node address and all the nodes in communication range will hear the broadcast [33]. According to the neighborhood discovery procedure, the recipient nodes will save the sender node as a communication neighbor in their memory. A communication neighbor is one-hop neighbor of a node with which a direct communication link exists.

The neighborhood discovery procedure in algorithmic form is given in Fig. 1. The update_Neighbor_Table function stores the information of communication neighbors in memory along with their ids and locations.

2) Boundary Discovery of the Network Procedure: Once the neighbor nodes are discovered the next step is to discover the boundary of the network to ensure that boundary nodes are not put to sleep state and the boundary of the network

Require: $\emptyset$
Ensure: Set of communication neighbors
1: INITIALIZATION: send Broadcast Discovery packets
2: **if** Received Broadcast Discovery Packet **then**
3: update_Neighbor_Table()
4: **end if**

Fig. 1. Neighborhood Discovery Algorithm

remains intact. The boundary of the randomly deployed WSN can be discovered using a distributed boundary discovery procedure [34], [35]. The goal is to discover the minimum perimeter coverage of the network i.e. the minimum number of nodes that are covering the perimeter of the network will be declared as boundary nodes. For this purpose, minimum perimeter coverage procedure from [34] is used.

3) Distributed Slot Scheduling Procedure: Each node runs the edge making procedure where it makes edges with some nodes in its sensing range and puts some nodes to sleep state. Since some nodes will go to sleep state, the edge making process of one node will affect the edge making process of other nodes. The nodes that go to sleep state should not make edges and the nodes that have been made edges with, will not be put to sleep state. A distributed scheduling procedure [36], [37] is used to schedule the time slots for each node in which it will run the edge making procedure in a distributed manner.

4) Edge Making Procedure: In this procedure, each node makes edges according to the proposed edge making criteria. Every node works at its own allotted time assigned using the distributed scheduling procedure. Edge making procedure is used by nodes in the network for building the proposed face structure. A node tries to make edges with the nodes that lie inside its sensing range and sort them based on the distance from farthest to nearest nodes. Then, the node checks the validity of the edges with each node in the sorted order, starting from farthest node. If there is a node inside the circle formed by the possible edge as diameter, the node will check if it can be put to sleep state. If the node is a boundary node B_N, it will not be put to sleep state, because making a boundary node asleep will result in reduced boundary of the network.

If the network is represented as a graph $G = (V, E)$, nodes in the network represent the vertices V of the graph and communication links between nodes represent the edges E. In proposed face structure, if a node has two neighbors, it will pick the farthest neighbor node and check the validity of edge with this node.

An example of a node making edges with all its sorted neighbors according to edge making criteria is shown in Fig. 2. Node $n1$ has five neighbors inside sensing range $[n2, n3, n4, n5, n6]$, given in sorted order as $[n3, n4, n2, n5, n6]$. $n1$ checks the validity of edge with $n3$, $n5$ is lying inside the circle formed by the edge $e_{n_1n_3}$, and it is not a boundary node, hence it is put to sleep state, and an edge is made. Then, $n1$ checks the validity of edge with n_4. n_6 is lying inside the circle formed by edge $e_{n_1n_4}$ as diameter and it is not B_N hence it is put to sleep state and edge $e_{n_1n_4}$ is termed valid. Next node in the sorted list is n_2, and there is no node inside the circle formed by edge $e_{n_1n_2}$ as diameter, hence it is also valid. Next in the sorted

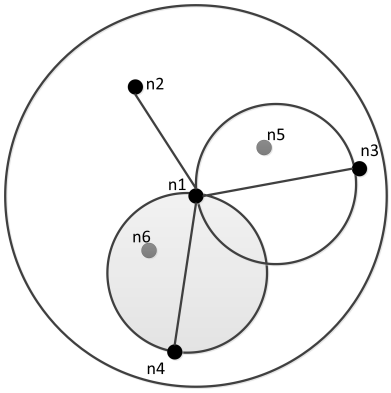


Fig. 2. Edge Making for Proposed Face Structure

Require: Nodes in sensing range
Ensure: Edge nodes and sleep nodes
1: Sort_Sensing_Neighbors()
2: **for** all sorted nodes **do**
3: **if** active node **then**
4: **if** no I_{SN} node inside circle formed by edge as diameter **then**
5: edge is valid
6: **else**
7: edge is invalid
8: **end if**
9: **end if**
10: **end for**

Fig. 3. Edge Making Algorithm

list are n_5 and n_6, which are both asleep.

Initially, there were only active nodes (A_N) and boundary nodes (B_N) in the network. Once nodes have started making edges, there will be two more types of nodes in the network, sleep nodes (S_N) and nodes that have made edges (AE_N). These nodes broadcast their information so that their neighbors know their current status. The sleep nodes will not make any edges.

When further nodes make edges, they will check if a node in their sensing range is a sleep node and they will not make edges with sleep nodes. It is also possible that S_N or AE_N lie inside the circle formed by the edges as diameter. In case of S_N, edge will be valid because it is already a sleep node. If S_N or A_N lie inside the circle formed by the edge e between nodes u and v, the edge e is valid. In case of A_N, it will be put to sleep state. S_N does not need to be put to sleep state because it is already asleep.

In case of AE_N, edge will be termed invalid, because it is a node that has already made edges. If it is put to sleep, its edges will be broken. If its edges had put some other nodes to sleep, they will have to be awoken. It can be called *reverse effect* in which the process that has already been done needs to be undone. In order to avoid *reverse effect* the edge is termed invalid.

Edge between u and v is invalid, because a boundary node B_N or AE_N lie inside the circle formed by the edge. Since these nodes cannot be put to sleep, they are called invalid sleep nodes IS_N.

Edge making procedure in algorithmic form is given in Fig. 3.

