Detecting Movements of a Target Using Face Tracking in Wireless Sensor Networks

Guojun Wang, *Member, IEEE*, Md Zakirul Alam Bhuiyan, *Member, IEEE*,
Jiannong Cao, *Senior Member, IEEE* , and Jie Wu, *Fellow, Member IEEE*

Abstract—Target tracking is one of the key applications of wireless sensor networks (WSNs). Existing work mostly requires organizing groups of sensor nodes with measurements of a target's movements or accurate distance measurements from the nodes to the target, and predicting those movements. These are, however, often difficult to accurately achieve in practice, especially in the case of unpredictable environments, sensor faults, etc. In this paper, we propose a new tracking framework, called FaceTrack, which employs the nodes of a spatial region surrounding a target, called a *face*. Instead of predicting the target location separately in a face, we estimate the target's moving toward another face. We introduce an edge detection algorithm to generate each face further in such a way that the nodes can prepare ahead of the target's moving, which greatly helps tracking the target in a timely fashion and recovering from special cases, e.g., sensor fault, loss of tracking. Also, we develop an optimal selection algorithm to select which sensors of faces to query and to forward the tracking data. Simulation results, compared with existing work, show that FaceTrack achieves better tracking accuracy and energy efficiency. We also validate its effectiveness via a proof-of-concept system of the Imote2 sensor platform.

Index Terms—Wireless sensor networks, target tracking, sensor selection, edge detection, face tracking, fault tolerance

1 Introduction

Wireless sensor networks (WSNs) have gained a lot of attention in both the public and the research communities because they are expected to bring the interaction between humans, environments, and machines to a new paradigm. WSNs were originally developed for military purposes in battlefield surveillance; however, the development of such networks has encouraged their use in healthcare, environmental industries, and for monitoring or tracking targets of interest [1], [2].

Fig. 1 illustrates a typical scenario of an enemy vehicle tracking application. Sensor nodes are informed when the vehicle under surveillance is discovered, while some nodes (such as black nodes) detect the vehicle and send a vigilance message to the nodes on the vehicle's expected moving path, so as to wake them up. Thus, the nodes (such as grey nodes) in the vehicle's moving path can prepare in advance and remain vigilant in front of it as it moves. To be energy efficient and to accurately track the vehicle, only the nodes close to the path can participate in tracking and providing continuous coverage.

Regardless of various types of targets , there are three common procedures involved in existing solutions of target tracking [1], [2], [3], [4], [5], [6], [7], [8], [9], [10]: 1) sensor nodes should be localized with as few errors as possible, and a distance measurement from the nodes to a target, or a measurement of the target's movements is crucial; 2) nodes should be organized into groups (e.g., clusters) to track a mobile target; 3) leader sensors generally report the target's movement to a central sink (or a user)—the sink is a resource-rich node that gathers information from the leaders [1].

Regarding these procedures above, if we want to work with scenarios like that of Fig. 1, achieving high accuracy of tracking together with energy efficiency in WSNs is a challenging problem, due to several apparent difficulties:

- Organizing groups of nodes with accurate measurements of a target's movements is difficult, as WSNs are dense/sparse, unattended, untethered, and deployed in usually unpredictable environments.
- Obtaining accurate target localization is impossible in a real operation field, even when different kinds of noises/disturbances are added during detection.
- Maintaining operations of nodes in a timely fashion is difficult, i.e., turning their services off most of the time, and enabling only a group of nodes to be functional in the target's moving path, as in Fig. 1.
- Loss of tracking or node failure is often possible, since WSNs are prone to fault or failure.

Research about target tracking can be roughly divided into three categories: 1) tree-based schemes [4]; 2) cluster-based schemes [5]; and 3) prediction-based schemes [11]. In this paper, we propose FaceTrack, a framework to detect movements of a target using face tracking in a WSN, which does not fall into existing categories and is, to the best of our knowledge, the first of its kind. The

- *G. Wang is with the School of Information Science and Engineering, Central South University, Changsha 410083, China. E-mail: csgjwang@mail.csu.edu.cn.*
- *M.Z.A. Bhuiyan is with the School of Information Science and Engineering, Central South University, Changsha 410083, China, and with the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong. E-mail: zakirulalam@gmail.com.*
- *J. Cao is with the Department of Computing, The Hong Kong Polytechnic University, Kowloon, Hong Kong. E-mail: csjcao@comp.polyu.edu.hk.*
- *J. Wu is with the Department of Computer and Information Sciences, Temple University, Philadelphia, PA 19122. E-mail: jiewu@temple.edu.*

Manuscript received 3 Aug. 2012; revised 10 Mar. 2013; accepted 13 Mar. 2013; date of publication 26 Mar. 2013; date of current version 21 Feb. 2014.
Recommended for acceptance by A. Nayak.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPDS.2013.91

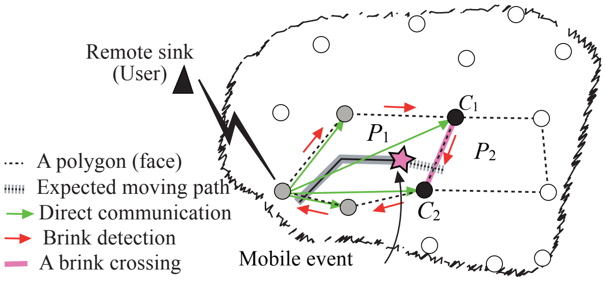

Fig. 1. An example application with a sink showing a vehicle being tracked through a polygonal-shaped area.

concept of FaceTrack is depicted in Fig. 1, and is inspired by computational geometry, geographic routing, and face routing, in particular [12], [13]. FaceTrack mitigates the discussed difficulties, when satisfying our objectives—achieving tracking ability with high accuracy and reducing the energy cost of WSNs.

Main concept. The idea of a planarized graph, such as the Voronoi diagram and related neighborhood graph (RNG), is mostly used in the network domain [14]. In such a graph, a plane with p points is partitioned into spatial non-overlapping regions, known as *faces*, by using the term in a face routing strategy [12], [13], such that each face contains some of the points that are connected. Every two points of a face share an edge that is also a common edge between two neighboring faces. A target is assumed to be surrounded by the perimeter of ith face P_i , e.g., the target lying inside P_1, as shown in Fig. 1, can be detected as it goes across an edge/link (such as, $<C1, C2>$) toward P_2. The two points (e.g., the black nodes in Fig. 1) become *couple nodes* chosen from all of the points (neighboring nodes), through a selection process to lead the tracking the target from P_i to P_j.

Normally, the faces can be of different sizes and geometrical polygonal-shaped forms in the WSN. For the sake of simplicity, we call them *polygons* throughout this paper. Initially, a complete WSN is generated by using the planarization algorithm where inter-node edges of polygons are identified logically. However, when a target moves from P_i toward P_j, the generated polygons may not be preserved, due to cases of failure-prone WSNs, unpredictable environments, the presence of void regions, etc. Therefore, we need to verify the nodes and edges of P_j, like generating the polygons further.

We introduce an edge detection algorithm to generate jth polygon P_j further, in such a way that the nodes of P_j can wake up and prepare before the target moves to P_j , which greatly helps track the target in a timely fashion. In this case, the common edge between P_i and P_j (i.e., the target is about to cross) is called a *brink* and the end nodes are the couple nodes. Detecting a brink is a way of making a rectangular/square space around the target as it moves toward the brink. The space could be called a 'follow spot,' much like a spotlight follows and moves with a musician during a concert. It is easy to think about the moving spotlight from the 'space + time' point of view. This idea provides natural support for target information dissemination that exhibits 'right-place' and 'right-time' semantics, including the 'just-in-time' requirement [15], [16].

