J. Parallel Distrib. Comput. 70 (2010) 212–224

Contents lists available at ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

Scalable continuous object detection and tracking in sensor networks

Shin-Chih Tu [a], Guey-Yun Chang [a,*], Jang-Ping Sheu [b], Wei Li [a], Kun-Ying Hsieh [a]

[a] Department of Computer Science and Information Engineering, National Central University, Jhongli, Taiwan, ROC

[b] Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, ROC

ARTICLE INFO

Article history:
Received 22 May 2008
Received in revised form
24 November 2009
Accepted 1 December 2009
Available online 16 December 2009

Keywords:
Boundary detection
Moving objects
Object tracking
Wireless sensor networks

ABSTRACT

With the advancement of MEMS technologies, sensor networks have opened up broad application prospects. An important issue in wireless sensor networks is object detection and tracking, which typically involves two basic components, collaborative data processing and object location reporting. The former aims to have sensors collaborating in determining a concise digest of object location information, while the latter aims to transport a concise digest to sink in a timely manner. This issue has been intensively studied in individual objects, such as intruders. However, the characteristic of continuous objects has posed new challenges to this issue. Continuous objects can diffuse, increase in size, or split into multiple continuous objects, such as a noxious gas. In this paper, a scalable, topology-control-based approach for continuous object detection and tracking is proposed. Extensive simulations are conducted, which show a significant improvement over existing solutions.

1. Introduction

Object detection and tracking is an essential capability in many sensor network applications, such as military (tracking enemy vehicles, detecting illegal border crossings), civilian (tracking the movement of wild animals in wildlife preserves), environmental monitoring (e.g., traffic, habitat, security), and *etc*. There are two major components, collaborative data processing and object location reporting, in object detection and tracking problem. The former aims to have sensors collaborating in determining a concise digest of object location information (i.e., location information of those sensors which detect the existence of objects), while the latter aims to transport the concise digest to sink in accurately and in a timely manner.

Objects to be tracked can be classified into two major categories: continuous and individual. Continuous objects usually spread in a very large region and may diffuse, increase in size, or split into multiple continuous objects, such as a noxious gas, biochemicals, chemical liquids, herd moving, and army advances. Different from continuous objects, individual objects have a fixed size, such as intruders, tanks, and animals. Besides, individual objects usually have size much smaller than continuous objects. A considerable number of studies has been conducted on detection and tracking of one or several individual objects [3,5,9,10,12,15,18,19,22]. Recently, importance has been attached to the study of continuous object detection and tracking [6,11,13].

Define a sensor node to be an *event node* if it detects objects, and *normal node* otherwise. In general, object location is estimated by the aid of the location information of event nodes. Hence, efficient determining a concise digest of location information of event nodes is an important issue in collaborative data processing. This issue has been intensively studied in individual object detection and tracking. However, large numbers of event nodes in continuous objects have posed new challenges. Define an event node to be a *reporter* if its location information is required to report to the sink. Due to spatial locality of event nodes, reducing the number of reporters is possible and critical in continuous object detection and tracking.

In [11], a cluster-based approach, called DCSODT, is proposed. Of course, there are two essential phases in DCSODT: collaborative data processing and object location reporting. The former is concerned with reducing the number of reporters, while the latter is concerned with transmission of the location information of reporters. The main idea of their collaborative data processing phase is to choose event nodes at continuous objects' boundaries as reporters. For this purpose, they defined reporters to be event nodes with one or more (one-hop) neighboring normal nodes. For object location reporting, all reporters are organized into clusters in a distributed manner. Each cluster head sends the location information of those reporters within the same cluster to the sink by available routing protocols. For cost saving, each current reporter sends its location information to the old cluster head by utilizing route information in an old reporter and old cluster head (if it exists). If the local boundary of an object is going to move out of the area covered by a cluster, the old cluster head chooses the reporter which is closest to the midpoint of the cluster, as a

* Corresponding author.
E-mail address: gychang@csie.ncu.edu.tw (G.-Y. Chang).

0743-7315/$ – see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jpdc.2009.12.001

new cluster head. The main drawback of this protocol is the lack of scalability, i.e., the number of reporters in this protocol increases as node degree (number of neighbor nodes) increases. Consider a scenario where an object is detected by some sensors, each of whose communication range is 10 m. And the node density of the sensor network is assumed to be one node per 10 m^2. Then each node has $10^2 \times 3.14/10 = 31$ neighbor nodes on average. In other words, a normal node may be in 31 event nodes' communication range. That is, a normal node may bring in 31 reporters, which results in a large number of reporters. On the other hand, although these reporters are close to each other (all in a normal node's communication range), most of them are of no benefit to improving the accuracy of object location estimation. Although DCSODT can be improved by reducing the sensor nodes' communication range, it will cause the estimated object location information to be more inaccurate. Similar approaches are also investigated [4]. In [4], a sensor node s is called to be near the object boundary if it is an event node and there exist normal nodes within the distance of d to s. Clearly, the value of d determines the performance of the algorithms. By the aid of the number of event nodes and the number of normal nodes within the predefined distance of D ($>d$) to s, the authors have designed three approaches: the statistical approach, the image processing approach, and the classifier-based approach, to minimize the value of d [4]. In [6], a collaborative data processing scheme in wireless sensor networks with faulty sensor nodes is proposed. A sensors node is called faulty if its reading deviates significantly from other readings of neighboring sensors. However, event nodes at continuous objects' boundaries are usually mistaken for faulty ones since they are neighboring to both event nodes and normal nodes. Two schemes are provided to separate real faulty nodes out. In [26], a collaborative data processing scheme without sensor nodes' location information is introduced. In [21], a distributed algorithm for discovering sensor nodes on the boundary of the sensor networks is suggested. Based on the observation that a sensor node on the boundary creates irregularities in hop count distances, i.e., its minimum hop counts to a given node is locally maximal, sensor nodes on the boundary could be identified by the aid of a shortest path tree.