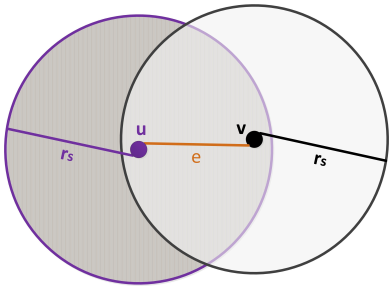


Fig. 4. Coverage of edge in energy-based face structure

Further in this section, few characteristics of energy-based face structure are discussed, such as, coverage property, condition on minimum number of edges, and proof that proposed face structure is a planar graph.

a) Coverage Property of Edges: All the edges in proposed face structure are fully covered as one node makes edges with nodes that lie inside the sensing range. Hence, a node leaving a face by crossing one of its edges will be detected. This ensures that a target can never be lost due to escaping the network through a coverage hole. As shown in Fig. 4, an edge *e* between nodes *u* and *v* is covered by sensing ranges of both nodes. Even if one node dies the other node will be able to detect the presence of target in the intersected region.

b) Condition on Minimum Number of Edges: For a node to make a face, it needs to make at least two edges. If a node makes less than two edges it will result in incomplete face structure as shown in Fig. 5. In Fig. 5a $n1$ has three nodes in its sensing range $n2, n3, \& n4$, it made edge with $n4$ and put $n2$ and $n3$ to sleep. Now there are no other nodes to make edges with, hence $n1$ is only able to make one edge, so it cannot make even one face. In this scenario, proposed face structure is incomplete.

If such a case occurs, in which a node is unable to make at least two edges, it will try to make Gabriel edges. In Fig. 5b, node $n1$ made Gabriel edges with nodes $n2$ & $n3$. Hence, it made a complete face.

c) Energy-based Face Structure is Planar Graph: The proposed face structure is a planar graph, i.e. there are no crossing edges in the graph.

In Fig. 6a, a fully connected graph between nodes $p, q, r \& s$ is shown. The $e6$ and $e5$ are crossing each other, hence the fully connected graph between these nodes is not a planar graph. If p checks the validity of edge $e6$ with q as shown in Fig. 6b, nodes r & s lies on the circle formed by the edge $e6$ as diameter.

In the proposed graph, if node p makes edges first, it will pick the farthest node q and check the validity of edge $e6$. Nodes r & s are lying on the circle formed by the edge $e6$ as diameter, so they are put to sleep. Hence, the proposed graph does not have any crossing edges as shown in Fig. 6c.

An example of energy-based face based WSN is shown in Fig. 7. n1, n4, n10, n11, n20, n27, n35, n37, n38, n39, n40, n31, n26, n18, n5, & n3 are boundary bodes of the network. Some nodes have been put to sleep and did not make any edges. Even though, coverage of the network is reduced due to putting nodes to sleep state, but the edges between the nodes are still covered. This property will be crucial in designing the target tracking procedure.

5) Face Discovery Procedure: After the face structure is built, every node knows the information of its immediate neighbors N_{im} but is not aware of its spatial neighbors N_{sp}. Immediate neighbors are the neighbors to which a node has made edges. These are also called direct neighbors. The number of faces, the nodes is adjacent to is equal to the number of edges of nodes. All nodes in adjacent faces of a node are called its spatial neighbor nodes. To get the information of these nodes, a node needs to run a face discovery procedure [6] in which the information of all the nodes in adjacent faces is gathered. As shown in Fig. 7, node $n23$ has three adjacent faces $F8, F10, F11$ and has three immediate neighbors $n14, n24, n27$. All the nodes in adjacent faces are its spatial neighbors that are $n27, n35, n37, n38, n24, n5, n14, n13$.

B. Distributed Target Tracking Protocol

Before the target tracking is triggered, all the nodes in network that are part of face topology are in a sleep-awake periodic state also called duty cycle to conserve energy where nodes wake up periodically to check for any incoming messages and if there is no message, they go back to sleep to save energy [38]. First step as soon as, the target crosses the boundary of the network, it is detected and all the nodes in the network are in sleep-awake periodic state where they become active upon receiving a message. The nodes that detect the target, start the tracking process and the remaining nodes go back to sleep-awake periodic state.

The proposed target tracking procedure has following characteristics.

- Due to the fact that each edge is covered by two nodes, a target cannot cross an edge without being detected by the two nodes on the crossing edge.
- No prediction mechanism is required.
- Only the nodes of face in which target is located cooperate with each other to track the target.

The proposed target tracking procedure does not require prediction to track the target as the face envelops the target in which the target is located.

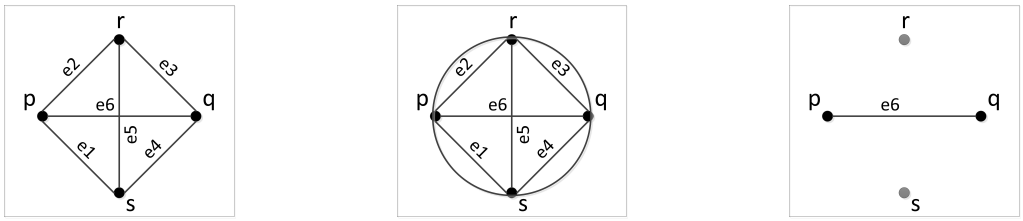
1) Target Detection: When target detection is triggered, a node becomes active and starts sensing the environment. If the target moves into the sensing range of a node it will be detected by the node. The location of target can be calculated using a distributed target localization technique [39], [40]. The nodes that do not sense any target in their vicinity, go back to default duty cycle state. The nodes that are sensing the target check if they are the nearest node to the target. Since a node has the location information of its neighbor nodes, it can calculate if it is the nearest node to the target. Among all the nodes that are sensing the target, the node that is the nearest node to the target becomes the representative/beacon node (B) and the nodes that are detecting the target but are not the nearest node go back to the default duty cycle state. A beacon node is responsible for asking the nodes to cooperatively track the target and communicating with mobile sink node.

The beacon node will detect the face in which the target is present using a face detection procedure. The face detection procedure is designed using the ”point in polygon” problem [41] from computational geometry. In this problem, it is

(a) Proposed graph

(b) Gabriel graph

Fig. 5. Condition on minimum number of edges



(a) Fully connected graph

(b) circle around e5 & e6 as diameter

(c) Proposed graph

Fig. 6. Proof that proposed face structure is planar graph

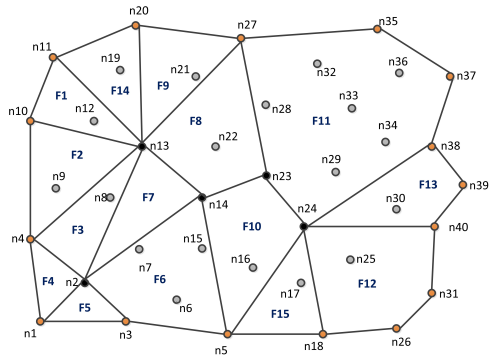


Fig. 7. Proposed Face based WSN

found if a point lies inside a polygon or not. Since a node has the information of its adjacent faces, it will take its faces as polygons and target as point and find out in which face the target is present by solving the point-in-polygon problem for each face. When the face in which the target is present is found, the target detection process is complete. Next, the beacon node will start the target tracking process by asking the nodes in face, in which target is present, to become active and start tracking the target.