Benefits. Some benefits of FaceTrack are highlighted, which help us reach our objectives. 1) When an event of sensor fault occurs, or there is an event of loss of tracking, FaceTrack mitigates such events without recalibrating the whole network. 2) Nodes locally detect the presence of the target and decide whether to continue tracking tasks, i.e., they do not need to communicate with the sink frequently. However, the sink is informed by the couple nodes whether or not the target enters the surveillance area. 3) Nodes do not predict or maintain the target's movement history completely, but keeps only the most recently reported information and time instance. 4) If the number of active nodes is large, the tracking accuracy is higher, but the energy cost is higher too. FaceTrack relies on accumulated detection from a selected number of nodes that are in the polygons.

A preliminary version of this work appeared in [17]. The four main contributions of this paper are as follows:

- We design FaceTrack, a new tracking framework that detects the movements of a target using polygon (face) tracking, inspired by the planarized algorithm, which does not rely on any global topology.
- We develop a brink detection algorithm that enables the WSN to be aware of a target entering the polygon a bit earlier, and to work in a timely fashion.
- We formulate an optimal selection algorithm to select couple nodes on the target's moving path to keep the number of active sensors to a minimum.
- We evaluate the performance of FaceTrack extensively through simulations and compare with existing solutions [3], [6], [18]. We also present a proof-of-concept implementation of this design using the TinyOS [19] on the Imote2 platform [20] and deploy it in an outdoor environment. The results show that FaceTrack has the ability to track a target with high accuracy and reduces the energy cost of WSNs.

The rest of this paper is organized as follows. Related work is in Section 2. Section 3 explains preliminaries and models. Section 4 presents the design of FaceTrack. Target detection through polygon tracking is in Section 5. The performance is analyzed in Section 6. Simulation and experimental evaluation are in Sections 7 and 8, respectively. Finally, Section 9 concludes this paper.

2 Related Work

For a comprehensive discussion of the related work, please refer to Appendix A.[1]

3 Preliminaries and Models

In this section, we first present the objectives of FaceTrack. Then, we briefly discuss the preliminaries and introduce the system models.

3.1 Objectives

The objective of this paper is to design FaceTrack to achieve an efficient and real-time tracking through detecting the movement of a target using face tracking. To measure the performance of FaceTrack, two of the important metrics are

[1] Appendices are attached to the supplementary file of this paper, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPDS.2013.91.

TABLE 1
Mathematical Notations

Symbol	Description
P_c	Active/current polygon (where a target is now in)
P_f	Forward/future polygon (where a target is moving to)
P_i	The number of neighboring polygons (PI) of a node
P_N	The number of sensor nodes (PNs) in a polygon
N_N	The number of neighboring nodes (NNs) in a polygon
O_N	The optimal number of sensor nodes (ONs)
C_N	Couple nodes (CNs)
D	Brink Length
c_v	Localization error covariance
d_{ik}	Distance between two sensor nodes
d_{ij}	Distance between a sensor node and a target
r_s	Sensing range
r_c	Communication range
T	Given time window of the whole measurement period
t	Time window for an observation or a track

as follows: 1) tracking accuracy: decreasing tracking errors found (TEF) by nodes that are involved in tracking and increasing tracking ability rate (TAR), i.e., the degree of successful tracking; 2) energy cost and energy- efficiency of the WSN.

3.2 Assumptions and Notations

Some of basic assumptions of FaceTrack are as follows:

- The mobile target (event) that is of interest is sensed and optionally observed by a WSN, such as tracking an enemy vehicle, an intruder, or a moving wild animal [2], [7]. We consider a single target, i.e., a vehicle is being tracked in the WSN with a maximum off/on-road speed of around 10 m/s.
- Sensors are assumed to be homogeneous. The sink is assumed to be a user, where the system is controlled.
- All nodes are synchronized and follow a state transition policy to be active/inactive, as detailed in [21].
- The WSN is assumed to have some faulty/damaged nodes. It is randomly set after initialization.
- If a target is detected by a node after a time window, a target is detected by another node. It is assumed to be the same target. This assumption is made because the target does not carry any form of classification [3], nor can any different target be distinguished.

Table 1 gives the mathematical notations that are used throughout this paper.

3.3 Network Model

We consider a WSN $G = (V, E)$ composed of a set V of N nodes and a set E of edges in a 2D planar field, and the nodes are able to tune their range up to radio range r_c. Let $N(u) = \{v \| (u, v)| \leq r_c\}$ be the set of neighbors of node u, and there is a sink or user in the WSN that requires information about a target. Consequently, all $u \in V$ and $v \in V$ together define a unit disk graph (UDG), which has an edge (u, v) if, and only if, the Euclidean distance $\|(u, v)\| \leq 1$. To track the target route, extracting planar graphs is needed to guarantee the information delivery before the target arrives at a region [12].

Related neighborhood graph is an example algorithm that creates a planar graph [13], [22], [23]. The main idea is that two nodes, u and v, from a planar graph, are within each other's communication range, if there is no other neighbor, w, called a *witness*, within their common area that is closer to either u or v. We can obtain a connected planar subgraph $G' = (V, E')$ that maintains connectivity with fewer edges. The planar subgraph contains one or more closed *polygons* (or faces). Such a polygon contains at least three nodes. A polygonal region is a topological concept that can be defined abstractly, without use of exact coordinates (a detailed description of this network model and initial polygon construction and its limitations is in Appendix B.1, available in the online supplemental material.)

3.3.1 Distributed Measurement Model

Consider a target moving, e.g., a vehicle, in a restricted area $(\mathbb{R})$, and its movement is detected by a WSN. The target may accelerate or decelerate at any time. Let $s_i \in \mathbb{R}$ be the location of the ith node, and $L_i = \{s_i : 1 \leq i \leq N\}$. The target always emits a signal that is unidirectional and can be detected by the node in its sensing range, r_s.

We model the sensor measurement problem by using a standard estimation theory [24]. In this framework, all sensors are acoustic, measuring only the amplitude of the sound signal. Let $e_s(t)$ be the time-dependent average signal energy measurements over t, then a sensor can make the following measurement:

$$e_s(t) = S_i(t) + \varepsilon_i(t) \tag{1}$$

where $S_i(t)$ is the signal and $\varepsilon_i(t)$ is the noise energy, respectively. The background noise has a distribution with the mean, which is equal to σ_i^2, and the variance, which is equal to $2\sigma_i^2/M$. M can be larger, for example, 40. In FaceTrack, the brink detection depends on the target's location. To estimate the brink, the location information is estimated with an adjustment on error covariance, c_v. We adjust the approximate target location information by using a covariance bound that is similar to the formulation of the *Cramer-Rao lower bound (CRLB)* of the variance [9], [11]. See Appendix C, available in the online supplemental material, for the remaining part of explanation of this measurement model and localization error adjustment.

4 Design of FaceTrack

We first define how the polygons can be localized in FaceTrack. Then, we present our brink detection algorithm. At last, we introduce our optimal node selection algorithm and its features.