In this paper, a Scalable, topology-control-based protocol for COntinuous Object detection and Tracking (SCOOT) is put forward. In the collaborative data processing phase, a two-step process is introduced to determine whether an event node is of benefit in determining the object's location. Event nodes whose location information is of no benefit are not chosen as reporters. In the object location reporting phase, a clustering method with a low communication overhead and low data packet overhead is introduced to efficiently transmit location information of reporters to the sink. Compared with the previous approach, simulation results show that there is a great improvement in our protocol in reducing the number of reporters, control message overhead and data message overhead. It also reveals that our protocol is scalable in node degree and accommodates continuous objects with large size.

The rest of this paper is organized as follows. In Section 2, some necessary notations are introduced. In Section 3, our distributed protocol for object detection and tracking is proposed. In Section 4, we explain how our protocol deals with exceptions. In Section 5, the performance of our protocol is evaluated through simulations. Finally, in Section 6, we conclude this paper with some remarks.

2. Preliminaries

Throughout this paper, it is assumed that sensor nodes are static, uniformly distributed without communication hole [7,8] and aware of their own location either by the Global Positioning System (GPS) or other positioning methods [16,17,20]. Initially, each sensor node collects two-hop neighboring information that contains the node's ID and location information by exchange of "*hello*" messages.

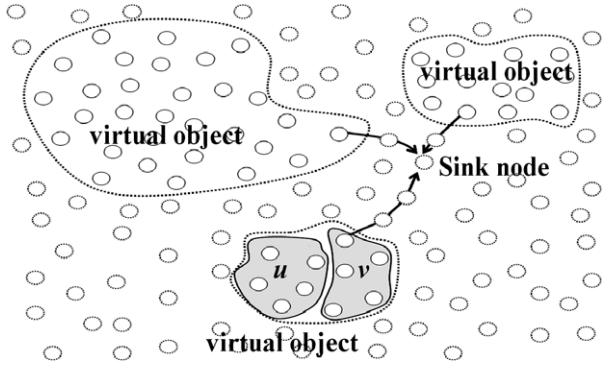

Fig. 1. An example of virtual objects.

A sensor network is represented by an undirected graph $G = (V, E)$, where each node u in V denotes a sensor node, and each edge (u, v) in E denotes that u and v can communicate directly. Each node v able to communicate directly with u is called a *neighbor* of u. Let $N(u)$ denote the neighborhood of u, i.e., $N(u) = \{v|(u, v) \in E\}$.

In a densely deployed wireless sensor network, many typical problems are aggravated by the large number of neighbors: many nodes interfere with each other; there are many possible routes, nodes might needlessly use large transmission power to talk to distant nodes directly; routing protocols might have to re-compute routes even if only a small node movement has happened. Topology control is a common method used to deliberately restrict the set of nodes that is considered neighbors of a given node in densely deployed wireless sensor networks. Delaunay triangulation is one of popular methods used in topology control [1,2]. Therefore, we assume that Delaunay triangulation has been applied to the sensor networks. The Delaunay triangulation [1,14] of a node set is a collection of edges satisfying the "empty circle" property: for each edge (u, v), there exists a circle containing nodes u and v but not containing any other nodes. Let $G^T = (V, E^T)$ denote the Delaunay triangulated graph on V, i.e., each edge (u, v) in E^T denotes that u and v share an edge in their Delaunay triangulation. G^T can be constructed in a distributed manner if each node has one-hop neighbor information [24]. Let $N^T(u)$ be the neighborhood of u in the triangulated graph G^T, i.e., $N^T(u) = \{v|(u, v) \in E^T\}$. Notice that $N^T(u) \subseteq N(u)$ for each node u.

Sometimes real objects are indistinguishable. Refer to Fig. 1. Two gray areas denote two close clumps of noxious gas. If nodes u and v are neighbors in G^T, then it is impossible to distinguish these two objects. For ease of the following discussion, if two distinct real continuous objects O_1 and O_2 are detected by the same node or two neighboring nodes, respectively, at the same time, O_1 and O_2 are assumed to belong to the same virtual object in the rest of this paper. Define a *virtual object* to be a real object or union of several real objects and which satisfies the following three conditions: (1) Event nodes detecting those objects included in the same virtual object (at the same time) should be connected in G^T; (2) No event node detects two distinct virtual objects at the same time; (3) Two event nodes detecting two distinct virtual objects at the same time, respectively, are not neighboring to each other in G^T. We define the *boundary* of a virtual object to be the set of points which encloses all event nodes but which does not enclose any normal nodes. Throughout this paper, unless specified otherwise, virtual objects are assumed.

Based on this assumption, two arbitrary objects are far enough from each other (see Fig. 1). In other words, event nodes detecting

one object are far enough from event nodes detecting others. Hence objects can be considered independently. For ease of the following discussion, we assume that there is only one object in the sensor network.

Define the *enclosed region* of node set V' to be a polygon with an ordered node set $V' = \{v_1, v_2, v_3, \ldots, v_n\}$ and polygon edges set $E' = \{(v_1, v_2), (v_2, v_3), (v_2, v_3), \ldots, (v_{n-1}, v_n)\} \cup \{(v_n, v_1)\}$, where E' is a subset of E. For example, an enclosed region of node set $\{1, 2, \ldots, 7\}$ in Fig. 2(a) is illustrated in Fig. 2(b). A node v is said to be *enclosed by node set* V' if v is located in the enclosed region of V'. For example, in Fig. 2(b), nodes 1, 2, ..., and 9 are enclosed by node set $\{1, 2, \ldots, 7\}$.

For simplicity, let *event set* denote the set of all event nodes detecting the object. A subset of event set, say V', is said to be *sufficient* if all event nodes are enclosed by V' and no normal nodes are enclosed by V'. Refer to Fig. 2(a) again. If node set $\{1, 2, \ldots, 9\}$ is an event set, then node set $\{1, 2, \ldots, 7\}$ is a sufficient set. Obviously, a sufficient set consists of event nodes which are close to object boundaries. For ease of the following discussion, let a *sensor node's location information* denote its absolute/relative coordinates and let *objection location information* denote the digest of all event nodes' location information.