Fig. 8 presents target detection algorithm where each node senses the environment and the node nearest to the target becomes beacon node and remaining nodes go back to default duty cycle state.

2) Target Tracking: Once the beacon node has detected the face in which target is present, it forwards a coop_track_packet (ctp) in the face to ask the nodes in that face to cooperatively track the target. As shown in Fig. 10a, the target is detected in face F1 by beacon node $B1$ and it sends a ctp in face F1. The ctp travels in the face using right hand rule. The nodes that receive this packet become active, update their status as coop nodes (C) and start tracking the target. A coop node is a node which tracks the target on the request of beacon node. This way the whole face has completely trapped the target and it cannot leave the face

Require: $\emptyset$
Ensure: selection of beacon node
1: **if** sensing the target is true **then**
2: **if** nearest node is true **then**
3: **if** beacon node is true **then**
4: face_detection()
5: target_tracking()
6: **else**
7: default duty cycle state
8: **end if**
9: **else**
10: default duty cycle state
11: **end if**
12: **else**
13: default duty cycle state
14: **end if**

Fig. 8. Target Detection Algorithm

without being detected.

The target is tracked by beacon and coop nodes at periodic intervals. At each interval the location of the target is taken and is stored in a list by each tracking node. The current and previous location of target forms a small segment of target trajectory and each node checks the intersection of path segment with its two adjacent edges to the face containing the target. If at any point, the target is leaving the face through an edge e, the intersection of target trajectory segment with edge e will be detected by the nodes to which that edge belongs. As shown in Fig. 10b, the target leaves the face by crossing the edge $e4$ and the intersection of target path with edge $e4$ is detected by the nodes $C3$ & $C4$.

Once the target leaving the face by crossing an edge is detected, one of the nodes of the edge that is nearest to the target will become new beacon node. The new beacon node will send a beacon_update_packet (bup) in the old face which the target left. The coop nodes upon receiving the bup packet will give-up their coop status and stop tracking the target and

```
1: while sensing the target do
2:    store the location of target in memory
3:    check for intersection with edges adjacent to current
      face
4:    if intersection is true then
5:       if nearest node to target then
6:          beacon node is true
7:          send BEACON_UPDATE_PACKET to previous
            face
8:          send COOP_TRACK_PACKET to current face
9:       end if
10:   end if
11: end while
```

Fig. 9. Target Tracking Algorithm

go back to sleep-awake periodic state. The old beacon node upon receiving the bup packet will give up its beacon status and stop tracking the target. Since the bup packet is sent by the new beacon node, it will store it as next beacon node. It will remain active for the purpose of communicating with the mobile sink node when it inquires about the target. When the sink will reach this beacon node, it will guide the sink to the next beacon node.

On the crossing edge $e4$ in Fig. 10b, among the nodes $C3$ & $C4$, $C4$ is the closest to the target, hence it becomes the new beacon node $B2$. $B2$ sends a bup packet in face F1 as shown in Fig. 10c. $B1$ will store $B2$ as next beacon node.

The new beacon node will send a coop_track_packet (ctp) to the new face in which the target is present now. The nodes in new face upon receiving ctp packet will become active, update their status as coop nodes and start tracking the target. As shown in Fig. 10d, new beacon node $B2$ sends ctp packet in face F2 and all the nodes in face F2 become coop nodes.

Fig. 9 presents target tracking algorithm. Target tracking is a process of sensing the target until it is present in the sensing range and reading its location at set intervals. While the node is sensing the target, it checks for the intersection of target trajectory with its edges adjacent to the current face. If the target crosses and edge the intersection will be detected. If the node that detected the intersection is nearest to the target, it becomes the new beacon node and sends a beacon_update_packet in the previous face to inform the nodes to stop tracking the target and a coop_track_packet in current face to ask the nodes to start tracking the target.

When a mobile sink will enter the network, it will go to first beacon node and will ask it about the location of the target, if the beacon node has target in its vicinity, it will inform the sink node about target's location else it will provide the information of next beacon node. Mobile sink will follow the trail of beacon nodes until it reaches the last beacon node which still have the target in its sensing range, hence the sink node will eventually capture the target.

C. Fault Tolerance Mechanism

When the battery of a node is depleted completely, it will die and its edges will break too causing partitioning in the network. The time of failure of a node due to battery depletion can be predicted ahead of time by monitoring battery voltage. Hence, the fault tolerant mechanism can be triggered before the complete failure of node. Since we have sleep nodes in network it is a good idea to use them to replace the faulty node. The fault tolerant mechanism makes use of the sleep nodes in network as replacement nodes to keep the network functioning for as long as possible.

Two battery threshold levels are defined, first threshold level ($TL1$) and second threshold level ($TL2$), where ($TL2 < TL1$), for complete execution of fault tolerance procedure.

When the battery level of node reaches below first threshold level ($TL1$), the nearest sleeping node from one of its faces is selected to be the replacement node and a "replacement node packet" is sent asking the sleeping node to become the replacement node. The dying node will also send the information of its edges in the "replacement node packet".

The sleeping node will wake up upon receiving the "replacement node packet" and will store the information of edges of dying node sent in the packet. It will remain awake and will wait to receive another message from the dying node.

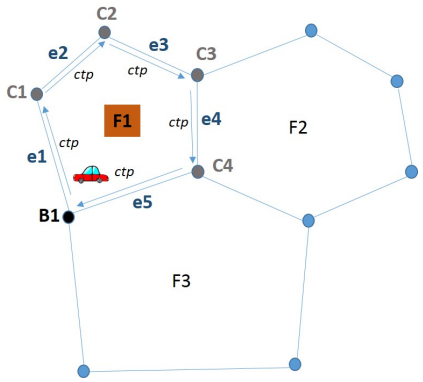
Once the battery level reaches below second threshold level ($TL2$), it break its own edges and informs its neighbor nodes to break edges with it, sends another message to the selected replacement node and then goes to sleep. The replacement node on receiving the second packet will make edges using the edge making algorithm. It will only consider the nodes with which the dead node had edges with and the information of these nodes was sent in replacement node packet.

Two battery threshold levels are used so that, a node has enough time to select a replacement node and send it the information of its edges. After the second threshold level, a node has enough time to inform its immediate neighbors to break edges.