4.1 Localized Polygon

In order to describe the problem of detecting the movement of a target as an unauthorized target traversal problem through polygon tracking, we see an example of the generated polygons as shown in Fig. 2. We use polygons to describe the target moving path. The polygon is not necessarily a convex, but it must not be self-overlapping. Let a number of nodes in a polygon be $P_N = (v_1, v_2, \ldots, v_p)$, where $p \geq 3$. Suppose that the target is detected by some nodes somewhere in the WSN, and it is surrounded by the nodes in a polygon, e.g., P_2. Then, P_2 is called an *active polygon* (P_c), and nodes (e.g., v_5) in P_2 are *active nodes*. In Fig. 2, P_1 is a triangle, P_2 is a pentagon, and P_7 is a tetragon. Node

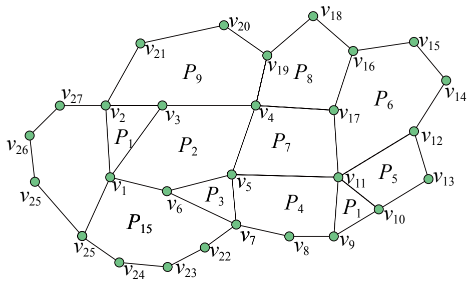

Fig. 2. An example of the sensor network, demonstrating polygonal shaped regions (or faces).

v_5 in P_2 is aware of the following information: 1) its own information; 2) the information of its adjacent (or 1-hop) neighbors v_4, v_{11}, v_7, and v_6; 3) the information of its active neighboring nodes v_6, v_1, v_3, and v_4; 4) the information of the neighbors in P_2, P_3, P_4, and P_7 through direct communication or the 1-hop intermediate nodes after deployment. Thus, v_5 stores information about four polygons that are adjacent to it in G–$\{v_5, v_4, v_{17}, v_{11}\}$, $\{v_5, v_{11}, v_{19}, v_8, v_7\}$, $\{v_5, v_7, v_6\}$, and $\{v_5, v_6, v_1, v_3, v_4\}$.

The target may move from P_c to any of the adjacent polygons, e.g., P_7. The adjacent polygon is called a *forward polygon* (P_f). v_5's *adjacent neighbors* that correspond to P_c, with respect to the target detection, are called *immediate neighbors*. Thus, node v_5 can have only two immediate neighbors, v_4 and v_6, out of the four adjacent neighbors in G. Either v_4 or v_6 becomes active as the target crosses edge (v_5, v_4) or edge (v_5, v_6). Suppose the target travels toward polygon P_7; it crosses edge (v_5, v_4), thus, we call v_5 and v_4 *couple nodes* (CNs). The process of selecting the couple nodes is described in a later section. All of v_5's neighboring nodes in P_2 are denoted by NNs. The working area of v_5 covers all of the edges between the adjacent neighbors and itself. Thus, a node corresponds to a number of polygons (P_i) that depends on the number of edges or adjacent neighbors. The size of a polygon is defined by the number of edges surrounding the polygon. The average size of a polygon is $\overline{P} \leq 2v_i/(v_i - e_i + 2)$, where v_i and e_i are the numbers of nodes and edges of the polygon, respectively. The relationship between nodes, edges, and polygons is given as $P_i + v_i - e_i = 2$, where P_i is the number of polygons corresponding to a node according to Euler's formula [25]. This implies that FaceTrack has cells for a planarized WSN, with as many edges as possible.

Some observations on underlying issues/advantages of this localized polygon are discussed in Appendix B.2, available in the online supplemental material. We provide a representative example in Appendix D, available in the online supplemental material, which elaborates two important concerns: 1) how does the system detect the target in a polygon in the beginning; 2) which polygon is the target moving toward.

4.2 Brink Detection Algorithm

We introduce an edge detection algorithm, which is used to reconstruct another conceptual polygon, called a *critical region*, by generating an edge, called a *brink*, to the active polygon, P_c. As the brink is generated on the boundary of P_c, the polygonal region problem turns into a *critical region* problem. In the algorithm, our objective is to detect the brink, while the target is moving to a brink between CNs, that confirms that the target is leaving P_c and moving to P_f, which could allow for tracking the target in a timely fashion. As explained in Appendix D, available in the online supplemental material, after the detection of the target and the reconstruction of P_c around the target, this algorithm is applied during the target movement from P_c to P_f.

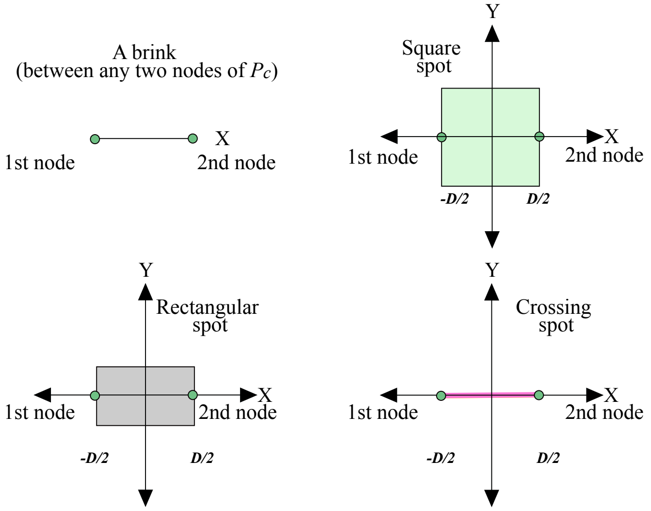

Fig. 3. Three-phase detection spots, where the *X*-axis shows the brink crossing.

In the algorithm, the edges of P_c are mapped by the brinks. As the target moves to a brink, the target is focused on a spot, called a *follow spot*. In the follow spot, a brink between CNs can be similar to an 'automatic door.' Often found at supermarket entrances and exits, an automatic door will swing open when it senses that a person is approaching the door. The door has a sensor pad in front to detect the presence of a person about to walk through the doorway. Therefore, the door can be called an *entrance door* or *entrance brink*.

When a person accesses the entrance sensing area, the door opens; however, if the person does not pass through the door and waits in front, the door is closed automatically after a period of time. Hence, in the case that the waiting period occurs in the algorithm, the CNs do not need to broadcast the message to P_f. Suppose that the person/target passes toward the door/brink from P_c to P_f. As the target moves toward a brink of P_c, the follow spot is divided into the following three-phase detection spots (see Fig. 3 for the three phases and Appendix E, available in the online supplemental material, for more details):

- *Square detection phase.* This implies that the target is preliminarily detected by any two nodes inside P_c but does not guarantee that the target may cross the brink between them.
- *Rectangular detection phase.* This implies that the target may cross the brink between the nodes.
- *Crossing phase.* This implies that the target is about to cross the brink between the nodes.

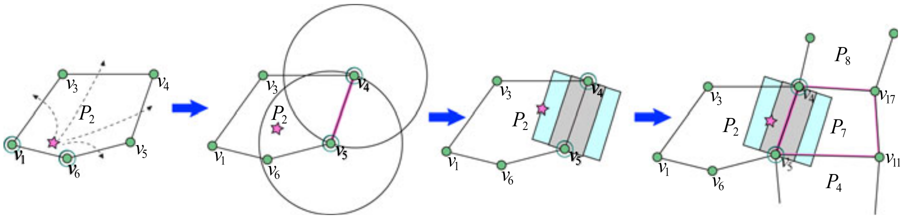

Fig. 4. A simple scenario of the brink detection process.

By using the three-phase detection, each brink in P_f has to be identified during the target's crossing over, as shown in Fig. 4. To estimate the phases, we consider the brink to be mapped over the X-axis, as shown in Fig. 3. Let D be the length of the brink, and let i and k be the couple nodes, respectively. We assume that $D \propto d_{ik}$ and $\frac{D}{2} \leq r_s$. D is achieved from $(-D/2)$ to $(D/2)$. $D \leq 2r_s$ is a length of both square and rectangular spots. Hence, $\mathrm{A} = D^2$ is for the total square spot, and $\mathrm{A} = \frac{D}{2} \times D$ is for the total rectangular spot.

Suppose that the target is in the square phase. When it touches the rectangular phase, a joint-message is broadcast to P_f. When the target passes the crossing phase, P_f becomes the new P_c. All of the brinks in the previous P_c are removed, and the previous P_c becomes inactive and remains as a neighboring polygon. Variability of different parameters of the brink, i.e., 1) brink length, 2) local mean length, and 3) local standard deviation, allow the CNs to identify the brink more easily.