Notice that accuracy of objection location estimation is similar no matter which sink obtains the location information of all event nodes or which sink obtains the location information of nodes in a sufficient set. The reasons are as follows. First, when a sensor node becomes an event node, it is guaranteed that part of the continuous object is in the sensor node's sensing range. And no further knowledge about the relationship of object location and the sensing range is available. Second, due to objects possibly changing over time and that the transmission of object location information takes time, object location information transmitted to the sink goes out of date. Clearly, these two problems cannot be solved even if the sink obtains the location information of all event nodes.

Define an object to be *ring-type* if it has a sufficient set, and *non-ring-type* otherwise. Examples of ring-type objects and non-ring-type objects are shown in Fig. 3(a) and Fig. 3(b), respectively. In Fig. 3(a) (or Fig. 3(b)), light gray area denotes object location and node set $\{1, 2, \ldots, 8\}$ is event set. Clearly, node set $\{1, 2, \ldots, 6\}$ is a sufficient set corresponding to the object in Fig. 3(a), while there is no sufficient set corresponding to the object in Fig. 3(b) because normal node 9 is enclosed by node set $\{1, 2, \ldots, 6\}$.

3. The proposed protocol

In this section, a continuous object detection and tracking protocol is proposed. There are two essential phases in our protocol: collaborative data processing and object location reporting. The former is concerned with a reduction of the number of reporters, while the latter is concerned with transmission of the location information of reporters. Collaborative data processing phase is triggered when sensor nodes are aware of object changes (e.g., object moving) and an object location reporting phase is executed periodically. In this section, only ring-type objects are considered, and non-ring-type objects are considered in Section 4.

3.1. Collaborative data processing phase

For a ring-type object, all event nodes can be enclosed by a sufficient set. That is, the enclosed region of a sufficient set is an estimation of object location. So, a sufficient set could be chosen as the set of reporters. A scenario is shown in Fig. 3(a). There the light gray area denotes the object's location and dark gray nodes constitute a sufficient set. Obviously, the enclosed region of these dark gray nodes, i.e., the polygon, is an estimation of the object's location. Since it is desirable to keep the number of reporters (sufficient set) small, we aim to determine a sufficient set as small as possible in this phase.

Let the *difference* of two node sets V_1 and V_2, written $V_1 - V_2$, be the set of nodes in V_1 but not in V_2. Define a subset of event set, say V'', to be *unnecessary* if the difference of event set and V'' is sufficient. Refer to Fig. 3(a). Node set $\{7, 8\}$ is unnecessary when the node set $\{1, 2, \ldots, 8\}$ is the event set in this example. Since the difference between the event set and an unnecessary set is also sufficient, determining a small sufficient set is tantamount to determining a large unnecessary set. So, in the rest of this section, we try to determine an unnecessary set as large as possible. Our idea is that event nodes whose location information is of no benefit to improving the accuracy of object location estimation should constitute an unnecessary set. In order to realize this idea, two steps, coarse reduction and fine reduction, are taken and introduced in Sections 3.1.1 and 3.1.2, respectively.

3.1.1. Coarse reduction

Intuitively, event nodes not close to object boundaries are of no benefit (to improving the accuracy of object location estimation). Refer to Fig. 4(a). There the gray area denotes the object's location and seven big circles denote the communication ranges of nodes 20, 21, ..., and 26. That is, nodes 1, 2, ..., and 19 are event nodes and nodes 20, 21, ..., and 33 are normal nodes. Clearly, event nodes not neighboring to normal nodes, say event nodes 18 and 19, are not close to object boundaries. Even if event nodes are neighboring to normal nodes, they may not be close enough to object boundaries. For example, event nodes 13, 14, ..., and 17 are not close enough to object boundaries, i.e., event nodes 13, 14, ..., and 17 are of no benefit. Notice that the number of event nodes which is neighboring to normal nodes and not close to object boundaries increases as the node degree increases. In order to determine an unnecessary set as large as possible, we devote our effort to the identification of such event nodes.

Before a coarse step, the following preprocessing is required. (1) Each sensor node collects two-hop neighbor information. (2) Apply Delaunay triangulation to the sensor network in a distributed manner. (3) Apply a distributed time synchronization scheme to the sensor network [25]. Let G^T denote the resulting triangulated graph. The main idea is that an event node without neighboring normal nodes in G^T is of no benefit. For example, Fig. 4(b) denotes the triangulated graph, G^T, of the sensor network in Fig. 4(a). Since each of the nodes 13, 14, ..., and 17 is not neighboring to any normal nodes in G^T, we have that nodes 13, 14, ..., and 17 are of no benefit. In this case, nodes 13, 14, ..., and 19 constitute an unnecessary set U.

Details of a coarse step are described as follows. When a normal node u detects an object, u becomes an event node. Then, u informs its neighbors of its role change, i.e., $N(u)$, by broadcasting an advertisement message. An advertisement message consists of the sender's ID, sender's role, and the ID of arbitrary one of the sender's neighboring normal nodes in GT (used in a fine reduction step in Section 3.1.2). There are four kinds of role played by a node: (1) a normal node; (2) an event node near the margin of the network; (3) an event node with at least one neighboring normal node in GT; (4) otherwise. Notice that in our protocol, no matter which role change a node has, it should broadcast an advertisement message. When an event node v receives an advertisement message (i.e., some node in $N(v)$ has changed its role), v should check whether its role changes. This is because a node's role change may result from a neighboring node's role change. For example, in Fig. 4(b), if node 26's role changes from the first kind to the third, then node 9's role may change from the third kind to the fourth.

