CODA: A Continuous Object Detection and Tracking Algorithm for Wireless Ad Hoc Sensor Networks

Wang-Rong Chang[1], Hui-Tang Lin[1,2], and Zong-Zhi Cheng[2]
[1]Department of Electrical Engineering, National Cheng Kung University, Taiwan (R.O.C.)
[2]Institute of Computer and Communication Engineering, National Cheng Kung University, Taiwan (R.O.C.)
E-mails: {n2892138, htlin, q3694104}@mail.ncku.edu.tw;

***Abstract*-Wireless sensor networks make possible many new applications in a wide range of application domains. One of the primary applications of such networks is the detection and tracking of continuously moving objects, such as wild fires, biochemical materials, and so forth. This study supports such applications by developing a Continuous Object Detection and tracking Algorithm, designated as CODA, based on a hybrid static/dynamic clustering technique. The CODA mechanism enables each sensor node to detect and track the moving boundaries of objects in the sensing field. The numerical results obtained using a Qualnet simulator confirm the effectiveness and robustness of the proposed approach.**

***Keywords*-Cluster Scheme; Wireless Ad Hoc Sensor Network; Continuous Object; Detection; Tracking;**

I. INTRODUCTION

The miniaturization and integration of electronic components made possible by modern Micro-Electro-Mechanical Systems (MEMs) techniques, coupled with advances in communication protocols and technologies, have enabled the design and implementation of large-scale Wireless Sensor Networks (WSNs) comprising a huge number of sensor nodes [1]. Typically, these nodes are equipped with sensing, communicating, and data processing units such that they can collect, exchange, and process information relating to their environment. With their ability to operate autonomously within environments, WSNs have been widely deployed in a variety of military and civil applications, including remote surveillance, disaster relief monitoring, temperature, and noise level sensing, traffic volume measurement, and so on.

One of the most common application scenarios in WSNs is that of target detection and tracking [1-6]. In such applications, thousands of sensors are deployed within a large-scale area to monitor the intrusion or diffusion of specific objects or materials of interest, such as enemy vehicles, wild fires, bio-chemical materials, and so forth. A Dynamic Convoy Tree-based Collaboration (DCTC) framework was reported in [1][2] to detect and track mobile targets in WSNs. DCTC is based on a tree structure, known as a *convoy tree*, consisting of all the sensor nodes around the moving target. The tree is dynamically configured by adding and pruning sensor nodes as the target moves through the sensing field. In [3] and [4], Xu *et al*. presented a Prediction-based Energy Saving (PES) scheme and a Dual Prediction-based Reporting (DPR) mechanism, respectively, for Object Tracking Sensor Networks (OTSNs). The two schemes were designed to achieve energy savings by enabling the sensor nodes and base station to intelligently predict the future movement of the mobile object. In [5], Chen *et al*. devised and evaluated a fully decentralized, light-weight, dynamic clustering algorithm for acoustic target tracking. In the proposed approach, an acoustic target is detected and subsequently tracked by forming a sensor cluster whenever a designated Cluster-Head (CH) sensor within a static backbone network determines that the acoustic signal length exceeds a predetermined threshold. In such an event, the CH broadcasts a solicitation packet inviting sensors in the immediate vicinity to join the cluster and to provide their own sensing information. After receiving a sufficient number of replies, the CH estimates the location of the acoustic target using a localization technique and then issues a report to the subscribers.

The schemes presented above are aimed specifically at the tracking of individual targets, e.g., people, animals, vehicles, and so forth. However, in many cases, it is desirable to track the spread of continuous objects or phenomena, such as wild fires, toxic gases, and so on. Continuous objects differ from multiple individual targets in that they are continuously distributed across a region and usually cover a large area rather than discretely distributed over the area of interest. A fundamental characteristic of continuous objects is that they tend to diffuse, increase in size, change in shape, or even split into multiple relatively smaller continuous objects over time. In [6], Xiang *et al.* explored the feasibility of using WSNs to detect and track continuous objects, and proposed a Dynamic Clustering Scheme (DCS) to monitor movements of their boundaries and to fuse and disseminate the boundary information. In the DCS approach, the sensors automatically declare themselves as *boundary sensors* when they detect the presence of the object in their vicinity but have one or more one-hop neighbors which do not detect the same object. A clustered structure is then constructed by grouping these *boundary sensors* into a small number of dynamic clusters. Within each cluster, the head fuses the local boundary information and transmits this information to the sink. Once the sink has received boundary information from each of the dynamic clusters, it estimates the global boundary of the continuous object of interest. Adopting this approach yields a significant reduction in the overall communication overheads compared to schemes in which the sinks communicate directly with the individual *boundary sensors*.