Let ρ, $\rho^{'}$, and $\rho^{''}$ be the detection probability for the three phases, respectively, by the closest sensor that is one of the CNs , which are expressed as:

$$\begin{aligned}\rho &= \frac{1}{A}\int_{-D/2}^{D/2} e_s(CN,j)dx \int_{-D/2}^{D/2} dy,\\ \rho' &= \frac{1}{A}\int_{-D/2}^{D/2} e_s(CN,j)dx \int_{-D/4}^{D/4} dy,\\ \rho'' &= \frac{1}{A}\int_{-D/2}^{D/2} e_s(CN,j)dx \int_{-D/8}^{D/8} dy.\end{aligned} \tag{2}$$

Note that the values of ρ, $\rho^{'}$, and $\rho^{''}$ completely rely on the length of the brink. In order to detect the target specifically, a node should satisfy two conditions: 1) the node must be in an active polygon; 2) the node must be in the active state when the target passes through the brink along its sensing range. The detection probability of the closest sensor to a target totally depends on the length of the brink and an intersection of the sensors' sensing range. As the brink lies along the X-axis, and has a length of D, the different random values of intersecting nodes should be in a range of $(-D/2)$ to $(D/2)$. Thus, the derivation in the above can be justified easily.

4.3 Optimal Node Selection Algorithm (O_N)

Generally, tracking a target requires an optimal number of sensors in the network to aggregate data among the sensors. With FaceTrack, among the available sensors in a polygon, not all of the sensors provide useful information that improves accuracy. Particularly, if the number of sensors in a polygon is large, we need to minimize the number of active sensors. Furthermore, some information might be useful, but redundant.

We offer an optimal selection mechanism to choose the appropriate sensors, which can result in having the best detection and a low energy cost for transmitting data across the polygon; this also saves both power and bandwidth costs. We have already described a localized polygon mechanism, and the idea of routing without knowing global knowledge about sensor locations. A selection function is utilized to select the appropriate sensors on the target's moving path, and is based on the local decisions of all of the sensors in a polygon.

After the brink is formed between the CNs, the nodes query and send a message to all of the neighbors (NNs) that correspond to the forward polygon. The message contains the estimation of the target and sender information. While receiving the message, each NN combines its own measurements of the target with the CNs' estimation. Each NN computes its weight and checks whether it is about to be a CN by using a selection function; then, the NN responds to the previous CNs via a bid (e.g., ID, d_{ij}, etc.). When a node detects the target, it sends the bid to its immediate neighbors. It also receives a similar bid from the neighbors if both of its immediate neighbors detect the target, which then evaluates the received bids and ranks them according to the weight of the bids. Then, the node compares the weight of the bids with its own bid, and ranks them. It locally decides whether it should join in tracking, or withdraw itself from the tracking. If it has the "best" weight, it can easily determine its CNs from the rank. In this way, we can select the best nodes on the target's target moving path as the CNs. We use the selection function as a mixture of both detection information and the energy cost [11]. Suppose that the number of optimal nodes is $O_N(\leq P_N)$, the selection function is stated as:

$$\begin{aligned}\psi(\delta(\overline{x}|N_N, C_N)) = \alpha * \lambda_{use}(\delta(\overline{x}|N_N, C_N))\\ - (1-\alpha) * \gamma_{\text{cost}}(N_N).\end{aligned} \tag{3}$$

We describe the function as follows:

- $\delta(\overline{x}|N_N, C_N)$ is the estimate of the target, formed by each node and NNs.
- $\lambda_{use}(\delta(\overline{x}|N_N, C_N))$ is the information usefulness measurement function given as:

$$\begin{aligned}\lambda_{use}(\delta(\overline{x}|N_N, C_N)) &= \lambda_{use}(x_i, \overline{x})\\ &= (x_i - x)^T c_v,\end{aligned} \tag{4}$$

where x_i is the location vector of the ith sensor node and $\overline{x}$ is the location vector of the target that is estimated by the ith sensor node and one of the CNs.

- $\gamma_{\text{cost}}(N_N, C_N)$ is a function that refers to the energy cost of communications between NNs and previous CNs; thus, the geometric measure of the function is given as:

$$\gamma_{\text{cost}}(N_N, C_N) = (x_i - x_c)^T (x_i - x_c), \tag{5}$$

where x_c is the location of a CN and $(x_i - x_c)$ is the distance between the neighbors and one of the CNs.

- α is the relative weight of the usefulness and cost.

Finally, the selection function, (3), can be reduced by substituting (4) and (5) as follows:

$$\psi(\delta(x_c, x_i, \overline{x})) = \alpha * (x_i - x)^T c_v - (1 - \alpha) * (x_i - x_c)^T (x_i - x_c). \tag{6}$$

This function only relates with CNs, NNs, and the target's locations: x_c, x_i, and $\overline{x}$. The number of NNs in a polygon that will become the optimal nodes is based on a parameter, i.e., a threshold denoted by $N\text{th}$ of the selection function. The threshold is defined by a value in which the number of optimal nodes chosen should be no more than $N\text{th}$. If $N\text{th} < 1$, all of the bids are chosen. However, we accept $O_N = 2$ for this tracking framework, where $O_N = mod(N\text{th})$, i.e., selecting CNs. The benefits, and more details of this algorithm, can be found in Appendix F, available in the online supplemental material.

The optimal number of sensors can be many, which depends on the system demand $(O_N \leq P_N)$. It is expected that the chosen optimal node number should be no more than the number of P_N. According to different tracking tasks, O_N can be changed by the sink broadcasting a message containing O_N to the network. Nevertheless, the optimal selection is very important, which not only impacts the tracking accuracy, but also the energy efficiency of the WSN. The overall system can exceptionally benefit from using network resources by using this algorithm. When the number of P_N in a polygon is large, the normal tracking goes on to only select CNs. If $P_N = 3$, i.e., the polygon is a triangle, one of the nodes of the polygon serves two terms as a CN.

5 Movement Detection through Polygon Tracking

In this section, we provide an overview of target detection through the polygon tracking process. We also discuss the fault tolerance in the WSN during tracking.

5.1 Overview of the Polygon Tracking Process

The framework of the tracking in FaceTrack is shown in Fig. 5. There are five steps in the framework. The Step 1 is about the system initialization, including initial polygon construction in the plane. A node has all of the corresponding polygons' information after the WSN planarization. Initially, all nodes in the WSN are in a low-power mode [21] and wake up at a predefined period to carry out the sensing for a short time. As described in Appendix G, available in the online supplemental material, we presume that a sensor node has three different states of operation, namely, *active* (when a node is in a vigilant manner and participates in tracking the target), *awakening* (when a node awakes for a short period of time), and *inactive* (when a node is in a sleeping state). We consider that a sensor should be kept awake so long as its participation is needed for a given task.