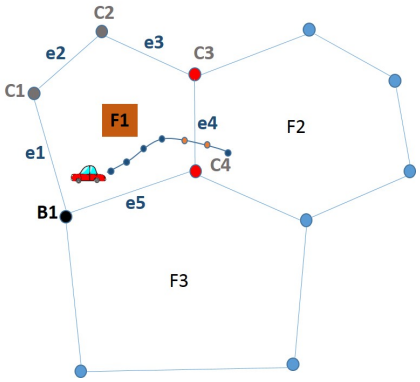
1) Replacement Node Selection: A node N_d is monitoring its battery at small time intervals Δt. When the battery level of the node reaches below $TL1$, it searches for the nearest sleeping node in its sensing range. Since each node stores the information of its neighbors in the form of a neighbor table, it sorts all the sleeping nodes from the neighbor table and finds the nearest sleeping node N_s. It is possible that a node does not have any sleeping node in its range, in that case, selection of a replacement node is not possible. Once the nearest sleeping node N_s is selected, N_d has to check if N_s lies in one of its adjacent faces. Since N_d knows the location of N_s, this is a *point in a polygon* problem, where N_s is a point and adjacent faces of N_d are polygons. In *point in a polygon* problem, it is checked if a point lies inside a polygon [41]. If N_s lies in one of the adjacent faces of N_d, it is selected as a replacement node R_p. Otherwise, N_s cannot be selected as R_p because it will not be able to make edges with edge nodes E_d of N_d.

Upon successful selection of R_p, N_d will send a "replacement_node_packet" to R_p containing the information of its edge nodes E_d. R_p will wake up on receiving this packet and stores the received data in its memory and waits for further messages from N_d.

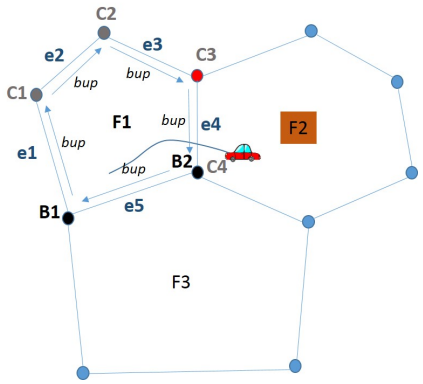
2) Edge Formation: An edge making algorithm is used by replacement node R_p to make edges. R_p has a list of nodes E_d to make edges with provided by N_d. R_p will try



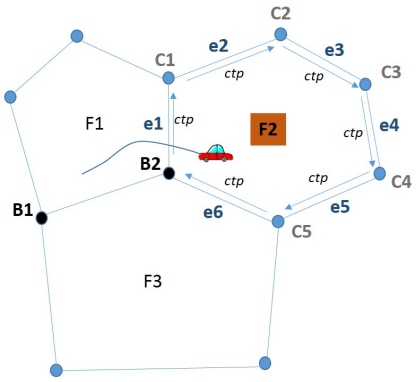
(a) coop_track_packet to face F1



(b) Intersection of target with edge $e4$



(c) beacon_update_packet to face F1



(d) coop_track_packet to face F2

Fig. 10. Target tracking scenario

to make edges with each node in the list E_d one by one.

The replacement node cannot decide the validity of edges on its own as once it was put in sleep state it had no information regarding the surrounding area or neighbor nodes. So it sends an "is_valid_edge_packet" to each node in E_d one by one. The recipient nodes will check the validity of edge with replacement node using the edge validity algorithm and will reply with an "edge_validity_reply_packet" indicating the validity of edge with R_p.

Before replying to replacement node, the recipient node runs a two-step process; 1) edge validation step and 2) edge reformation step. In the first step, it is checked if the edge is valid or not. The second step will be performed in case the edge is valid. In the second step, it is checked if some reformation of edges is required or not. We may need to reform some edges i.e. break old edges to make new edges.

a) Edge Validation Procedure: The edge validity criteria is very similar to the edge validity criteria of edges in proposed face structure i.e. an edge is valid if there is no active node that has already made edges inside the circle formed by edge as diameter.

The R_p node will ask all nodes in E_d, one by one, if the edge of R_p is valid with these nodes. The edge of replacement node R_p with $E_d[i]$ is valid if there is no node with edges inside the circle formed by the edge as diameter, otherwise, the edge with replacement node is invalid. $E_d[i]$ will check the possibility of edge with R_p because it has the information of all the nodes in neighborhood.

An example of edge validation is shown in Fig. 11. Node $n5$ asks $n6$ if the edge between $n5$ and $n6$ is valid. Node

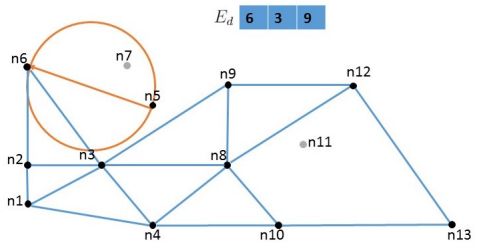


Fig. 11. Edge Validation Example

$n6$ checks if any of its edge nodes $E_d[n2, n3]$ come inside the circle formed by the possible edge as diameter. Since no nodes come inside the circle, the possible edge is valid. $n6$ replies $n5$ with the validity of the edge.

b) Edge Reformation Procedure: In case the edge between $E_d[i]$ and R_p is valid, the edge reformation step will be performed in which it will be checked if any reformation of edges is required. $E_d[i]$ checks if R_p is coming inside the circle formed by any of the edges of $E_d[i]$ with its immediate neighbors. In that case, already existing edges of $E_d[i]$ are becoming invalid, hence the reformation of edges will be required.

Before, the replacement node R_p was a sleep node and had no effect on any of the edges, but now it is active and has made edges, so the validity of other edges around it need to be checked again. If an edge of $E_d[i]$ with its immediate neighbor has become invalid, it will be broken and $E_d[i]$ and its immediate neighbor node will be connected R_p.

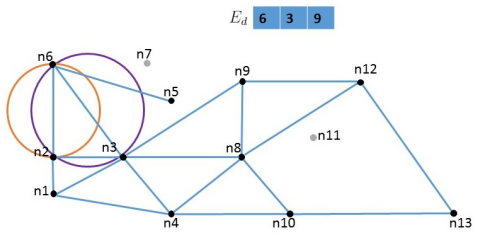


Fig. 12. Edge Reformation Example

An example of edge reformation is shown in Fig. 12. Node $n6$ after deciding that edge with replacement node $n5$ is valid, checks if any reformation of its edges is required. $n6$ has edges with nodes $n2$&$n3$. It checks if $n5$ lies inside the circle formed by any of its edges as diameter. Since, $n5$ does not lie inside any of the circles, no reformation is required.