Although the DCS approach provides substantial energy savings, it still requires a significant volume of communications since the *boundary sensors* are identified by requiring each sensor which detects the emergence of the object to communicate with all of its one-hop neighbors to determine whether or not they too detect the same object. Consequently, this strategy results in a very high number of message exchanges, particularly when the boundary of the object moves at a rapid pace. Furthermore, the number of CHs

1-4244-1457-1/08/$25.00 © IEEE
This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

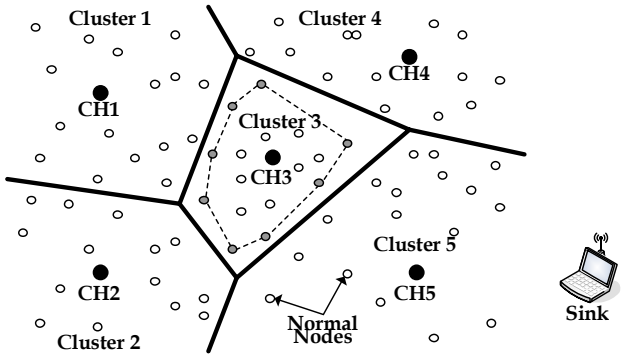

Fig. 1. Static clustering scheme.

required increases as the span of the continuous object increases. Therefore, developing suitable strategies for determining an appropriate number of CHs and then choosing these CHs from the *boundary sensor* set are challenging yet crucially important problems in DCS-based sensing networks.

This study develops a target tracking mechanism, designated as the Continuous Object Detection and tracking Algorithm (CODA), based on a hybrid static/dynamic clustering scheme for the automatic detection and tracking of continuous objects. In developing the CODA mechanism, a static backbone comprising a designated number of static clusters is constructed during the initial network deployment stage. In each static cluster, any sensors detecting the object in their vicinity transmit the detection information directly to the CH. Upon receiving this detection information, the CH executes a local boundary estimation function to determine the *boundary sensors* of the continuous object boundary which lies within its cluster. After forming these *boundary sensors* into a dynamic cluster, it then sends the boundary information of this dynamic cluster to designated sinks. As in the DCS scheme, the sinks compile the local information received from the various CHs to evaluate the global boundary of the continuous object. Compared with the DCS mechanism proposed in [6], CODA has two principal advantages. First, the *boundary sensors* of the continuous object are identified by the CHs of the static clusters rather than via a process of message exchange among the local sensors, and thus the communication overhead is substantially reduced. Second, selecting the CHs in each dynamic cluster from the *boundary sensor* set is not necessary. Consequently, an explicit header election scheme is not required and excessive message exchanges are avoided.

The remainder of this paper is organized as follows. Section II briefly highlights the major challenges involved in detecting and tracking continuous objects. Section III describes the proposed CODA protocol. Section IV presents the simulation results. Finally, Section V draws some brief conclusions.

II. Issues of Detecting Continuous Objects

A typical example of continuous object detection and tracking using a WSN is that in which a toxic gas or biochemical agent is deliberately or inadvertently released and it is necessary to monitor its diffusion and direction of travel as it diffuses under the effects of wind currents. One approach is simply to allow any sensors which detect the object to periodically send their sensing data and identification to one or more sinks. However, this approach induces a high communication overhead since all sensors detecting the object are required to transmit their sensing data and identification. The problem is further exacerbated when the continuous object spans a wide geographical area or travels with a high velocity. A far more efficient approach is to detect and track only the boundary of the continuous object. Broadly speaking, continuous object tracking in WSNs involves two critical operations, namely

- ***Monitoring***: the sensors in the WSN detect and track the movement of the mobile continuous object.
- ***Reporting***: the sinks collect location information only from those sensors which detect the object and are located at its boundary.

Since continuous objects tend not only to change in shape, but also to drift under the influence of the wind, tracking the boundary movement in real-time is a challenging problem. Nevertheless, the capability to do so is essential to facilitate the deployment of environmental monitoring systems for public health and security purposes. Accordingly, the following section discusses the features of the CODA scheme presented in this study which enable a continuous object to be detected and tracked with a high degree of precision and minimal communication costs.

III. Description of CODA Mechanism

Monitoring continuous objects without incurring excessive communication costs requires an efficient target detection mechanism. Collaborative data processing in targeting-centric sensor networks is generally performed using a cluster architecture. Generally speaking, clustering mechanisms can be classified as either static or dynamic. In static clustering schemes (e.g., LEACH [7], TEEN [8], PEGASIS [9], and so forth), the nodes in the WSN are separated into static clusters at the time the network is initially deployed. The attributes of each cluster, e.g., its size, the area it covers, its members, and so on, remain unchanged (or lowly change) over time. However, such clustering schemes are unsuited to the tracking of continuous objects since the fixed membership of each cluster cannot adapt to highly dynamic scenarios [5]. By contrast, dynamic clustering schemes trigger the formation of a cluster only when an event of interest is detected, and therefore represent a more suitable approach for continuous object tracking. However, high communication costs may be incurred in such schemes if the tracked object moves rapidly across the sensing field since the sensors are required to exchange messages more frequently in order to construct appropriate dynamic clusters to keep up with the object's progress. To address these deficiencies of the static and dynamic clustering schemes, the CODA mechanism proposed in this study applies a hybrid static/dynamic clustering scheme to achieve continuous object detection and tracking with low communication overheads.