Clearly, each node belonging to the fourth kind is that event node which is not close to object boundaries. Let U be the set of

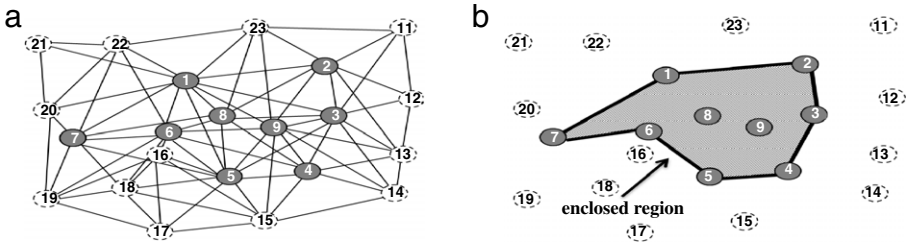

Fig. 2. An example of an enclosed region. (a) A subgraph of G. (b) The enclosed region of node set {1, 2, 3, 4, 5, 6, 7}.

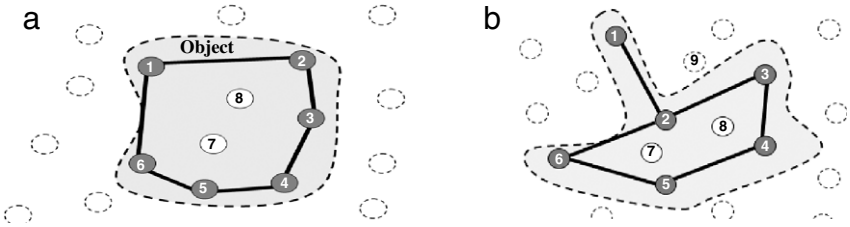

Fig. 3. Two kinds of object. (a) Ring-type. (b) Non-ring-type.

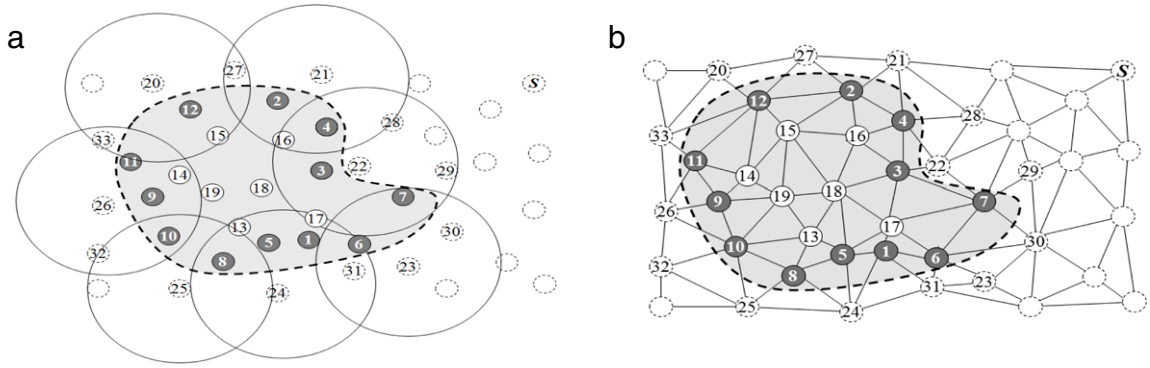

Fig. 4. Unnecessary set. (a) Unnecessary set {13, 14, 15, 16, 17, 18, 19} in G. (b) Unnecessary set {13, 14, 15, 16, 17, 18, 19} in G^T.

nodes belonging to the fourth kind. Clearly, U is unnecessary. Then a sufficient set, S, which is the difference between an event set and U can be obtained. Clearly, S consists of event nodes belonging to the second and the third kinds. In Fig. 4(b), S consists of nodes 1, 2, …, and 12.

3.1.2. Fine reduction

In a coarse reduction step, an unnecessary set U is obtained. However, the determined sufficient set, i.e., S, may still have some nodes which are not beneficial to improving the object location estimation accuracy, such as node 9 in Fig. 4(b). So, a large unnecessary set U^* which is an extension of original U is determined in this step. The main idea is that if event node u in S belongs to $U^* - U$, then $U \cup \{u\}$ is unnecessary. For example, in Fig. 4(b), $U = \{13, 14, \ldots, 19\}$. If $U^* = U \cup \{5, 9\}$, then both $U \cup \{5\}$ and $U \cup \{9\}$ are unnecessary.

For ease of the following discussion, the definition below is needed. Suppose that event nodes u, v_α, and v_β are in S (i.e., they are not in U). Define a node u to be *located between* nodes v_α and v_β if the following two conditions hold:

(1) u, v_α and v_β are mutual neighbors (i.e., $u \in N(v_\alpha) \cap N(v_\beta)$, $v_\alpha \in N(u) \cap N(v_\beta)$, and $v_\beta \in N(u) \cap N(v_\alpha)$); and
(2) There exists a path $v_\alpha, \ldots, u, \ldots, v_\beta$ in G^T satisfying that all nodes on this path are in S.

Let $I(v_\alpha, v_\beta)$ be the set of nodes located between v_α and v_β. Notice that all nodes in $I(v_\alpha, v_\beta) \cup \{v_\alpha, v_\beta\}$ belong to S. Refer to Fig. 5. If nodes 8, 5, 1 and 6 are mutual neighbors in G, then nodes 5 and 1 are located between nodes 8 and 6, i.e., $I(8, 6) = \{5, 1\}$.

Suppose that event node $u \in S$. According to our observation, $U \cup \{u\}$ is unnecessary if u satisfies all the following four conditions.