When node $n7$ was about to die, it selected node $n5$ as replacement node. Node $n7$ had edges with nodes $n9, n3$ and $n6$. Node $n5$ successfully made edges with these nodes. $n3$ had an edge with $n8$ which broke during edge reformation and $n8$ connected to replacement node $n5$.

After edge validation and edge reformation procedures are complete, the final step is the face discovery of newly formed faces. The replacement node $n5$ will run the face discovery procedure and discover its faces, which are [$n5$, $n3$, $n6$] and [$n5$, $n3$, $n9$, $n8$]. Then $n5$ will send a face information packets in its faces to inform the nodes in faces about the discovery of new faces.

IV. Experimental Evaluation

In this section the performance of proposed distributed target tracking (DTT) protocol is compared with dynamic object tracking (DOT) protocol [27] and face track protocol [31]. The proposed work and related work have been evaluated through extensive simulations in Castalia-3.2 framework [42] in OMNeT++ v3.4 network simulator. The distributed target tracking (DTT) procedure and fault tolerance procedure is simulated on energy-based face structure (EBFS). DOT [27] and face track [31] are simulated on Gabriel-based face structure (GBFS). The performance of these approaches is compared in terms of energy consumption, lifetime and accuracy.

A. Performance Metrics

Performance is evaluated by using following performance metrics:

1) Energy Consumption: Average energy consumption in building the face topology and running target tracking procedure .

2) Lifetime of Network: The average amount of time until the first node in the network dies. [43]

3) Accuracy of Target Tracking: Tracking accuracy is determined as number of successful tracking events out of total number of tracking events.

B. Simulation Settings

The nodes are deployed in a network of 40m x 40m in 2D square plane in a random distribution manner. Face structure is built on 50, 100, 150 and 200 sensor nodes. The wireless communication links between sensor nodes are bidirectional. A target in the sensing range of node can be localized using a target localization procedure . All the nodes in network have same communication and sensing range. The communication range (r_c) of a node is set at greater than or equal to twice the sensing range (r_s) of node i.e. ($r_c \geq 2r_s$). In the simulations the sensing range of nodes is $5m$ and communication range of nodes is $10m$.

TABLE II
Number of Sleep Nodes for Different Densities

Number of Nodes (n)	Sleep Nodes	Density of Network (n/m^2)
50	9	0.03125
100	15	0.0625
150	22	0.09375
200	29	0.125

All nodes in network synchronize with the sink in first $1-10$ ms. The default speed of target is 1 m/s and default background noise is set to zero.

The simulation begins by sensor nodes gathering information of neighbor nodes. Once the information of neighbor nodes is gathered, face topology is built and face discovery procedure is run for the purpose of face recognition. Initially 100s time is set for topology construction phase and 10s is set for running face discovery procedure . At this point, the network is ready for target tracking. The target is introduced in the network at a random time and random location. The target moves in the network in a random path. Target tracking is triggered when the target is detected by the network. The experimental results are averaged over 100 simulation runs.

The initial energy of nodes is set to 50 J. For wireless communication between nodes, CC2420 radio parameters are used in simulations. CC2420 is IEEE 802.15.4 compliant RF transceiver that is low cost, low power and highly integrated solution for robust wireless applications [44]. The radio works in three modes transmit (TX), listen (RX) and sleep. In transmission mode the power consumption is 42.24 mW at transmission level of -5 dBm. In listening mode, the power consumed is 62 mW, and minimum power consumption is in sleep mode i.e., 1.4 mW.

C. Simulation Results

Once the face structure is built, some number of nodes have become asleep depending on the density of the network. Density of the network is defined as number of nodes per unit area of the network [45]. Simulations have been performed for various densities of the network as shown in table II.

1) Evaluation of Energy Consumption and Network Lifetime: The average energy consumption and average lifetime of the network for the proposed distributed target tracking (DTT) protocol is compared to the DOT [27] and face track [31] procedure as shown in Fig. 13 and Fig. 14.

The energy consumption of the distributed target tracking (DTT) is much less than face track and DOT procedure . This is mainly due to the presence of sleep nodes in the network, as nodes in sleep state consume least amount of energy. Less energy consumption means nodes will be active for longer periods of time i.e. the life time of network will