A. Static Clustering Scheme

The basis of the CODA detection and tracking mechanism is a backbone network comprising static clusters constructed during the initial network deployment stage. As shown in Fig. 1, one node within each cluster is nominated as the CH and plays the role of a local controller, while the remainder serves as regular sensing nodes. The normal nodes sense their

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

```
1: Input: a set of points S = {P = (P.x,P.y)}
2: Select the rightmost lowest point P0 in S.
3: Sort S angularly about P0 as a center.
4:       For ties, discard the closer points.
5: Let P[N] be the sorted array of points.
6: Push P[0]=P0 and P[1] onto a stack W.
7: while i < N
8: {
9:       Let PT1 = the top point on W
10:      Let PT2 = the second top point on W
11:      if (P[i] is strictly left of the line PT2 to PT1) {
12:         Push P[i] onto W
13:         i++   // increment
14:      }
15:      else
16:         Pop the top point PT1 off the stack
17: }
18: Output: W = the convex hull of S.
```

Fig. 2. Pseudo-code of Graham Scan algorithm.

immediate environment and send or relay the sensing data to the CH. The CH not only generates sensing data of its own, but also collects the data sent from the normal nodes in the cluster and fuses and transfers this information to the sink in accordance with a reactive (e.g., [10]) or proactive (e.g., [11]) routing protocol. Note that this study does not prescribe a specific clustering scheme to construct the static clusters. In practice, the network can be constructed using any existing static clustering scheme, such as LEACH, TEEN, PEGASIS, and so forth.

As shown in Fig. 1, an assumption is made that each static cluster is assigned a unique cluster ID by its CH. When a new node joins the cluster, a message exchange takes place between itself and the CH during which it is informed of the cluster ID. It is assumed that each sensor is aware of its location and sends this information to the CH when it first joins the cluster. Upon receiving the location information from all of the normal nodes in the static cluster, the CH determines those sensors residing at the boundary of the cluster and then sends them a message to notify them that they are the *boundary sensors* of the static cluster. The *boundary sensors* among the normal nodes in the cluster can be determined by resolving the so-called convex hull problem [12], as defined in the following:

Definition 1(Convex Hull). For a subset S of d-dimensional space R^d, the convex hull *conv(S)* is defined as the smallest convex set in R^d containing S.

Definition 2(Convex Hull Problem). The convex hull problem is defined as the determination of *conv(S)* for a given finite set of n points $S = \{P^1, P^2, \ldots, P^n\}$ in R^d.

The convex hull *conv(S)* is generally solved using the so-called "Graham Scan" algorithm [13]. Due to length constraints, the full details of this algorithm are not discussed in this study. However, the pseudo-code is presented in Fig. 2.

In the current case, the normal nodes in a static cluster represent the subset S and the CH distinguishes the *boundary sensors* among them by solving the corresponding convex hull problem using the Graham Scan algorithm. Note that hereafter, the *boundary sensors* are referred to as Static-cluster Boundary-sensors (SBs), while the remaining nodes are referred to as Static-cluster Inner-sensors (SIs).

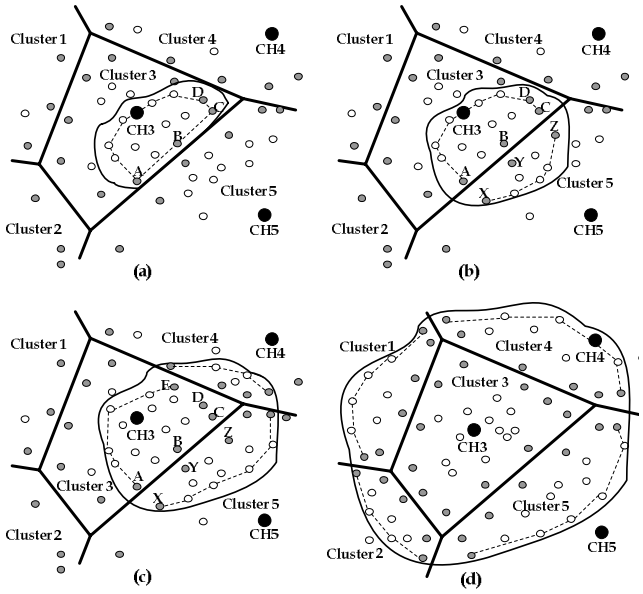

Fig. 3. Boundary tracking of continuous object: (a) object shape lies within a single static cluster; (b) object shape spread across two static clusters; (c) object shape spread across three static clusters; and (d) object shape fully covers a static cluster.