(1) There exist two event nodes in S, say v_α and v_β, such that u is located between v_α and v_β;
(2) All nodes located between v_α and v_β are at the same part divided by L;
(3) No normal node is enclosed by $I(v_\alpha, v_\beta) \cup \{v_\alpha, v_\beta\}$;
(4) For each node w located between v_α and v_β, w and the neighboring normal node of w are at different parts divided by L,

where L denotes the line passing through v_α and v_β. Notice that conditions (3) and (4) can be verified with the aid of advertisement messages and two-hop neighbor information. Refer to Fig. 5. If nodes 8 and 6 are neighbors, then $U \cup \{5, 1\}$ is unnecessary because nodes 5, 1 are located between nodes 8 and 6; all nodes located between nodes 8 and 6 (i.e., nodes 5 and 1) are at the same part divided by the line L'; no normal node is enclosed by $I(8, 6) \cup \{8, 6\}$ (see blue area in Fig. 5); neither node 5 nor node 1 has neighboring normal nodes (i.e., nodes 24 and 31) at the same part (divided by L') as nodes 5 and 1 are, where L' denotes the line passing through nodes 8 and 6.

However, an event node $v \in S$ satisfying that $U \cup \{v\}$ is unnecessary may not belong to $U^* - U$. Refer to Fig. 5 again. Assume that nodes 8 and 6 are not neighbors in G, $I(8, 1) = \{5\}$,

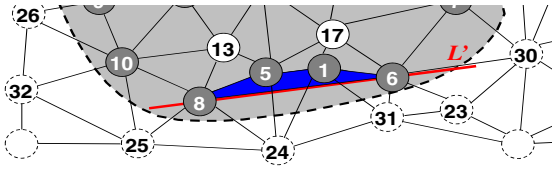

Fig. 5. Blue area denotes the enclosed region of $I(8, 6) \cup \{8, 6\}$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

and $I(5, 6) = \{1\}$. Then both $U \cup \{5\}$ and $U \cup \{1\}$ are unnecessary. But $U \cup \{5, 1\}$ is not unnecessary (because nodes 8 and 6 are not neighbors in G). Notice that if $U \cup \{5\}$ ($U \cup \{1\}$, respectively) is unnecessary, then neither node 8 nor node 1 (neither node 5 nor 6, respectively) belongs to U^*. In order to determine $U^* - U$ in a distributed manner, node priority is used to break such a tie. In this paper, the priority of a node u (i.e., $\epsilon(u)$) is assumed to be its ID. For improving the readability, all symbols used are summarized in Appendix A.

Summarized the discussion above, we have that every event node in S could locally determine whether it is a reporter or not by the following rule.

Unnecessary node determination rule:

An event node $u \in S$ belongs to U^* if there exist two event nodes in S, say v_α and v_β, such that all the following conditions are satisfied.

(1) Node u is located between v_α and v_β, i.e., $u \in I(v_\alpha, v_\beta)$.
(2) All nodes located between v_α and v_β are at the same part divided by L, where L denotes the line passing through v_α and v_β.
(3) No normal node is enclosed by $I(v_\alpha, v_\beta) \cup \{v_\alpha, v_\beta\}$.
(4) For each node w located between v_α and v_β, w and the neighboring normal node specified in the advertisement message sending by w are at different parts divided by L.
(5) For each node $w \in I(v_\alpha, v_\beta)$, the priority of w should be lower than v_α's and v_β's, i.e., $\epsilon(w) < \epsilon(v_\alpha)$ and $\epsilon(w) < \epsilon(v_\beta)$.
(6) Two arbitrary nodes w, x in $I(v_\alpha, v_\beta) \cup \{v_\alpha, v_\beta\}$ are mutual neighbors (i.e., $w \in N(x)$ and $x \in N(w)$).

Such an event node u is called to satisfy the unnecessary node determination rule with respect to v_α and v_β. Refer to Fig. 5. Assume that nodes 8 and 6 are not neighbors in G, $I(5, 6) = \{1\}$, and $I(8, 1) = \{5\}$. Since node IDs also represent node priorities, i.e., $\epsilon(8) > \epsilon(6) > \epsilon(5) > \epsilon(1)$, it is easy to check that node 1 satisfies the unnecessary node determination rule with respect to nodes 5 and 6, but node 5 does not satisfy the unnecessary node determination rule with respect to nodes 8 and 1.

Recall that each node collects its two-hop neighbor information in the preprocessing of the coarse reduction step. So, the unnecessary node determination rule can be performed by each event node in S. Let U^* be the union of U and the set of nodes satisfying the unnecessary node determination rule. It is proved that U^* is unnecessary in Appendix B.

Since U^* is unnecessary, a smaller sufficient set, S^*, which is the difference between the event set and U^* can be obtained. In our protocol, S^* also denotes the set of reporters.

Let the total number of role changes between event nodes and normal nodes be the *the magnitude of changes to the contour* (i.e., C). The communication cost of collaborative data processing phase is less than 2^*C. This is because the communication cost of the coarse reduction step which results from sending advertisement messages for role changes is less than 2^*C (i.e., the number of role changes between role 1 and remaining roles is C, and the number of role changes between roles 2, 3, and 4 is not greater than C) and no communication cost is required in the fine reduction step. Unnecessary node determination in the fine reduction step contributes to the computation cost of the collaborative data processing phase. The computation cost is $O(\sqrt{n}/2)^2 m \leq O(nC/4)$, where m is the number of reporters and n is the maximum number of neighbors of a node.

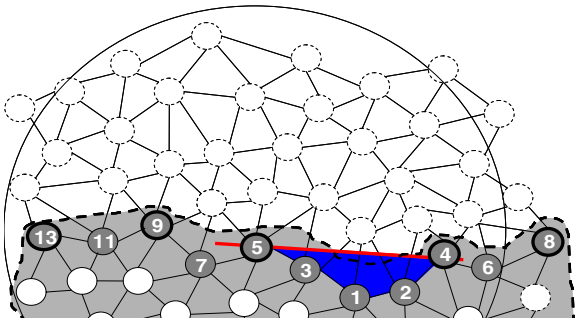

Fig. 6. Nodes 6 and 13 are two farthest candidate neighboring reporters of node 5.