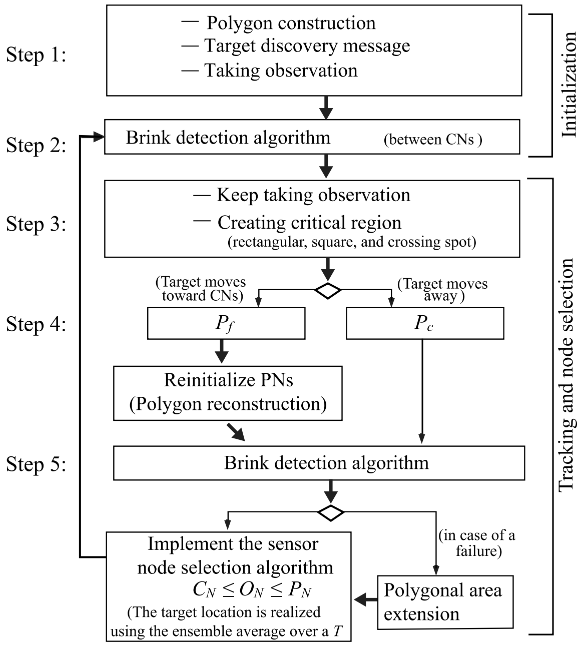
Step 1:
— Polygon construction
— Target discovery message
— Taking observation
Initialization
Step 2:
Brink detection algorithm
(between CNs)
Step 3:
— Keep taking observation
— Creating critical region
(rectangular, square, and crossing spot)
(Target moves toward CNs)
(Target moves away)
Step 4:
P_f
P_c
Reinitialize PNs
(Polygon reconstruction)
Step 5:
Brink detection algorithm
Tracking and node selection
Implement the sensor node selection algorithm
$C_N \leq O_N \leq P_N$
(The target location is realized using the ensemble average over a T
(in case of a failure)
Polygonal area extension

Fig. 5. Illustration of polygon-based tracking framework.

In the beginning, when a target is detected by some nodes, as shown in Appendix D, available in the online supplemental material, the nodes communicate to all of its adjacent neighbors with their detection information, and reconstruct the polygon (Step 2). Once the target is surrounded by the perimeter of a polygon, it becomes P_c. Steps 3 to 5 (including brink detection through the three-phase detection, optimal sensor selection, and polygonal area extension in the case of faults in the WSN or loss of tracking) are continued during the target tracking.

Whenever the CNs are selected by the optimal selection algorithm, the detection probability $\rho^{'}$ or $\rho^{''}$, confirms that the target is about to cross the rectangular phase and then the crossing phase (Step 3). A joint-request message is sent to P_f at the moment the target touches the rectangular phase, saying that the target is approaching (Step 4). All NNs in P_f receive the request, change their state to an *awakening*, and then start sensing. When the target crosses the brink, another joint-request message is sent to the nodes in P_f, saying that the target is crossing the brink. After the target crosses over the brink (i.e., it is now in the new P_c), another message is sent to the NNs in the previous P_c. After receiving the message, all NNs, except the previous CNs, return to the *inactive* state.

The target may move in any way toward any brink. When the target speed is lower or the target moves away, it does not influence the tracking. We think of the target's faster speed. When it is faster, the movements may be abrupt. The CNs keep sensing continuously until the target leaves/enters the square phase. The CNs use the difference in distance d_{ij} between two consecutive sensing results. The results are measured by reducing CRLB covariance to obtain fewer errors in three-phase detection. Since the target

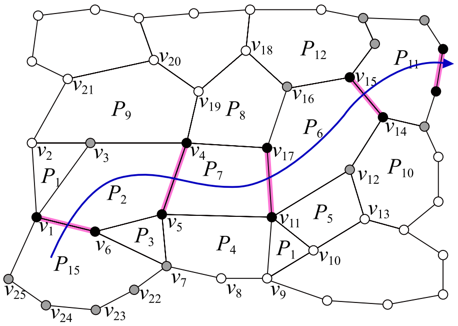

Fig. 6. Detecting target's movements through polygons.

travels across the square phase, and then the rectangular phase, d_{ij} decreases accordingly. The CNs are aware of it (as detailed in Observation E.2). If the target leaves the square phase for the same P_c (as detailed in Observation E.3), the CNs send a message instantly to the NNs in P_c. The NNs remain active and are ready to receive the message. If they receive the message, shortly there-after, they start sensing further. The next procedures go on in the aforementioned way. However, if any rectangular phase is not generated, there is no P_f selected.

According to the framework in Fig. 5, Fig. 6 illustrates the target movement detection through the polygons. The target is initially detected by sensors v_1 and v_6 (shaded black to indicate the CNs) in the polygon P_{15}, and the rest of the corresponding nodes (shaded grey) in P_{15} are in the vigilant manner, and the rest of the nodes in the sensor network are in the *inactive* state when the target is in P_{15}. As shown in Fig. 6, the target travels through the polygons. The tracking of the polygons represents the target tracks. A tracking sequence can be $P_{15} \rightarrow P_2 \rightarrow P_7 \rightarrow P_6 \rightarrow P_{11}$, and so on.

5.2 Fault Tolerance and Tackling Loss of Tracking

Generally, the WSN planarization does not have any fault tolerance support. Thus, initially constructed polygons may not be preserved during tracking. While the target is moving to P_f, if a node cannot execute itself (i.e., it is out of service because of an internal error such as battery depletion, failing to detect itself, or missing from its location) or there is a link failure due to inter-node wireless channel fluctuations, tracking can be interrupted. These result in the event of loss of tracking. There are several ways that we mitigate these situations: by using the outside area of P_c, by extending the polygon area coverage, or merging two or more polygons into one. A detailed elaboration on the fault tolerant detection and tracking, and its associated cost analysis, can be found in Appendix H and Appendix I.2, available in the online supplemental material, respectively.

6 Performance Analysis and Issues

In this section, we briefly analyze some performance issues, such as complexity of the algorithm, the energy costs, and the energy-efficiency of the WSN in FaceTrack. We compare our sensor selection algorithm with the autonomous node selection (ANS) and global node selection (GNS) algorithms for target tracking proposed in [6], [18] (reviewed in the related work in Appendix A, available in the online supplemental material). For each iteration, the computational complexity of GNS and ANS algorithms are $O(P_N)^2$ and $O(P_N - O_N)^2$, respectively, while it is $O(P_N - O_N)$ in FaceTrack. Thus, the computational complexity for all iterations can be given as: $\sum_{i=0}^{O_N}(P_N - i)$ in FaceTrack, $\sum_{i=0}^{O_N}(P_N)^2$ in GNS, and $\sum_{i=0}^{O_N}(P_N - i)^2$ in ANS.

An important goal of FaceTrack is to reduce the total energy cost required by nodes in polygons. Regarding the localized polygon mechanism, we try to minimize the energy cost for message transmissions in each tracking event, and for the optimal node selection. As explained in Appendix I.4, available in the online supplemental material, let $E_t(P_N)$ be the energy cost for nodes in a polygon at time t. Thus, the total energy cost in the WSN during a whole simulation run is given by:

$$E_T(N) = \sum_{P_N \leq N} E_t(P_N). \tag{7}$$

Effective energy cost percentage (EECP). Besides the energy cost analysis above, we use a concept of EECP as a metric to better evaluate the energy-efficiency of the WSN. Usually, in many existing schemes (including GNS and ANS), a large number of nodes are proactively woken up to become prepared for an approaching target. All of these nodes are kept active for a long time. At one time, some of the nodes may participate in tracking for a very short period of time and are active (stay idle) for the rest of the time. On the other hand, some other nodes basically do not actively participate in the tracking operation at all, and they are also active (stay idle). As a result, the network unnecessarily wastes a significant amount of energy. In FaceTrack, a number of nodes outside of P_c may be able to detect the target; we reduce this number of nodes by focusing on the nodes of P_c. The background behind EECP is usually the sensor state transition model [21] that handles the nodes' duty cycles, and the number of neighbors in P_f that are woken up by the CNs by using three-phase detection spots

$$EECP = \frac{E_{the\ nodes\ that\ can\ detect}}{E_{all\ the\ nodes\ that\ are\ in\ the\ active\ state}}. \tag{8}$$

We define EECP in (8), where EECP is the percentage of the energy used by those nodes that can detect the target to the energy used by all of the nodes that are in the active state. In *FaceTrack*, the active nodes are the nodes of P_c and then the nodes of P_f. In a scheme, the higher the EECP gain is, the higher the energy-efficiency would be. An extensive performance analysis of the computation complexity and energy-cost model is carried out in Appendix I, available in the online supplemental material.