B. Boundary Detection

This subsection describes the use of the static clustering scheme presented above to perform the boundary tracking of a continuous object. For simplicity, the discussions consider a network system comprising just five static clusters, as shown in Fig. 3. Note that the shaded gray nodes represent the SBs of the corresponding cluster. In tracking the boundary of the object, two control messages are used to transmit the detection information to the CH whenever the object is detected.

- ***"SENSE"* message**: This message is used only by the SIs and is sent to the CH once the SIs detect the target object.
- ***"REPORT"* message**: This message is used only by the SBs. It is in the form of a bitmap in which the n^{th} bit is set to '1' if the detected object is identified within Cluster n. Having detected the object, a SB communicates with all of its one-hop neighboring SBs in other clusters to query their object detection information. Once the SB has received this information, it sets the corresponding bit(s) of its bitmap to '1' and sends the REPORT message to the CH such that the CH can determine all of the static clusters within which the target object has spread.

Depending on the number of static clusters to which the object spreads, four particular scenarios can be identified, as described in the following.

CASE I: Object Within Single Static Cluster

Fig. 3(a) shows the scenario in which the object (indicated by the solid curve) is located entirely within a single cluster (Cluster 3). In this case, after sensing the continuous object, the SIs within the object boundary broadcast SENSE messages to notify CH3 that they have detected the object. Meanwhile, when the SBs in Cluster 3 (i.e., Sensors A, B, C and D) detect the object, they query the object detection information of their

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

one-hop neighboring SBs in Clusters 4 and 5 and determine that the detected object does not extend beyond the boundary of Cluster 3. Accordingly, the four SBs set the 3rd bit of their respective bitmaps to '1' and send REPORT messages to CH3. Upon receiving these messages and examining the bitmap information contained within them, CH3 determines that the detected object is currently spread only within its own cluster. Using a similar technique to that employed to distinguish the *boundary sensors* of the static cluster (see Subsection A), CH3 treats all of the sensors within the object boundary (including both the SIs and the SBs) as a subset *S* and applies the Graham Scan algorithm to determine the corresponding *conv(S)* (indicated by the dotted line in Fig. 3(a)).

CASE II: Object Crosses Two Static Clusters

Fig. 3(b) illustrates the scenario in which the continuous object straddles two static clusters, i.e., Clusters 3 and 5. Note that since the sensors in the two clusters have an equivalent functionality when detecting the object, the following discussions focus arbitrarily on the sensors deployed in Cluster 3 for illustration purposes.

As described in Case I above, the SIs in Cluster 3 transmit SENSE messages to CH 3 for as long as they detect the continuous object. Meanwhile, SBs A, B, and C query their one-hop neighboring SBs, i.e., X, Y and Z, respectively, to see whether they too can detect the same object. In this case, these nodes report that the object has indeed spread to Cluster 5, and thus SBs A, B and C set the 3rd and 5th bits of their respective bitmaps to '1' and then send their REPORT messages to CH3. (Note that SB D queries its neighbor in Cluster 4, and based upon the reply sets only the 3rd bit in the bitmap to "1") When CH3 receives these REPORT messages, it learns that the detected object is not confined solely within its own cluster, but has also spread to Cluster 5. However, its role is simply to estimate the portion of the object boundary lying within the confines of its own cluster. It achieves this by evaluating the total *conv(S)* among the SI and SB nodes in Cluster 3 which have detected the object and then eliminating the redundant sensors in accordance with the following procedure:

- Calculate the straight line distance between each pair of SBs along the common border between Clusters 3 and 5 which have detected the object, i.e., A and B, B and C, and A and C in the current example.
- Eliminate all SB(s) other than the two SBs constituting the sensor pair having the greatest separation distance.

In the current example, the distance between SBs A and C is greater than that between SBs A and B or B and C, respectively, and thus SB B is eliminated. CH 5 applies an equivalent procedure to determine the portion of the boundary of the moving object within Cluster 5. As a result, CH 3 and CH 5 collectively have a knowledge of the total boundary position of the continuous object.

CASE III: Object Crosses more than Two Static Clusters

Fig. 3(c) shows the case in which the continuous object extends across three static clusters. Since the CODA scheme operates independently in each cluster, the following discussions focus on the sensors deployed within Cluster 3.