3.2. Object location reporting phase

After the collaborative data processing phase, the object location reporting phase aims to periodically transmit the location information of reporters to the sink node. In order to reduce the traffic load or the communication cost, all reporters are organized into clusters. For the purpose of constructing clusters, some reporters are chosen as initiators. An initiator can construct a cluster by unicasting its information (i.e., priority, *etc.*) to its two nearest neighboring reporters, and each of its nearest neighboring reporters, say v, proceeds to forward this information to the other nearest neighboring reporters of v, and so on. A *nearest neighboring reporter* v of a reporter u is a u's neighbor in G satisfying that

(C1) v is a reporter, and
(C2) There is no other reporter located between reporters u and v.

An example is shown in Fig. 6. There, nodes 4, 5, 8, 9 and 13 are reporters and the big circle denotes the communication range of node 5. Clearly, nodes 4 and 9 are nearest neighboring reporters of 5. When a reporter receives the information of some initiator, it should join the cluster if it has not joined any cluster. For tie-break purposes, an initiator should have a higher priority than its nearest neighboring reporters. However, to reduce the communication cost, each reporter is not informed who are its nearest neighboring reporters (i.e., each node is not informed which neighbors satisfy the unnecessary node determination rule) in the collaborative data processing phase. In Fig. 6, when node 1 finds that it satisfies the unnecessary node determination rule with respect to one node pair, say (3, 2), it regards itself as unnecessary. For reducing the computation cost, node 1 stops applying the unnecessary node determination rule in the object location reporting phase. Similarly, nodes 2 and 3 may only know that they satisfy the unnecessary node determination rule with respect to node pairs (3, 4) and (5, 4), respectively. And each of nodes 5 and 4 regards itself as a reporter in the collaborative data processing phase. Keep in mind that they do not inform their neighbors whether they are reporters or not. In Section 3.2.1, a novel scheme is applied by reporters to locally determine the nearest neighboring reporters without message exchanges.

3.2.1. Nearest neighboring reporter identification

In this subsection, we devote our effort to determining the nearest neighboring reporters for each reporter. Recall that each node u has two-hop neighbor information and maintains the set of $N(u) \cap S$ by the aid of advertisement messages in the collaborative data processing phase. According to the definition of

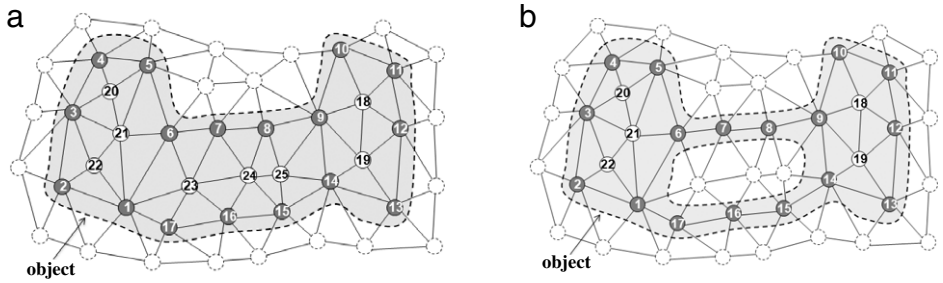

Fig. 7. An object in the network. (a) Without hole. (b) With hole.

nearest neighboring reporters (i.e., conditions (C1) and (C2)), every reporter can locally determine its nearest neighboring reporters by the following rule.

Nearest neighboring reporter identification rule

Each reporter u decides the event node $v \in N(u) \cap S$ as its nearest neighboring reporter if the following two conditions hold:

(1) v does not satisfy the unnecessary node determination rule with respect to u and some other node (i.e., (C1) holds).
(2) For each $w \in I(u, v)$, w should satisfy the unnecessary node determination rule with respect to u and v (i.e., (C2) holds).

In the following, we show how a reporter applies the nearest neighboring reporter identification rule to locally determine its nearest neighboring reporters. Refer to Fig. 6 again. Recall that nodes 5 and 4 are neighboring reporters of each other. When node 5 finds that it is a reporter, it picks up its farthest neighbors from the set of gray nodes (i.e., S), i.e., node 6, and determines whether node 6 is its neighboring reporter. According to the unnecessary node determination rule, if nodes 5 and 6 are neighboring reporters, then gray nodes located between nodes 5 and 6 (i.e., nodes 1, 2, 3, and 4) must satisfy the unnecessary node determination rule with respect to node pair (5, 6). In this case, node 5 finds that node 6 is not its neighboring reporter because node 4 does not satisfy the unnecessary node determination rule with respect to node pair (5, 6). Then node 5 picks up its second farthest neighbor from the set of gray nodes, i.e., node 4, and determines whether node 4 is its neighboring reporter. This time node 5 finds that node 4 is its neighboring reporter because all gray nodes located between nodes 5 and 4 (i.e., nodes 1, 2, and 3) satisfy the unnecessary node determination rule with respect to node pair (5, 4). Of course, node 4 performs a similar process as node 5 to determine that node 5 is its neighboring reporter. The correctness of the distributed nearest neighboring reporter identification algorithm is proved in Appendix C.

4. Non-ring-type objects

In the previous section, only ring-type objects are taken into consideration. In this section, we consider non-ring-type objects.

Refer to Fig. 7(a) and (b). There light gray areas denote the object's location and dark gray nodes, say nodes 1, 2, …, 17, are reporters. Notice that both objects in Fig. 7(a) and (b) have the same set of reporters. That is, these two cases are not distinguishable by the location information of reporters. In order to solve this problem, the following two definitions are needed.

Define a *branch node,* say v, to be an event node in S whose neighboring event nodes (in G^T) are all in S (i.e., $N^T(v) \cap$ event set $\subset S$). An example is shown in Fig. 7(b), gray nodes denote event nodes in S and white nodes denote event nodes not in S. By definition, nodes 7, 8, 15, 16 and 17 are branch nodes. Define a *join node* to be an event node in S which is not a branch node and has a neighboring branch node in G^T. For example, nodes 1, 6, 9 and 14 are join nodes in Fig. 7(b), they connect with branch nodes 17, 7, 8 and 15, respectively.