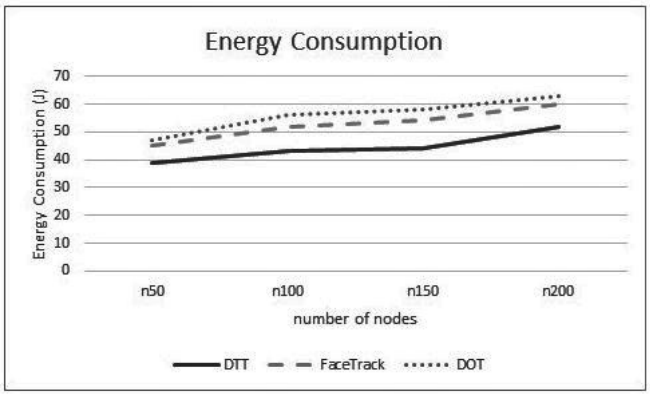


Fig. 13. Average Energy Consumption of Network

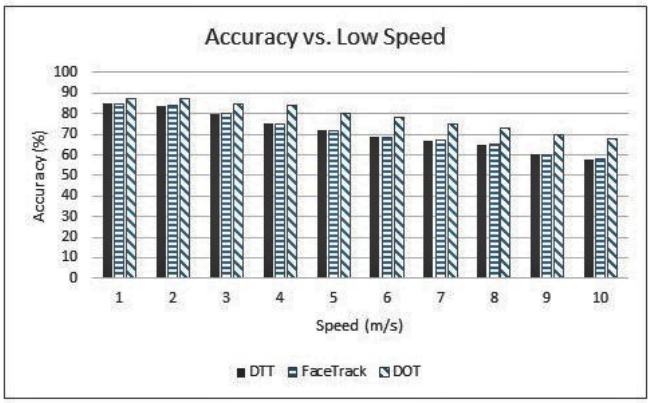


Fig. 15. Accuracy of Tracking vs. Low Speed

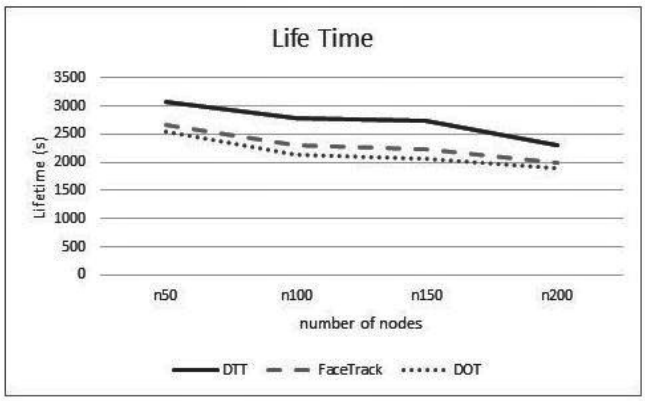


Fig. 14. Average Lifetime of Network

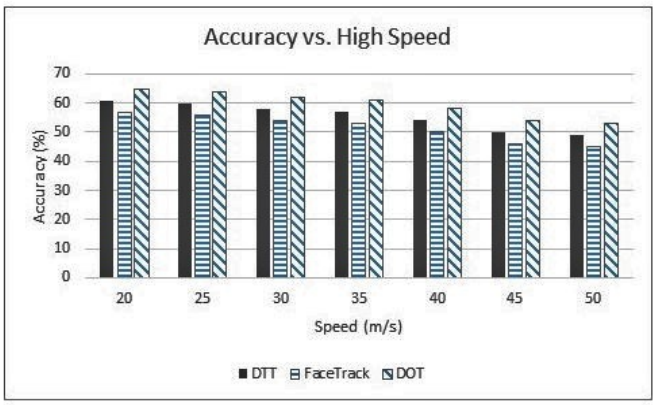


Fig. 16. Accuracy of Tracking vs. High Speed

be improved. The energy consumption of DOT procedure is higher than face track procedure due to the involvement of all spatial neighborhood nodes in target tracking as compared to only one face tracking the target in face track procedure . Both these procedures don't have any sleep nodes in the network hence the gap between proposed procedure and related procedures is larger as compared to gap between DOT and face track procedure .

2) Evaluation of Accuracy: The accuracy is defined as the number of successful target tracking events among total number of events. The total number of tracking events in each simulation is 100. Many factors can affect the accuracy of target tracking such as, the speed of target, background noise in environment and duty cycles of nodes.

a) Effect of Speed on Accuracy: The effect of speed of target on accuracy is studied at low speeds (1-10 m/s) and high speeds (20-50 m/s). At low speed, the effect on accuracy is almost the same for face track and distributed target tracking (DTT) procedure because in both of these procedures only the face in which the target resides is tracking the target as compared to DOT in which all the spatial neighborhood nodes are cooperatively tracking the target as shown in Fig. 15.

At higher speeds (20-50 m/s), a target may escape a face very quickly before being tracked by the face resulting in loss of tracking. At such speeds, the accuracy of distributed target tracking (DTT) procedure becomes better than face track procedure, because the faces in proposed face structure are generally bigger in size than Gabriel face structure. Hence, the target will need more time escaping a bigger face, as compared to small faces. The effect of high speeds of target on accuracy is shown in Fig. 16.

b) Effect of Duty Cycle on Accuracy: The duty cycles of nodes play a great part in determining the accuracy of the network. When the face topology is being built, all the nodes are fully active i.e., duty cycle of every node is 1.0. The nodes that go to sleep state, are at the duty cycle of 0.1. After the face topology is built, default duty cycle of the nodes that are part of topology can be varied from 0 to 1, to see its effect on accuracy. Once the target tracking is triggered, the nodes that are involved in target tracking are fully active and the remaining nodes are at default duty cycle to save the energy. The default duty cycle of nodes will impact the accuracy of target tracking as the nodes in next face will be sent a coop_track_packet to activate them and start tracking the target.

The trade off between accuracy and energy consumption can be achieved by setting an optimal value of duty cycle. For low duty cycles, the energy consumption will be low but the accuracy will also be less and for high duty cycles the accuracy will be high but the energy consumption will also be high.

As shown in Fig. 17, the accuracy for duty cycle of 0.1 to 0.5 is from 50% to 70%. For the duty cycle of 0.6 to 1.0 the accuracy of the tracking increases to a range of 70% to 90%. The accuracy of distributed target tracking (DTT) procedure

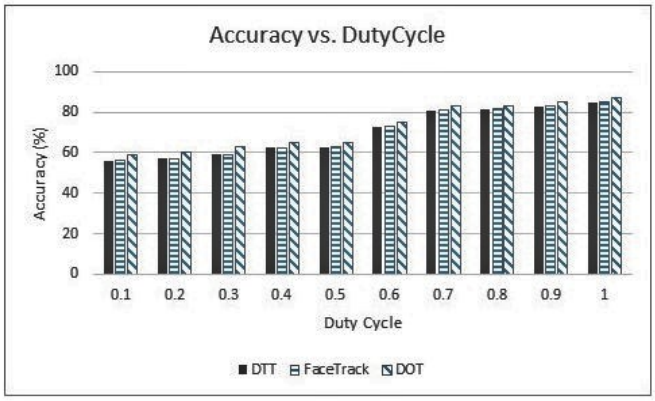


Fig. 17. Accuracy of Network vs. Duty Cycle

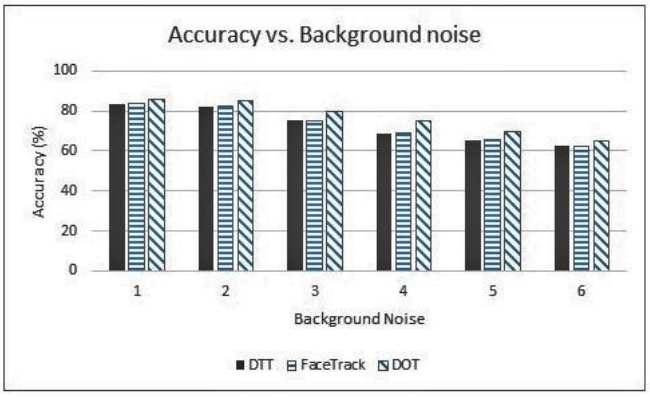


Fig. 18. Accuracy of Network vs. Background Noise

and face track procedure is extremely similar due to the fact that both these procedures use only one face at a time to track the target. The DOT procedure has higher accuracy at all duty cycles. This is mainly due to the fact that, in DOT procedure the beacon node asks all its spatial neighbors to cooperatively track the target.

c) Effect of Background Noise on Accuracy: The background noise in WSNs in practical scenarios is never zero and it affects the performance of WSNs. Messages can be lost and communication links quality can vary greatly in presence of back ground noise.