In contrast to Case II described above, in this scenario, SB C receives detection information from one-hop neighboring sensors not only in Cluster 5, but also in Cluster 4. Therefore, it sets the 3rd, 4th and 5th bits of its bitmap to '1' and then sends the REPORT message to CH 3. Once it receives REPORT messages from all of its *boundary sensors*, CH 3 perceives that the detected object is spread across three static clusters. As described in Case II, CH3 estimates only the portion of the object boundary lying within its own cluster. The estimation procedure comprises three steps. **Step (A):** Distinguish the *boundary sensors* among all of the sensors which have detected the object by solving the *conv(S)* problem. **Step (B):** Identify and eliminate the redundant sensors in *conv(S)*. In the current example, Sensors A, B, and C (or Sensors C, D, and E) all know that the continuous object spreads across two static clusters, i.e., Clusters 3 and 5 (or Clusters 3 and 4), respectively. CH3 can then apply the same approach as that described in Case II above to remove the redundant sensors (i.e., Sensors B and D in the current example). **Step (C):** Eliminate any *non-boundary sensor(s)* of the continuous object in *conv(S)*. In the current example, it is easily determined that SB C cannot be a *boundary sensor* of the continuous object since it is not only a SB in Cluster 3, but also has the information that the shape of the detected object spreads across more than two static clusters. Therefore, CH3 further eliminates SB C from *conv(S)*.

As shown in Fig. 3(c), the procedure described above results in the identification of the portion of the object boundary within Cluster 3. CH4 and CH5 execute identical procedures to identify the portions of the boundary curve within Clusters 4 and 5, respectively. As a result, the entire boundary of the continuous object is known. Note that the scenario in which the continuous object spreads across more than 3 static clusters is not discussed explicitly here since the three-step estimation procedure described above is applicable to the general case in which the object spreads over multiple static clusters.

CASE IV: Object Completely Covers Static Clusters

The final scenario concerns the case in which the continuous object completely covers one or more than one static clusters. For simplicity, the discussions consider a continuous object which completely covers Cluster 3 (see Fig. 3(d)). In this scenario, all of the SBs in Cluster 3 detect the object and learn from their one-hop neighboring SBs that the scope of the detected object extends into the neighboring clusters. Thus, once CH3 has received the REPORT messages from all of the SBs within its cluster it perceives that its cluster is completely covered. In other words, CH3 confirms that all of the nodes in Cluster 3 are *non-boundary sensors* of the detected object and therefore takes no further action. However, by inspecting the bitmaps they receive from their SBs, CHs 1, 2, 4, and 5 perceive that a portion of the object boundary lies within their cluster and thus employ the three-step estimation process described in Case III to estimate its location.

C. Boundary Recognition

After the portion of the object boundary within each static cluster has been identified, each CH organizes the *boundary sensors* within its cluster into a "*dynamic cluster*". Note that the term "*dynamic*" is used here to reflect the fact that the members of the cluster may be amended as the boundary of the continuous object moves. Each CH fuses the boundary information in a compact data format and then relays it to the sink. By compiling the integrated boundary information received from each of the CHs in the network, the sink is able to determine the entire boundary of the continuous object.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

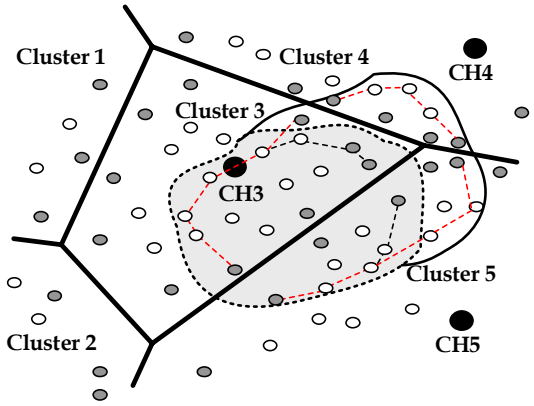

Fig. 4. Updating boundary sensors when tracking moving boundary.

Clearly, the process described above in which each CH is nominated during the initial network deployment phase and subsequently aggregates the boundary information within its own cluster for onward transmission to the sink incurs a significantly lower communication cost than schemes in which the CHs are determined dynamically within each dynamic cluster [6] or in which the sink collects the detection information from each sensor directly.

