Distributed Group Management in Sensor Networks: Algorithms and Applications to Localization and Tracking

Juan Liu, Jie Liu, James Reich, Patrick Cheung and Feng Zhao[†]

Palo Alto Research Center

3333 Coyote Hill Road, Palo Alto, CA94304, USA

Email: {*juan.liu, jieliu, jreich, pcheung, zhao*}*@parc.com*

Contact Author:

Juan Liu, tel: (650) 812-4828; fax: (650) 812-4334

Email: *juan.liu@parc.com*

[†] **Acknowledgment.** This work is partially supported by the DARPA Sensor Information Technology program under Contract F30602-00-C-0139. We are indebted to Prof. John Heidemann and Dr. Fabio Silva of USC/ISI for their help on directed diffusion. We would also like to thank our colleagues Dave Duff, Jaewon Shin, and Rebecca Hinden for their assistance in carrying out the experiment.

ABSTRACT

The tradeoff between performance and scalability is a fundamental issue in distributed sensor networks. In this paper, we propose a novel scheme to efficiently organize and utilize network resources for target localization. Motivated by the essential role of geographic proximity in sensing, sensors are organized into geographically local collaborative groups. In a target tracking context, we present a dynamic group management method to initiate and maintain multiple tracks in a distributed manner. Collaborative groups are formed, each responsible for tracking a single target. The sensor nodes within a group coordinate their behavior using geographically-limited message passing. Mechanisms such as these for managing local collaborations are essential building blocks for scalable sensor network applications.

KEYWORDS

Group management, detection, target localization, target tracking, collaborative information processing, distributed sensor network

Introduction

The study of distributed sensor networks is emerging as an exciting interdisciplinary research area, including aspects of signal processing, networking, distributed algorithms, and MEMS sensor technology. A wireless sensor network can be easily deployed in places where there is no a priori sensing infrastructure. This flexibility has led to an increasing interest in using these networks for large-scale applications such as environmental monitoring, security surveillance, and battlefield awareness. In contrast to traditional centralized sensor array processing where all processing occurs on a central processor, sensor networks distribute the computation among sensor units. Each sensor unit acquires local, partial, and relatively crude information from its surroundings. By exploiting the sensor network's spatial coverage and multiplicity of sensing modalities, the network can arrive at a good global estimate.

A key issue in distributed sensor networks is scalability, in both energy and spatial dimensions. Desirably, sensor networks must simultaneously track multiple phenomena, working within tight communication bandwidth, energy, and processing speed limits. Thus, it is critical to distribute the workload in an equitable way across only the "relevant" sensors, and leave other sensors available for other tasks. In this paper, we focus on target tracking applications, and present a scalable track initiation and maintenance scheme based on collaboration between sensors in local groups. The scheme is built on the basis of our previous work (Zhao et al., 2002; Liu et al., 2003; Chu et al., 2002) on single target tracking. To set up the proper context, we first briefly introduce the tracking problem.

Target tracking using distributed sensor networks

Tracking is one of the major uses of sensor networks, essential in many commercial and military applications, such as traffic monitoring, facility security, and battlefield situational awareness.

Assume a two-dimensional sensor field and a point target moving in it. The goal of a tracking system is to estimate the target location $x^{(t)}$ based on sensor measurements.

Each sensor node has a local measurement of the target over time. To be able to incorporate measurements from heterogeneous sensors, we use a statistical fusion method, where all sensor measurements are combined probabilistically in a common state space, based on the observational likelihood $p(z^{(t)}|x^{(t)})$. For sensor i, $\mathbf{z}_i^{(t)} = \{z_i^{(0)}, z_i^{(1)}, \cdots, z_i^{(t)}\}$ represents its local measurements, where the superscript indexes time. Let $\overline{z^{(t)}} = \{\mathbf{z}_1^{(t)}, \mathbf{z}_2^{(t)}, \cdots, \mathbf{z}_n^{(t)}\}$, where n is the number of nodes. The sensor network collectively computes the posterior belief $p(x^{(t)}|\overline{z^{(t)}})$ through Bayesian inference.

Noticing that the majority of the sensor measurements has little contribution to the global estimation of the target trajectory, we designed in (Liu et al., 2003) a leader-based tracking scheme to minimize resource usage. At any time instant t, there is only one leader k, which takes a new measurement $z_k^{(t)}$ and updates its estimate of the target location using sequential Bayesian filtering (Stark and Woods, 1994). Based on this updated belief, the leader *selects* the most "informative" sensor (according to some criterion measuring information gain) from its neighborhood, and passes it the updated belief $p(x^{(t)}|\overline{z^{(t)}})$. This new sensor becomes the next leader at time $t + \delta$ (where δ is the communication delay), the previous leader returns to an idle state, and the process of sensing, estimation, and leader selection repeats. We call this approach *information-driven sensor query* (IDSQ).

Organization of this paper

In theory, the IDSQ algorithm is scalable since all sensors except one (the leader) are in the idle state, free to pick up other tasks. Thus, the number of active nodes is proportional only to the number of targets, and is independent of the size of the sensor network. However, without proper ways of initiating new target tracks and maintaining local collaboration groups, scalability cannot be achieved in practice. We need a mechanism to decide who is responsible for initialization and how to handle contention between multiple sensors detecting the same target. Furthermore, the co-existence of multiple tracks leads to the problem of track maintenance. Special care must be taken when two targets come into the vicinity of each other.