All the experiments before are performed under ideal channel conditions i.e. the background noise is zero and the communication links between nodes are perfectly bidirectional. In this experiment, the procedure is simulated under more practical conditions for various levels of noise ranging from 1 to 6. The accuracy of tracking decreases with increasing levels of noise due to the occurrence of tracking errors as shown in Fig. 18. The accuracy of DOT procedure is still higher than our proposed procedure and face track procedure because of more number of faces tracking the target at the same time.

3) Evaluation of Fault Tolerance: The fault tolerance procedure is implemented on proposed energy-based face structure. Its performance is analyzed in terms of improvement in lifetime as compared to when no fault tolerant procedure was implemented. When the fault tolerance mechanism is active in the network, the lifetime of the network improves as the node whose battery reaches a critical threshold level is replaced by selecting a nearby sleeping node as replacement node.

The first threshold level $TL1$ of node is set at 10% and the second threshold level $TL2$ is set at 5% of remaining energy. The network lifetime is increased in the range of 16% to 25% depending on the number of sleep nodes in the network. For 50 nodes the increase in lifetime is 16% and for 200 nodes the increase in lifetime is 25% as shown in Fig. 19.

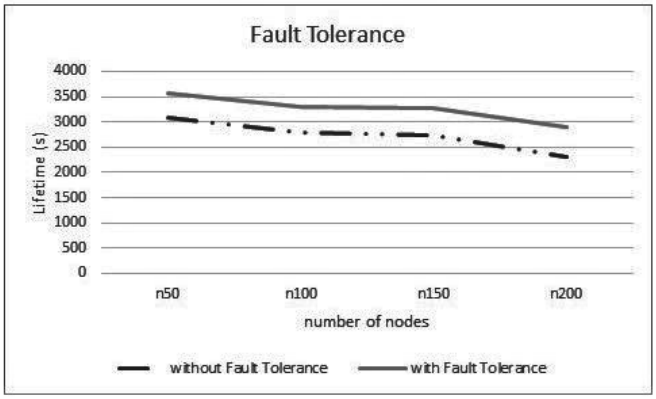


Fig. 19. Increase in Lifetime with Fault Tolerance

V. Conclusion

A new distributed fault tolerant target tracking scheme for new face-based WSN has been proposed. The proposed scheme includes a new efficient energy face structure which is better than existing one in terms of energy efficiency and lifetime of the network. The proposed face structure does not consist of all the nodes in network, instead some nodes are put in sleep state while ensuring the coverage of edges in the network. Also, proposed scheme includes a distributed target tracking algorithm is designed for proposed face structure, utilizing the property of the face structure that the edges are covered. The edge coverage property ensures that a target leaving a face is always detected. Moreover, due to the presence of sleeping nodes in the network, a distributed fault tolerant procedure is designed to use sleep nodes in network as replacement nodes in case of node failure to recover from faults. This approach maintains the network performance for as long as possible by avoiding the partitioning in the network.

References

[1] T. Takabatake, "Power consumption on topologies for a sensor-based home network," *IAENG International Journal of Computer Science*, vol. 39, no. 3, pp. 307 – 311, 2012.

[2] M. H. A. Awadalla, "Task mapping and scheduling in wireless sensor networks," *IAENG International Journal of Computer Science*, vol. 40, no. 4, pp. 257 – 265, 2013.

[3] J. Albath, M. Thakur, and S. Madria, "Energy constraint clustering algorithms for wireless sensor networks," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2512 – 2525, 2013.

[4] A. A. W. Osamy, Ahmed M. Khedr and A. El-Sawy, "Cluster-tree routing scheme for data gathering in periodic monitoring applications," *IEEE Access*, vol. 6, pp. 77 372–77 387, 2019.

[5] A. S. W. Osamy and A. M. Khedr, "An information entropy based-clustering algorithm in heterogeneous wireless sensor networks," *wireless networks, Springier*, 2018.

[6] Q. Huang, C. Lu, and G. C. Roman, "Reliable mobicast via face-aware routing," in *Proceedings - IEEE INFOCOM*, 2004.

[7] W. O. A. Salim and A. M. Khedr, "Ibleach: Effective leach procedure for wireless sensor networks," *Wireless Networks*, vol. 20, pp. 1515–1525, 2014.

[8] B. Karp and H. T. Kung, "GPSR: greedy perimeter stateless routing for wireless networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking - MobiCom '00*, 2000, pp. 243–254.

[9] A. Akl, T. Gayraud, and P. Berthou, "A metric for evaluating density level of wireless sensor networks," in *IFIP Wireless Days (WD)*, 2011, pp. 1–3.

[10] G. J. Wang, M. Z. A. Bhuiyan, J. N. Cao, and J. Wu, "Detecting Movements of a Target Using Face Tracking in Wireless Sensor Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 4, pp. 939–949, 2014.

[11] A. M. Khedr and W. Osamy, "Nonlinear trajectory discovery of a moving target by a wireless sensor network," *Journal of Computing and Informatics*, vol. 29, no. 5, pp. 1001–1016, 2010.

[12] A. M. Khedr and W.Osamy, "Effective target tracking mechanism in a self-organizing wireless sensor network," *Journal of Parallel and Distributed Computing*, vol. 71, no. 10, pp. 1318–1326, 2011.

[13] X. Ji, Y. Y. Zhang, S. Hussain, D. X. Jin, E. M. Lee, and M. S. Park, "FOTP: Face-Based Object Tracking Protocol in Wireless Sensor Network," in *Fourth International Conference on Computer Sciences and Convergence Information Technology*, 2009, pp. 128–133.

[14] F. Wang, X. Bai, B. Guo, and C. Liu, "Dynamic clustering in wireless sensor network for target tracking based on the fisher information of modified kalman filter," in *3rd International Conference on Systems and Informatics (ICSAI)*, Nov 2016, pp. 696–700.

[15] C.-Y. Lin, W.-C. Peng, and Y.-C. Tseng, "Efficient in-network moving object tracking in wireless sensor networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 8, pp. 1044–1056, 2006.