D. Boundary Tracking

In the hybrid static/dynamic clustering method described above, the *boundary sensors* of a continuous object are distinguished and assigned to *dynamic clusters* in accordance with their respective locations. The CHs of the static clusters then send the integrated local boundary information to the sink. Even though the continuous object may move and its boundary profile may change, the location of the boundary can still be estimated provided that it remains within the sensing ranges of the identified *boundary sensors*. However, when a portion of the object boundary moves out of the sensing range of the current *boundary sensors* (as a result of diffusion, for example), the membership of the *dynamic cluster* must be updated to indicate the new set of *boundary sensors* responsible for sensing the new boundary location. Fig. 4 illustrates a change in the position of the object boundary and the subsequent changes in the boundary profile portions in clusters 3, 4 and 5, respectively. Note that in this figure, the gray region represents the location of the continuous object in the previous time slot and the circles connected via black dashed lines represent the old boundary sensors. Meanwhile, the curve marked using a solid line represents the new object boundary. In the current time slot, the SIs and SBs in each static cluster which sense the object for the first time send SENSE and REPORT messages, respectively, to their CH. (Note that each CH maintains the reported detection information from one time slot to the next). Upon receiving these messages and examining the bitmap information they contain, each CH ascertains which of the four scenarios the continuous object now belongs to and then identifies the nodes within its cluster which represent the new *boundary sensors* of the object by executing the appropriate boundary estimation method (see Subsection B). (Note that if a sensor detects the disappearance of the object in its local area at the current time slot, it knows that the boundary of the object moved through its detection area during the past time slot. In such a situation, the sensor is permitted to transmit an "UNDETECTED" control message to notify its CH of updating the reported detection information) The circles connected via red dashed lines in each static cluster in Fig. 4 indicate the new *boundary sensors* of the object as determined by CHs 3, 4 and 5, respectively.

Note that in the procedure described above, only those sensors which detect diffusion of the object (i.e., those sensors which detect the object for the first time in the current time slot) or detect the disappearance of the object send control messages to the CH. In other words, only those nodes which have new information to report consume energy in sending a message to the CH. This policy conserves the total energy within the network and extends the overall lifetime of the network as a result.

IV. Performance Evaluation

The performance of the CODA was evaluated by performing a series of simulations using the Qualnet simulator [14].

A. Simulation Model and Evaluation Metrics

The simulations assumed a 100 m x 100 m sensing field within which the sensors were randomly deployed. The radio range of each sensor was assumed to be 6 m. Two continuous objects were tracked within the sensing field, namely a rectangle with a width and length of 20 m and 10 m, respectively, and a circle with a radius of 10 m. It was assumed that the rectangular and circular objects were initially centered at coordinates of (50, 60) and (50, 30), respectively. Finally, the sink was located at coordinates (0, 0).

The performance of the CODA scheme was evaluated using two metrics, namely the message cost and the precision of the estimated boundary. The message cost was defined as the total number of messages broadcast in establishing the boundary nodes, integrating the local boundary information and then disseminating this information to the sink. Since this study focuses specifically on the communication costs incurred in detecting and tracking the boundary of a continuous object, an assumption is made that the sensors already know their routes to the CHs and each CH knows its route to the sink. Hence, the message cost data presented in the following sections exclude the costs of route discovering in the network. In the simulations of the message cost, the sensing field was assumed to contain a total of 500 sensors. In determining the precision of the estimated boundary, the simulations computed the number of sensors within the actual object boundary (m) and then divided this number by the number of sensors enclosed by the estimated boundary l. Clearly, the higher the value of the ratio l/m, the greater the precision of the estimated boundary.

B. Numerical Results

The performance of the CODA scheme was evaluated by comparing its message cost and boundary estimation performance with that of the DCS mechanism. Note that the performance results presented for the DCS scheme in the following are taken directly from [6]. In Figs. 5-9, the width and length of the rectangular object and the radius of the circular object increase by 0.5 m in each time slot. As a result, after 20 time slots, the objects grow to such an extent that they overlap one another and form a single object.

Fig. 5 compares the variation of the number of control messages broadcast over time under the CODA and DCS object detection schemes. In the DCS scheme, the simulations

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

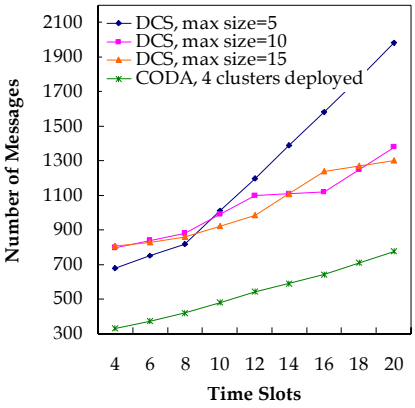

Fig. 5. Variation of message cost over time under CODA with 4 static clusters and DCS with various maximum cluster sizes

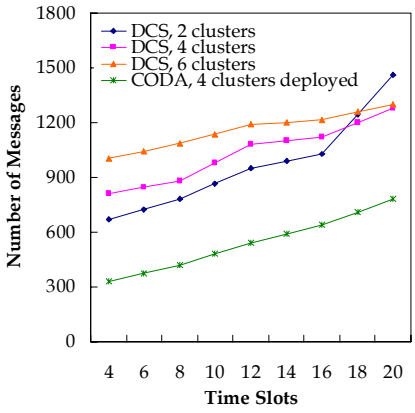

Fig. 6. Variation of message cost over time under CODA with 4 static clusters and DCS with various numbers of dynamic clusters.