7 Simulation Studies

7.1 Methods and Objectives

We evaluate the performance of the FaceTrack framework via simulation. We implement it on the OMNet++ v3.3p1 simulation environment using the Castalia simulator (http://castalia.npc.nicta.com.au/index.php). Here, our focus in conducting the simulations is on two aspects.

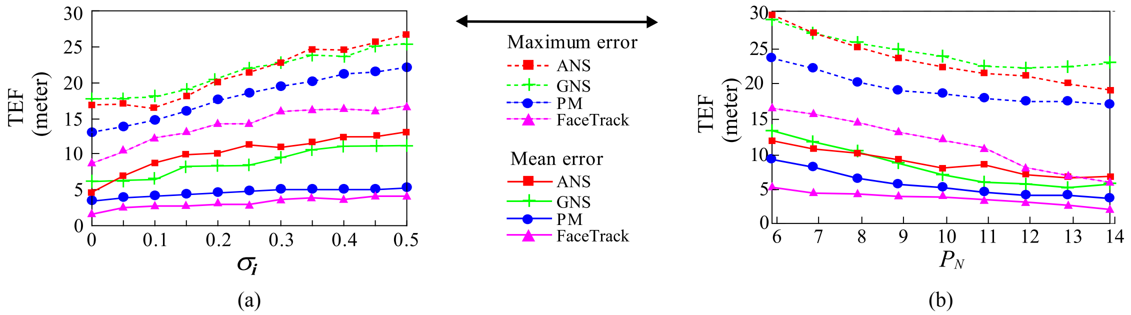

Fig. 7. (a) The effect of noise on tracking errors; (c) the effect of the number of nodes on tracking errors.

1) Tracking accuracy—to observe tracking accuracy, we analyze *tracking error found* (TEF) and *tracking ability rate*. TEF is defined as an averaged error found in meters by all of the nodes that are involved in tracking. TAR is the metric that can show the degree of successful tracking in a system against all the difficulties, such as the presence of high TEF, or faults in the WSN. TAR also includes sensors' duty cycles. Suppose that the target is moving in a trajectory corresponding to e events of the sensors' duty cycles. The number of successful tracking events divided by e is called the TAR, which reflects the tracking accuracy. 2) Energy cost and energy-efficiency—we mainly evaluate the total energy cost required by the number of sensors per tracking event, and energy-efficiency through the EECP. We compared the performance of FaceTrack with existing protocols (PM [3], ANS [18], and GNS [6]).

7.2 Simulation Settings

The simulation is performed within a $400\,\text{m} \times 400\,\text{m}$ 2D square planar field in an area of interest. For simplicity, N (200) sensors are randomly and uniformly distributed. Throughout the simulation, any two sensors can directly communicate via bi-directional wireless links. Their Euclidean distance is not greater than the communication range $(d_{ik} < r_c)$. The target's location in the plane can be perfectly monitored by the nodes if $(d_{ij} < r_s)$. Instead of considering all of the possible combinations of r_c and r_s, we focus on the case of $r_c \geq 2r_s$ in the simulation. All nodes within FaceTrack synchronize with the sink within 1-10 ms, as described in [7].

The plane is partitioned into p polygons, such that each polygon contains at least three nodes. Initially, a period of 10 s is set aside for generating polygons. We use localization for a single vehicle situation. Then, the tracking simulation begins where the target shows up at a random location on the plane with a maximum acceleration of $a_{\min} = 2$ m/s, and a maximum velocity of $v_{\max} = 10$ m/s. We consider the speed of the target (vehicle) from 2 to 10 m/s.

We use (C.5) and (C.8) to generate the acoustic energy readings. The target energy is set at 5,000 mv, and the background noise level is set up to 1 for all sensors. The SNR at different sensors depends very much on the distance between the CNs and the target locations. The energy variation, $\varepsilon_i(t)$, is modeled as a Gaussian random variable with $M = 100$. We use Intel Imote2's power settings where each sensor is with discrete power levels in the interval $\{-10\,\text{dBm}, 0\,\text{dBm}\}$ [20].

7.3 Simulation Results

Study of the tracking accuracy. We compare the tracking performance in terms of accuracy, based on the dynamic moving path, with the optimal path matching (PM), GNS, and ANS. We analyze the mean and maximum tracking error found, which is revealed from the performance results gathered by all of the nodes that involve in tracking over 100 simulation runs. Fig. 7a depicts the performance of different σ_i for the background noise (see (1)). It indicates that the noise brings in some tracking errors. However, FaceTrack shows relatively minimum errors compared to others, i.e., 20 to 50 percent lower in the case of mean errors, and 30 to 50 percent lower in the case of maximum errors.

Fig. 7b shows that the tracking error decreases (i.e., the accuracy of tracking increases) with an increasing P_N, and the polygon-based tracking in FaceTrack clearly achieves a superior performance compared to all PM, ANS, and GNS. Here, P_N is the number of nodes that are involved in tracking in other schemes, while it is the number of nodes in a polygon in FaceTrack. $O_N \leq P_N$, where the optimal number of nodes (O_N) should be taken into account in this discussion so as to know how many optimal nodes (which become the CNs) are used. Considering the target's speed up to 10 m/s and O_N, we compare our results with real data sets collected from PM, ANS, and GNS schemes. We set $\sigma = 0.5$. The tracking results are averaged over all 100 simulation runs for FaceTrack and when the localization error is averaged over in the square spot and rectangular spot. The PM is better for large P_N and O_N , but not for small O_N. In FaceTrack, the node selection appears to be robust against some mismatch between the estimated and actual errors. Overall, FaceTrack achieves a clear advantage when $2 \leq O_N \leq 3$ and $4 \leq P_N \leq 8$. We think that the choice of O_N is actually better in FaceTrack, where the results are opposite in both ANS and GNS.

An interesting observation on tracking accuracy can be seen in Fig. 8. We analyze tracking ability rate based on the overall simulation results, considering the underlying techniques such as a sensor's duty cycle and all the difficulties, such as the presence of high TEF, faults in the network, etc. Fig. 8a shows that the TAR varies according to the target's speed. We observe that when P_N is very small or large (sparse or dense), the TAR is slightly lower. The TAR is fairly close to 100 percent when $6 \leq P_N \leq 9$ and when the target's speed is 4-6 m/s. Fig. 8b depicts that ANS has slightly higher tracking ability than GNS, although ANS shows slightly more tracking errors. Although PM has

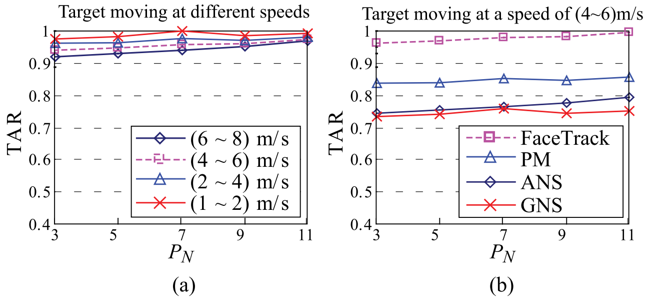

Fig. 8. Tracking ability rate (TAR): (a) at different speed of the target in FaceTrack; (b) in different schemes.