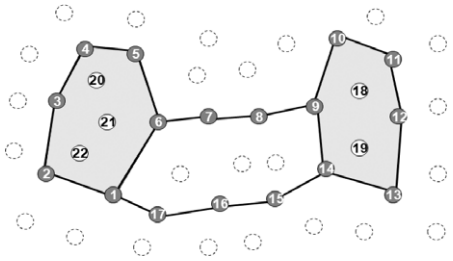

Fig. 8. An estimation of object in Fig. 7(b).

In order to obtain a more precise object contour, branch nodes and join nodes should be reporters even if they satisfy the unnecessary node determination rule. For this purpose, each branch node or join node, say u, re-informs its neighbors, i.e., $N(u)$, that its priority is a maximum in the coarse reduction step.

Similar with ring-type objects, S^* is the set of reporters for non-ring-type objects. However, S^* is not a sufficient set for a non-ring-type object (recall that a non-ring-type object does not have a sufficient set). Fortunately, object location can be estimated by the aid of location information of reporters and information on whether a reporter is a branch node or a join node. Consider the object in Fig. 7(b). By the aid of the information mentioned above, the sink node can determine an estimation of the object's location as in Fig. 8.

5. Performance evaluations

A simulator is implemented by NS-2 [23] to evaluate the performance of our proposed continuous object detection and tracking protocol. Sensors are randomly and uniformly deployed in a region of size 1000 m × 1000 m. The number of deployed sensors varies from 600 nodes to 1000 nodes with an interval of 100 nodes. The communication range of every sensor is 100 m. The node degree of deployed sensors varies from 19 to 31 with an interval of 3. The proposed protocol is compared with DCSODT presented in [11]. A continuous object is randomly placed in the sensor network, and its size varies from 200 m × 200 m to 400 m × 400 m.

The metrics for comparing performance are listed below:

- *Number of reporters*: A great number of reporters leads to high power consumption in the object's location reporting phase.
- *Data packet overhead*: It is concerned about the number of data packet exchanges (i.e., message exchange in sending the location information of reporters to the sink).
- *Control message overhead*: It is concerned with the total number of messages exchanged except data packet exchange.

5.1. Effects of node degree on the number of reporters

Fig. 9 shows that the number of reporters increases as the node degree increases. This is because more details of object's location

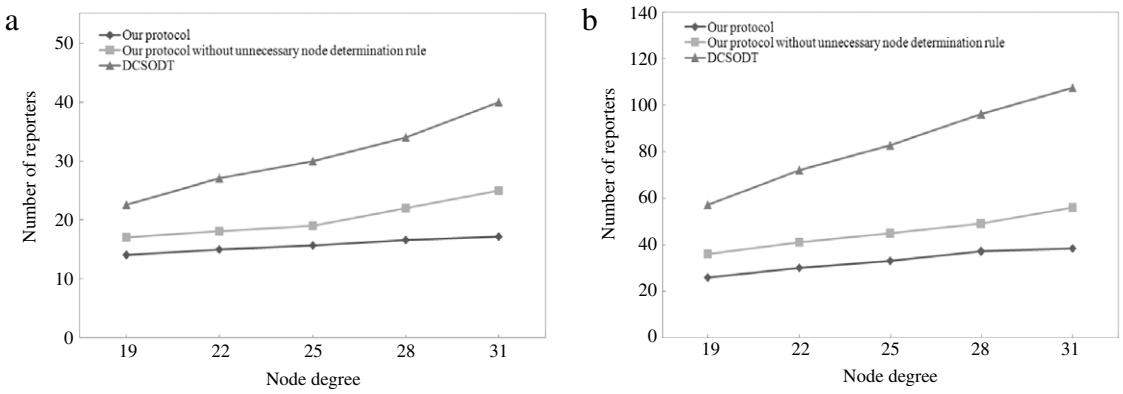

Fig. 9. Node degree and the number of reporters. (a) Object size is 200 m × 200 m. (b) Object size is 400 m × 400 m.

(object contour) are preserved by the location of event nodes when the node degree of the sensor network is high. Hence, more reporters are required to keep these details. It also shows that the number of reporters of the proposed protocol is obviously fewer that that of DCSODT no matter whether the continuous object has a large or a small size.

5.2. Effects of node degree on control message overhead

Consider the collaborative data processing phase. There are three kinds of control message in this phase: control message for collecting two-hop neighbor information, control messages for performing Delaunay triangulation, and advertisement messages about the sensor nodes' role change. The costs of the former two are ignored since they are performed once in the pre-processing (when there exist destroyed nodes in the sensor network, the cost of repairing the Delaunay triangulated graph can be found in Section 5.4). The latter exists in both our protocol and DCSODT. In the object location reporting phase, clustering methods are applied to both our protocol and DCSODT. According to the simulation result in Section 5.1, our protocol has obviously fewer reporters than DCSODT no matter how large the object size is. Since reporters constitute clusters, our protocol has a lower control message overhead on clustering. Besides, by the nearest neighboring reporter identification rule, the control message overhead is further reduced. So, our protocol has a lower control message overhead than the DCSODT in this phase.

Fig. 10 shows the simulation results of the total control message overhead for the collaborative data processing phase and the object's location reporting phase. There cluster size 2, 4, and 6 means that the number of hops from a cluster head to a cluster member in a cluster is not greater than 2, 4, and 6, respectively. In Fig. 10, when the cluster increases its size, the overhead of the control message in clustering is increased.

5.3. Effects of node degree on data packets overhead

When clusters are constructed, each cluster head aggregates the location information of its cluster members and finds a route to the sink by the geographic routing method [2] for transmitting aggregated location information. Fig. 11 shows the simulation results of the data packet overhead for clustering in different node degrees. Clearly, a small number of reporters leads to a lower data packet overhead. Therefore, the data packet overhead of our protocol is lower than the DCSODT. For a large-sized object, it is more obvious that the proposed protocol has a much lower data packet overhead than the DCSODT does, sketched in Fig. 11(b).