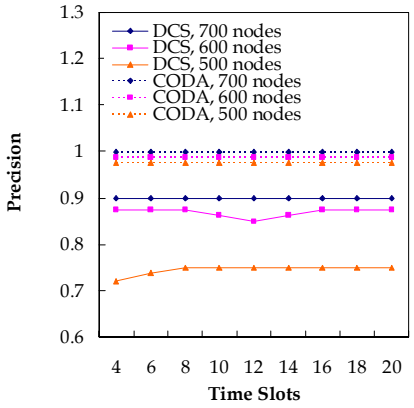

Fig. 7. Variation of boundary estimation precision over time as function of sensor density under CODA and DCS schemes.

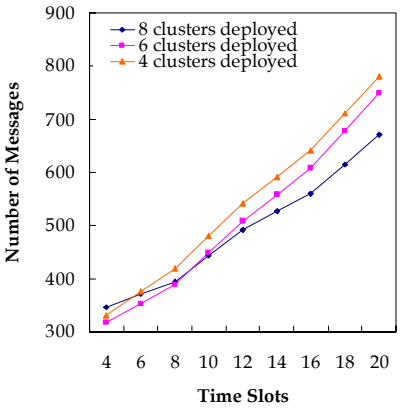

Fig. 8. Variation of number of control messages over time under CODA as function of number of static clusters.

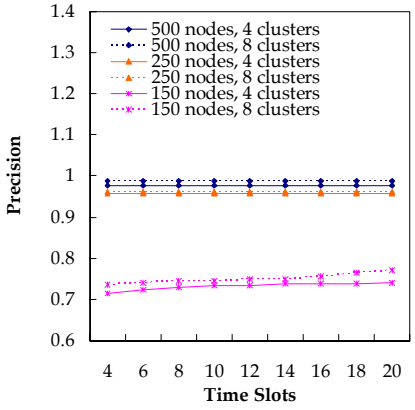

Fig. 9. Variation of boundary estimation precision over time as function of sensor density and number of static clusters in CODA scheme.

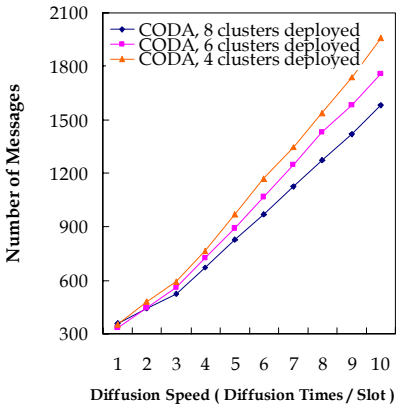

Fig. 10. Variation of number of control messages with rate of diffusion as function of number of static clusters in CODA scheme.

consider dynamic clusters comprising a maximum number of 5, 10, and 15 boundary sensors, respectively, while in CODA, the network comprises 4 static clusters. Comparing the two schemes, it is evident that CODA results in a considerably lower communication overhead due to its policy of using the CHs in the pre-deployed static clusters to determine the *boundary sensors* of the two moving objects and to construct *dynamic clusters*. By contrast, in DCS, the *boundary sensors* are determined by requiring every sensor detecting the emergence of the object to communicate with all of its one-hop neighboring sensors. Dynamic clusters are then built by performing an extensive message exchange among all of the *boundary sensors*. As a result, the total message cost of the DCS scheme is very high.

Fig. 6 illustrates the variation of the communication costs under DCS with a pre-defined number of dynamic clusters and under CODA with four static clusters. In this case, no limit is imposed in DCS regarding the size of each dynamic cluster. The results show that irrespective of the number of dynamic clusters in DCS, CODA yields a significantly lower communication overhead. Again, this result is to be expected since the fundamental principle of the CODA scheme is to use the CHs in each static cluster to estimate the object boundaries, rather than requiring all of the sensors detecting the object to exchange messages with their one-hop neighbors. In addition, the dynamic clusters are built directly by the individual CHs rather than by requiring the *boundary sensors* to communicate with one another as in the DCS approach.

Fig. 7 compares the estimation precision of the CODA and DCS in networks with various sensor densities. In general, the results show that both schemes retain an approximately constant estimation performance as the object size increases. For any given time slot, it can be seen that the precision with which the two schemes estimate the boundary increases as the node density increases. Furthermore, it is observed that CODA consistently outperforms DCS due to its policy of locating the boundary by solving the convex hull problem.

Fig. 8 presents the variation over time of the communication cost incurred under CODA when the scheme is implemented using 4, 6, and 8 static clusters, respectively. Comparing the three curves, it can be seen that following 8 time slots, the network constructed with 8 static clusters consistently yields

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.

the lowest communication cost. This result arises since when the object size increases to a certain degree and the number of static clusters is large, the size of each static cluster is very small, and thus the internal communication costs are reduced.