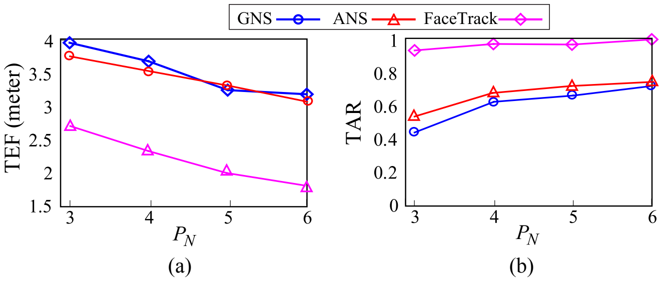

Fig. 10. Experimental tracking performance: (a) mean TEF; (b) mean TAR achieved.

higher TAR (about 30 percent) than both ANS and GNS, *FaceTrack* largely outperforms the PM, achieving higher TAR (about 40 to 70 percent) than other schemes. This also hints that the tracking ability of *FaceTrack* in the presence of high TEF (see Fig. 7b) is significantly higher than others.

Study of energy cost and energy-efficiency. Fig. 9a shows the performance of FaceTrack in terms of energy cost versus P_N. In FaceTrack, nodes in a polygon select O_N, in which $2 \leq O_N$ and $4 \leq P_N$. The total energy cost ($E_T(N)$) gradually increases as P_N or O_N increases. Simulation results show that the optimal sensor selection algorithm in FaceTrack outperforms ANS and GNS. On the contrary, the tracking error decreases as O_N increases and $4 \leq P_N \leq 7$. FaceTrack shows a tradeoff between the performance and P_N that is needed to save energy because of the value of P_N and O_N in the algorithm, and the geographic routing structure. ANS needs $4 \leq O_N$, and GNS needs $4 < O_N$ at a time window, which makes them energy cost-ineffective. As expected, when $O_N = 2$ and $4 \leq P_N \leq 7$ in the algorithm, FaceTrack performs better, even when $O_N = 1$ $3 \leq P_N \leq 4$ in some occasions, such as node failure. If $(D/2) \leq r_s$, CNs communicate over a long distance; D becomes longer, thus, the energy cost increases slightly. When there is a failure, P_N increases by almost double.

The relationship between EECP and the target speed is shown in Fig. 9b. We can observe that FaceTrack achieves superior EECP gain. Here, the higher EECP gain in a scheme means that the scheme gains higher energy-efficiency than others. FaceTrack achieves energy-efficiency by 50-90 percent compared to GNS and ANS. In addition, it shows the advantage of FaceTrack under varying speeds of the target as an example. We can say that when the target moves at a high speed (6 m/s or more), FaceTrack behaves better than GNS and ANS. It is obvious that we reduce the unnecessary energy cost by reducing the number of sensors involved in tracking that are not actually needed (they may become idle) during tracking. However, considering the delays of sensor state transitions (active, awakening, and inactive), some performance loss may be present.

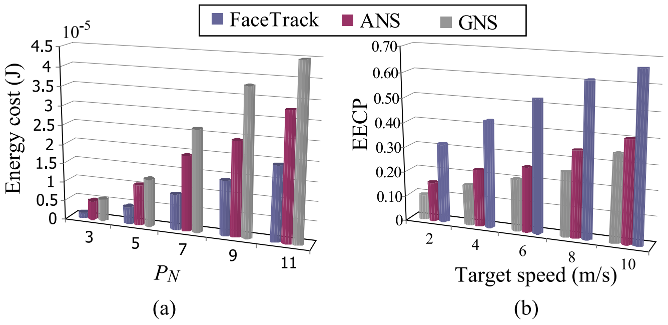

Fig. 9. (a) Energy cost of the optimal node selection algorithms, and (b) EECP versus target speed.

Additional simulation studies considering more metrics (sensor measurement errors and brink detection ability and simulation results) are given in Appendix J, available in the online supplemental material.

8 Proof-of-Concept System

To validate the applicability and benefits of FaceTrack, we implement a proof-of-concept system using the TinyOS [19]. The system is deployed in the Hong Kong PolyU campus stadium to track an experimental vehicle. The system contains 20 Imote2 wireless sensors [20]. Our objective of this implementation is to observe the tracking performance of FaceTrack, considering several aspects, including TEF and TAR.

We conduct a total of 10 rounds of experiments. We analyze the experimental results gathered from all of the rounds. Fig. 10a shows that the TEF reduces (i.e., the accuracy of tracking increases) with an increasing P_N, and FaceTrack has a lower TEF (around 60 to 70 percent) than both ANS and GNS. The target is enabled to move at speeds up to (3-6)m/s and $\sigma = 0.5$. We estimate TAR based on the experimental results, taking TEF and localization errors into account. In Fig. 10b, we observe that as P_N increases, the TAR also increases. It also reveals that the TAR is close to 100 percent when $5 \leq P_N \leq 7$, which validates our simulation results. FaceTrack outperforms both ANS and GNS, by achieving significantly higher tracking ability (more than 40 percent) than both of them. More metrics and experimental setups, as well as more extensive experimental results, can be found in Appendix K, available in the online supplemental material.

9 Conclusion

The main functionality of a surveillance wireless sensor network is to track an unauthorized target in a field. The challenge is to determine how to perceive the target in a WSN efficiently. We proposed a unique idea to achieve a WSN system for detecting movements of a target using polygon (face) tracking that does not adopt any prediction method.

Evaluation results demonstrated that the proposed tracking framework can estimate a target's positioning area, achieve tracking ability with high accuracy, and reduce the energy cost of WSNs. From the framework, two facts can be highlighted emphatically: 1) the target is always detected inside a polygon by means of a brink detection, and 2) it is robust to sensor node failures and target localization errors.

Two interesting problems, which we are currently investigating, are as follows: 1) the performance of variable brink lengths of the polygon versus adjustable transmission power levels in a WSN for target detection and its energy cost in the WSNs; 2) the impact of the target's dynamic movements, brink detection, and real-time polygon forwarding in target tracking.

Acknowledgments

This work is supported in part by the National Natural Science Foundation of China under Grant Nos. 61272151, 61272496 and in part by ISTCP grant 2013DFB10070, in part by HKRGC under GRF grant PolyU5106/11E and HK PolyU Niche Area Fund RGC under grant N_PolyU519/12, and in part by NSF under grants ECCS 1231461, ECCS 1128209, CNS 1138963, CNS 1065444, and CCF 1028167.

References

[1] O. Kaltiokallio, M. Bocca, and L.M. Eriksson, "Distributed RSSI Processing for Intrusion Detection in Indoor Environments," *Proc. Ninth ACM/IEEE Int'l Conf. Information Processing in Sensor Networks (IPSN)*, pp. 404-405, 2010.

[2] Y. Wang, M. Vuran, and S. Goddard, "Analysis of Event Detection Delay in Wireless Sensor Networks," *Proc. IEEE INFOCOM*, pp. 1296-1304, 2011.

[3] Z. Zhong, T. Zhu, D. Wang, and T. He, "Tracking with Unreliable Node Sequence," *Proc. IEEE INFOCOM*, pp. 1215-1223, 2009.

[4] W. Zhang and G. Cao, "Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks," *IEEE Trans. Wireless Comm.*, vol. 3, no. 5, Sept. 2004.

[5] Z. Wang, W. Lou, Z. Wang, J. Ma, and H. Chen, "A Novel Mobility Management Scheme for Target Tracking in Cluster-Based Sensor Networks," *Proc. Sixth IEEE Int'l Conf. Distributed Computing in Sensor Systems (DCOSS)*, pp. 172-186, 2010.

[6] L.M. Kaplan, "Global Node Selection for Localization in a Distributed Sensor Network," *IEEE Trans. Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 113-135, Jan. 2006.