Fig. 11 also shows that large-sized clusters have a higher data packet overhead. In general, data aggregation by clustering can be divided into two stages, from members to cluster head and from cluster heads to sink. For large-sized clusters, cluster members close to the cluster head will relay many farther cluster members' data packets to the cluster head. So, the data packet overhead in the first stage increases as the cluster size increases. Conversely, since the number of cluster heads decreases as the cluster size increases, the data packet overhead in the second stage decreases as the cluster size increases. It is more obvious when the object is far apart from the sink. In our simulation, the distance between the object and the sink is not so far that the decrement of data packet overhead in the second stage is not larger than the increment of the data packet overhead in the first stage. Hence, large-sized clusters have a higher data packet overhead.

5.4. Effects of destroyed nodes

We first consider the effects of destroyed nodes on the Delaunay triangulated graph. Clearly, the Delaunay triangulated graph G^T is destroyed when there are destroyed nodes. Refer to [24], G^T could be constructed/repaired in a distributed manner. When there are m destroyed sensors in the sensor network, it requires $O(md^2 \log d)$ time and $O(md^2 \log d)$ control messages to repair a Delaunay triangulated graph, where d is the maximum node degree in the resulted Delaunay triangulated graph. Since the values of m and d are usually small, the cost of repairing a Delaunay triangulation is rather low.

Next we investigate the relationship between the number of destroyed nodes and the number of reporters, and the relationship between the number of destroyed nodes and the number of advertisement messages. Consider the case that destroyed sensors are not able to send advertisement messages to inform their neighbors of their new roles. The experimental environment is almost the same with previous experiments except that there are in total 800 sensor nodes, the node degree is 25, the continuous object is a square with size 400 m × 400 m and the speed is 5 m/s, and the number of destroyed sensor nodes varies from 0 to 10. Each experiment takes 50 s. Suppose that the last role a destroyed sensor plays is a normal node.

According to the experimental results, the number of reporters increases little as the number of destroyed sensors increases (see Fig. 12(a)). The number of reporters increases no matter whether destroyed sensors are located inside the object or at the object boundary. According to the simulation results, destroyed sensors inside the objects bring in more reporters than destroyed sensors at the object boundary. In Fig. 13, black nodes are destroyed, gray nodes are normal nodes, blue nodes are reporters, yellow nodes are event nodes but not reporters. Fig. 13(a) and Fig. 13(b) show the situation when the sensor network has no destroyed sensors and the situation when the sensor network has destroyed sensors,

Fig. 10. Node degree and Control message overhead. (a) Object size is 200 m × 200 m. (b) Object size is 400 m × 400 m.

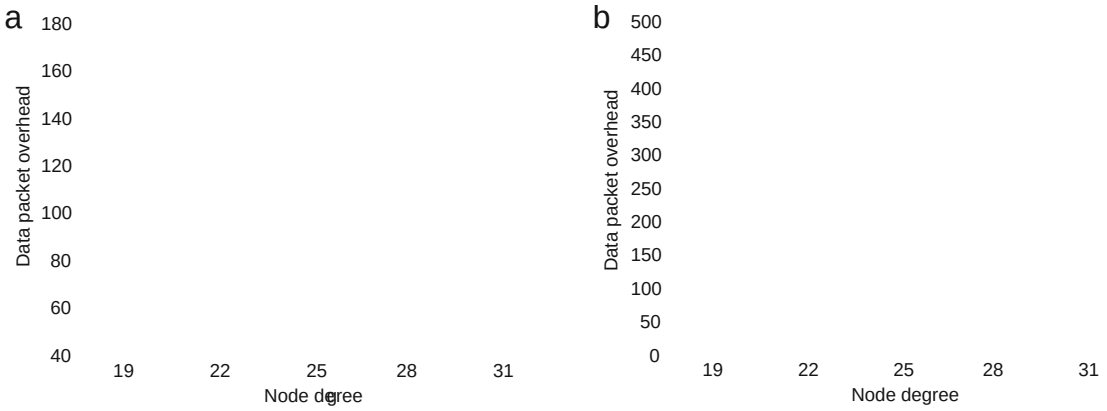

Fig. 11. Node degree and data packet overhead. (a) Object size is 200 m × 200 m. (b) Object size is 400 m × 400 m.

Fig. 12. Effects of destroyed nodes.

respectively. Refer to the black node at the upper left quarter in Fig. 13(b). Since it is inside the object, it brings in five reporters. Refer to the two black nodes at the lower right quarter in Fig. 13(b). Since they are at the boundary of the object, they bring in very few reporters.

Fig. 13(b) and (c) show the effect of destroyed sensors on our protocol and DCSODT, respectively. Since the number of destroyed nodes is usually small, the effect is not significant. Besides, the performance could be improved by applying some heuristic rules [6].

Fig. 12(b) shows that the number of advertisement messages decreases as the number of destroyed sensors increases. This because the destroyed sensors are not able to sense the existence of objects, i.e., no advertisement messages are sent to inform their own neighbors of their new roles.

Here we consider the effects of an object's speed on the number of reporters and on the number of advertisement messages. Intuitively, the number of reporters does not increase or decrease as the object's speed changes, while the total number of advertisement message increases as the speed of the object increases. This is because the number of reporters is determined by the object size, object shape, and node degree. And the total number of advertisement messages is determined by the number of sensor nodes' role changes.

5.5. Effects of distribution of sensor nodes

According to the experimental results in Sections 5.1 and 5.2, when the node degree of the sensor network is uniform, both the number of reporters and the number of advertisement messages increases little as the node degree increases. Here we consider the skewed distribution of nodes. The experimental environment is described below. The whole sensor network is divided into two sub-areas: inner area and outer area. The inner area is a square with size 300 m × 300 m at the center of the sensor network, and the outer area denotes the remaining part. The node degree is 19 or 31. We consider two distributions: distribution 1: the node degree of the inner area is 19 and the node degree of the outer