Fig. 9 shows the boundary estimation performance of the CODA scheme as a function of the network sensor density and the number of static clusters. It can be seen that even for low sensor densities (i.e., 150 nodes within the 100 m x 100 m sensing field), CODA still achieves a boundary estimation precision of greater than 0.7. In other words, CODA provides a robust boundary estimation performance. Furthermore, the results indicate that for a constant sensor network density, CODA's boundary estimation performance improves as the number of static clusters within the network is increased.

Finally, Fig. 10 illustrates the variation of the number of control messages broadcast under CODA with the diffusion speed in networks with 4, 6, and 8 static clusters, respectively. Note that in these simulations, the width and length of the rectangular object and the radius of the circle are increased by 0.5 m in each period of diffusion. In other words, increasing the number of diffusion periods per time slot results in an increased rate of diffusion. Note that the simulation results represent the average value of data collected over a total of 5 time slots. It can be seen that when the objects are diffused more than twice in each time slot, the lowest communication cost is obtained when the network is constructed using 8 static clusters. The reason for this is that on average less communications occur in each static cluster when tracking diffused objects with a large size and the size of each static cluster is very small. As a result, the CODA protocol provides a significant performance improvement as the number of static clusters is increased.

V. CONCLUSIONS

This study has presented a Continuous Object Detection and tracking Algorithm, designated as CODA, for the monitoring of continuous objects in a WSN. CODA reduces the communication overheads incurred in estimating the boundary of a moving continuous object by employing CH in the static clusters formed when deploying the network to estimate the boundary profile within each cluster and to communicate this information directly to the sink. The results of a series of numerical simulations have demonstrated that CODA incurs lower communication costs than DCS and achieves greater boundary estimation precision irrespective of the size of the continuous object and the sensor network density.

ACKNOWLEDGEMENT

The authors would like to thank the National Science Council, Taiwan, R.O.C., for supporting this research under grant NSC 95-2221-E-006-369.

REFERENCES

[1] W. Zhang and G. Cao, "DCTC: Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks," *IEEE Transaction on Wireless Communication,* vol. 3, no. 5, Sept. 2004.

[2] W. Zhang and G. Cao,"Optimizing Tree Reconfiguration for Mobile Target Tracking in Sensor Networks" in *Proc. IEEE INFOCOM*, vol. 4, pp. 2434-2445, Mar. 2004.

[3] Yingqi Xu, Winter J, and Wang-Chien Lee, "Prediction-Based Strategies for Energy Saving in Object Tracking Sensor Networks" in *Proc. IEEE MDM*, pp. 346-357, Jan. 2004.

[4] Yingqi Xu, Winter J, and Wang-Chien Lee, "Dual Prediction-Based Reporting for Object Tracking Sensor Networks," in *Proc. IEEE MOBIQUITOUS*, pp. 154-163, Aug. 2004.

[5] Wei-Peng Chen, Jennifer C. Hou, and Lui Sha, "Dynamic Clustering for Acoustic Target Tracking in Wireless Sensor Networks," *IEEE Trans. on Mobile Computing*, vol. 3, issue. 3, pp. 258-271, Aug. 2004.

[6] Xiang Ji, Hongyuan Zha, John J.Metzner, and George Kesidis ,"Dynamic Cluster Structure for Object Detection and Tracking in Wireless Ad-hoc Sensor Networks," in *Proc. IEEE ICC*, vol. 7, pp. 3807-3811, June. 2004.

[7] Handy M.J, Haase M, and Timmermann D,"Low Energy Adaptive Clustering Hierarchy with Deterministic Cluster-head Selection," in *Proc. IEEE MWCN*, pp. 368-372, Sept. 2002.

[8] Manjeshwar A., and Agrawal D.P.,"TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks," in *Proc. IEEE Int'l Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, pp. 2009-2015, Apr. 2001.

[9] Lindsey S., and Raghavendra C.S.," PEGASIS: Power-efficient Gathering in Sensor Information Systems," in *Proc. IEEE AERO,* vol.3, pp. 3-1125-3-1130, March. 2002.

[10] Naserian, M., Tepe,K.E., and Tarique,M., "Routing Overhead Analysis for Reactive Routing Protocols in Wireless Ad Hoc Networks," in *Proc. IEEE WIMOB,* vol. 3, pp. 87-92, Aug. 2005.

[11] Seung Jun Baek and Gustavo de Veciana, "Spatial Energy Balancing through Proactive Multipath Routing in Wirelss Multihop Networks," *IEEE/ACM Transaction on Networking,* vol. 15, pp. 93-104, Feb. 2007.

[12] W. Eddy, "A New Convex Hull Algorithm for Planar Sets", ACM Trans. Math. Software, vol. 3, no. 4, pp. 398-403, 1977.

[13] Joseph O'Rourke, Computational Geometry in C (2nd Edition), Chap. 3 "Convex Hulls in 2D", 1998.

[14] Qualnet Simulator Home. http://www.scalable-networks.com/

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE CCNC 2008 proceedings.