[7] T. He, P. Vicaire, T. Yan, L. Luo, L. Gu, G. Zhou, R. Stoleru, Q. Cao, J. Stankovic, and T. Abdelzaher, "Achieving Real-Time Target Tracking Using Wireless Sensor Networks," *Proc. 12th IEEE Real-Time and Embedded Technology and Applications Symp. (RTAS)*, pp. 37-48, 2006.

[8] M. Waelchli, M. Scheidegger, and T. Braun, "Intensity-Based Event Localization in Wireless Sensor Networks," *Proc. Conf. Int'l Federation for Information Processing Wireless On-Demand Network Systems and Services (IFIP WONS)*, pp. 41-49, 2006.

[9] Y. Zhou, J. Li, and D. Wang, "Posterior Cramer-Rao Lower Bounds for Target Tracking in Sensor Networks with Quantized Range-Only Measurements," *IEEE Signal Processing Letters*, vol. 17, no. 2, pp. 377-388, Feb. 2010.

[10] X. Wang, M. Fu, and H. Zhang, "Target Tracking in Wireless Sensor Networks Based on the Combination of KF and MLE Using Distance Measurements," *IEEE Trans. Mobile Computing*, vol. 11, no. 4, pp. 567-576, Apr. 2012.

[11] M. Chu, H. Haussecker, and F. Zhao, "Scalable Information Driven Sensor Querying and Routing for Ad Hoc Heterogeneous Sensor Networks," *J. High Performance Computing Applications*, vol. 16, no. 3, pp. 293-313, 2002.

[12] B. Leong, S. Mitra, and B. Liskov, "Path Vector Face Routing: Geographic Routing with Local Face Information," *Proc. IEEE Int'l Conf. Network Protocols (ICNP)*, pp. 47-158, 2005.

[13] Y.-J. Kim, R. Govindan, B. Karp, and S. Shenker, "Geographic Routing Made Practical," *Proc. USENIX Networked. Systems Design and Implementation (NSDI)*, pp. 217-230, 2005.

[14] M.A. Rajan, M.G. Chandra, L.C. Reddy, and P. Hiremath, "Concepts of Graph Theory Relevant to Ad-Hoc Networks," *J. Computers, Comm. and Control*, vol. 3, no. 2008, pp. 465-469, 2008.

[15] Q. Huang, C. Lu, and G.-C. Roman, "Mobicast: Just-in-Time Multicast for Sensor Networks under Spatiotemporal Constraints," *Proc. ACM/IEEE Int'l Conf. Information Processing in Sensor Networks (IPSN)*, pp. 442-457, 2003.

[16] K. Liu, N. Abu-Ghazaleh, and K.D. Kang, "JiTS: Just-in-time Scheduling for Real-Time Sensor Data Dissemination," *Proc. IEEE Pervasive Computing and Comm. (PerCom)*, pp. 42-46, 2006.

[17] M.Z.A. Bhuiyan, G. Wang, and J. Wu, "Polygon-Based Tracking Framework in Surveillance Wireless Sensor Networks," *Proc. IEEE Int'l Conf. Parallel and Distributed Systems (ICPADS)*, pp. 174-181, 2009.

[18] L.M. Kaplan, "Local Node Selection for Localization in a Distributed Sensor Network," *IEEE Trans. Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 136-146, Jan. 2006.

[19] TinyOS Reference Manual, http://www.tinyos.net, 2013.

[20] Imote2 Datasheet, http://docs.tinyos.net/tinywiki/index.php/Imote2, 2013.

[21] M.Z.A. Bhuiyan, G. Wang, and J. Wu, "Target Tracking with Monitor and Backup Sensors in Wireless Sensor Networks," *Proc. IEEE Int. Conf. Computer Comm. and Networks (ICCCN)*, pp. 1-6, 2009.

[22] J. Cartigny, F. Ingelrest, D. Simplot, and I. Stojmenovic, "Localized LMST and RNG Based Minimum-Energy Broadcast Protocols in Ad Hoc Networks," *Ad Hoc Networks (Elsevier)*, vol. 3, no. 2, pp. 1-16, 2005.

[23] G. Toussaint, "The Relative Neighborhood Graph of Finite Planar Set," *Pattern Recognition*, vol. 12, no. 4, pp. 261-268, 1980.

[24] M. Crocker, *Handbook of Acoustics*. John Wiley & Sons, 1998.

[25] M.D. Berg, M.V. Kerveid, M. Overmars, and O. Schwarzkof, *Computational Geometry*. Springer, 1998.

Guojun Wang received the BSc degree in geophysics and the MSc and PhD degrees in computer science from Central South University, China. He is currently a chair and professor of the Department of Computer Science at Central South University. He is also the director of the Trusted Computing Institute of the University. He has been an adjunct professor at Temple University, Philadelphia, Pennsylvania; a visiting scholar at Florida Atlantic University, Boca Raton, Florida; a visiting researcher at the University of Aizu, Japan; and a research fellow at the Hong Kong Polytechnic University. His research interests include network and information security, Internet of things, and cloud computing. He is a distinguished member of the CCF, and a member of the IEEE, ACM, and IEICE.

Md Zakirul Alam Bhuiyan received a BSc degree from International Islamic University Chittagong, Bangladesh, in 2005, and a MSc degree and a PhD degree from Central South University, China, in 2009 and 2013 respectively, all in Computer Science and Technology. He is currently a postdoctoral research fellow at Central South University, where he is a key member of the Trusted Computing Institute of the university. His research interests focus on designing distributed target tracking algorithms in wireless sensor networks, designing cyber physical systems (CPS), and sensor-cloud computing. He was a recipient of the top-notch PhD student award and of the best paper award in IEEE ISPA 2013. He was a research assistant (RA) in Hong Kong PolyU. He is a member of IEEE and a member of ACM.

Jiannong Cao received the BSc degree from Nanjing University, China, and the MSc and PhD degrees from Washington State University, Pullman, all in computer science. He is currently a chair professor and head of the Department of Computing at Hong Kong Polytechnic University. His research interests include parallel and distributed computing, computer networks, mobile and pervasive computing, fault tolerance, and middleware. He has co-authored four books, co-edited nine books, and published more than 300 papers in major international journals and conference proceedings. He has directed and participated in numerous research and development projects and, as a principal investigator, obtained over HK$25 million grants. He is a senior member of China Computer Federation, a senior member of the IEEE, and a member of the ACM. He is the chair of the Technical Committee on Distributed Computing of IEEE Computer Society. He has served as an associate editor and a member of the editorial boards of many international journals, and as a chair and member of organizing/program committees for many international conferences.

Jie Wu is the chair and a Laura H. Carnell Professor in the Department of Computer and Information Sciences at Temple University. Prior to joining Temple University, he was a program director at the National Science Foundation and Distinguished Professor at Florida Atlantic University. His current research interests include mobile computing and wireless networks, routing protocols, cloud and green computing, network trust and security, and social network applications. Dr. Wu regularly published in scholarly journals, conference proceedings, and books. He serves on several editorial boards, including IEEE Transactions on Computers, IEEE Transactions on Service Computing, and Journal of Parallel and Distributed Computing. Dr. Wu was general co-chair/chair for IEEE MASS 2006 and IEEE IPDPS 2008 and program co-chair for IEEE INFOCOM 2011. Currently, he is serving as general chair for IEEE ICDCS 2013 and ACM MobiHoc 2014, and program chair for CCF CNCC 2013. He was an IEEE Computer Society Distinguished Visitor, ACM Distinguished Speaker, and chair for the IEEE Technical Committee on Distributed Processing (TCDP). Dr. Wu is a CCF Distinguished Speaker and a Fellow of the IEEE. He is the recipient of the 2011 China Computer Federation (CCF) Overseas Outstanding Achievement Award.