[16] E. F. Nakamura, A. A. F. Loureiro, and A. C. Frery, "Information fusion for wireless sensor networks: Methods, models, and classifications," *ACM Comput. Surv.*, vol. 39, no. 3, Sep. 2007.

[17] Y. Zhu, A. Vikram, and H. Fu, "On topology of sensor networks deployed for tracking," in *Wireless Algorithms, Systems, and Applications*, Y. Cheng, D. Y. Eun, Z. Qin, M. Song, and K. Xing, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 60–71.

[18] Y. MALEH and A. Ezzati, "Lightweight intrusion detection scheme for wireless sensor networks," *IAENG International Journal of Computer Science*, vol. 42, no. 4, pp. 347 – 354, 2015.

[19] S. Bhatti, J. Xu, and M. Memon, "Energy-aware fault-tolerant clustering scheme for target tracking wireless sensor networks," in *7th International Symposium on Wireless Communication Systems*, 2010, pp. 531–535.

[20] M. B. L. de Souza, H. Vogt, "A survey on fault tolerance in wireless sensor networks," 2007. [Online]. Available: http://www.cobis-online.de

[21] I. Banerjee, P. Chanak, H. Rahaman, and T. Samanta, "Effective fault detection and routing scheme for wireless sensor networks," *Computers & Electrical Engineering*, vol. 40, no. 2, pp. 291 – 306, 2014.

[22] M. K. Watfa and R. A. Assi, "Daim: A distributed algorithm for isolating malfunctioning nodes in wireless sensor networks," in *Advances in Education and Management*, M. Zhou, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 523–530.

[23] G. T. Toussaint, "The relative neighbourhood graph of a finite planar set," *Pattern Recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[24] A. Razzaq, A. M. Khedr, and Z. Al Aghbari, "A redundancy-aware face structure for wireless sensor networks," in *2018 8th International Conference on Computer Science and Information Technology (CSIT)*, July 2018, pp. 38–42.

[25] M. Z. A. Bhuiyan, G. Wang, and A. V. Vasilakos, "Local Area Prediction-Based Mobile Target Tracking in Wireless Sensor Networks," *IEEE Transactions on Computers*, vol. 64, no. 7, pp. 1968–1982, 2015.

[26] T.-S. Chen, J.-J. Chen, and C.-H. Wu, "Distributed object tracking using moving trajectories in wireless sensor networks," *Wireless Networks*, vol. 22, no. 7, pp. 2415–2437, 2016.

[27] H.-W. Tsai, C.-P. Chu, and T.-S. Chen, "Mobile object tracking in wireless sensor networks," *Computer Communications*, vol. 30, no. 8, pp. 1811 – 1825, 2007.

[28] M. Z. A. Bhuiyan, G.-j. Wang, L. Zhang, and Y. Peng, "Prediction-based energy-efficient target tracking protocol in wireless sensor networks," *Journal of Central South University of Technology*, vol. 17, no. 2, pp. 340–348, 2010.

[29] J.-M. Hsu, C.-C. Chen, and C.-C. Li, "POOT: An efficient object tracking strategy based on short-term optimistic predictions for face-structured sensor networks," *Computers & Mathematics with Applications*, vol. 63, no. 2, pp. 391–406, 2012.

[30] W. M. Elsayed, S. F. Sabbeh, and A. M. Riad, "A Distributed Fault Tolerance Mechanism for Self-Maintenance of Clusters in Wireless Sensor Networks," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 6891–6907, 2018.

[31] G. Wang, M. Bhuiyan, J. Cao, and J. Wu, "Detecting movements of a target using face tracking in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 4, pp. 1–11, 2014.

[32] M. Z. A. Bhuiyan, G. Wang, and J. Wu, "Target Tracking with Monitor and Backup Sensors in Wireless Sensor Networks," in *Proceedings of 18th International Conference on Computer Communications and Networks*, 2009, pp. 1–6.

[33] R. Zhao and X. Shen, "Broadcast protocols for wireless sensor networks, smart wireless sensor networks," http://www.intechopen.com/books/smart-wireless-sensor-networks/broadcast-protocols-forwireless-sensor-networks, InTech China, 2010.

[34] A. M. Khedr and W. Osamy, "Minimum perimeter coverage of query regions in a heterogeneous wireless sensor network," *Information Sciences*, vol. 181, no. 15, pp. 3130–3142, aug 2011.

[35] A. M. Khedr, "Location-free minimum boundary coverage in a wireless sensor network," *Procedia Computer Science*, vol. 65, pp. 48–57, 2015.

[36] R.-H. Hwang, C.-C. Wang, and W.-B. Wang, "A distributed scheduling algorithm for ieee 802.15.4e wireless sensor networks," *Computer Standards & Interfaces*, vol. 52, pp. 63 – 70, 2017.

[37] P. Chaturvedi and A. K. Daniel, "A hybrid scheduling protocol for target coverage based on trust evaluation for wireless sensor networks," *IAENG International Journal of Computer Science*, vol. 44, no. 1, pp. 87 – 104, 2017.

[38] R. R. Rout and S. K. Ghosh, "Enhancement of lifetime using duty cycle and network coding in wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 12, no. 2, pp. 656–667, February 2013.

[39] A. Hajihoseini Gazestani, R. Shahbazian, and S. A. Ghorashi, "Decentralized Consensus Based Target Localization in Wireless Sensor Networks," *Wireless Personal Communications*, vol. 97, no. 3, pp. 3587–3599, 2017. [Online]. Available: https://doi.org/10.1007/s11277-017-4687-0

[40] R. Shahbazian and S. A. Ghorashi, "Distributed cooperative target detection and localization in decentralized wireless sensor networks," *The Journal of Supercomputing*, vol. 73, no. 4, pp. 1715–1732, 2017.

[41] D. Sunday, "Inclusion of a point in a polygon," http://geomalgorithms.com/a03-_inclusion.html.

[42] A. Boulis, "Castalia framework," https://github.com/boulis/Castalia, 2007.

[43] Y. Chen and Q. Zhao, "On the lifetime of wireless sensor networks," *IEEE Communications Letters*, vol. 9, no. 11, pp. 976–978, Nov 2005.

[44] T. Instruments, "Cc2420," http://www.ti.com/product/CC2420.

[45] S. Toumpis, "Mother nature knows best: A survey of recent results on wireless networks based on analogies with physics," *Comput. Netw.*, vol. 52, no. 2, pp. 360–383, Feb. 2008.