This paper establishes our solution to track initiation and maintenance problems in the following steps. Sec. 1 motivates the idea of organizing sensors into geographically-based local collaborative groups. Collectively, a group is responsible for the initiation and maintenance of one track (presumably corresponding to a single target). Sec. 2 describes in detail how a track is initiated, including detection and an efficient leader election scheme. Sec. 3 describes the algorithm for track maintenance, handling situations such as merging and splitting tracks. Overall, sensor nodes within the group coordinate their behavior by passing messages. Sec. 4 covers an implementation of the algorithms and an experiment on target tracking. Finally, we discuss in Sec. 5 the implications of distributed group management for more general sensor network applications.

1. Motivating Geographically-Based Group Management

A fundamental problem in sensor network design is the tradeoff between performance and scalability. Traditional centralized schemes move all sensor data to a central site and process it. While this is provably optimal in estimation performance, it exhibits poor scalability. The complexity of computation and communication grows rapidly with the total number of sensors, making these schemes impractical for sensor-rich systems. An interesting idea for balancing performance and scalability is to organize sensors into collaborative groups. Take tracking of some physical phenomena as an example. Sensors which jointly provide the best information about a phenomenon should form a group. Sensors which are less informative, or whose data are redundant, could be left out. By limiting the collaboration to a small number of sensors in a limited area, communication

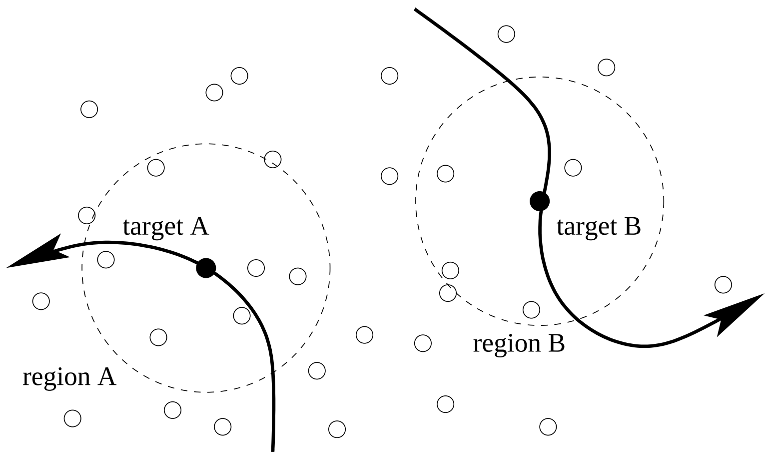


Figure 1. Geographically-based collaborative groups. The small circles are sensor nodes. The nodes inside a specified geographical region (e.g., region A or B) form a collaborative group.

and computation are made independent of the size of network. Since the group contains the most informative sensors, impact on the tracking performance will be limited.

In practical sensor network applications, the effects of physical phenomena usually attenuate with distance, producing a decreasing signal-to-noise ratio (SNR) and lower-quality observations. This points toward the idea of geographically-based collaborative groups. In the target tracking problem, for example, we can organize the sensor network into geographical regions, as illustrated in Fig. 1. Sensors in the region around target A are responsible for tracking A, and the region around B handles B. Partitioning the network into local regions assigns network resources according to the potential contributions of individual sensors.

Furthermore, the physical phenomena being sensed change over time. This implies that the collaborative groups also need to be dynamic. As the target moves, the local region must move with it. Sensor nodes that were previously outside of the group may join the group, and current members may drop out. This requires some method for managing group membership dynamically.

Geographically-based group initiation and management have to be achieved by a light-weight protocol distributed on all sensor nodes. The protocol needs to be powerful enough to handle complex situations such as those where data from multiple leaders are contending for processing resources, and be robust enough to tolerate poor communication qualities such as out-of-order delivery and lost or delayed packets. In addition, the propagation region of group management messages should be restrained to only the relevant nodes without flooding the entire network. This is not trivial considering that the group membership is dynamic as the targets move, and that the network is formed in an *ad hoc* way such that no nodes have the knowledge of the global network topology. The difficulties are tackled in our approach by taking advantages of two facts: 1) a leader-based tracking algorithm where at any time each group has a unique leader who knows the geographical region of the collaboration; and 2) recent advances in geographical-based network routing (Ko and Vaidya, 1999; Yu et al., 2001) that do not require the leader to know the exact members of its group.

2. Distributed Detection and Track Initiation

Consider a distributed sensor network monitoring a large field. When there is no target in the field, the sensor nodes should be in an energy-saving mode. They should watch for possible targets using only low-cost computation and a minimal amount of communication. When a target enters the sensor field, the nodes need to elect a leader and give it an initial belief state $p(x^{(0)}|z^{(0)})$. In this section, we describe an efficient geographically-based group formation scheme to accomplish this task. The algorithm is sketched out below, with more details in Secs. 2.1 — 2.4:

1. Each individual sensor performs a stand-alone detection by comparing the measurement with a precomputed threshold corresponding to the likelihood ratio test (LRT) described in Sec. 2.1.

2. Nodes with detections form a collaborative group and elect a single leader.

3. The leader suppresses all nodes in the collaborative group from further detection in order to prevent creation of multiple tracks for the same target.

4. The leader initializes the belief state $p(x^{(0)}|z^{(0)})$ and starts the tracking algorithm.

Throughout the algorithm's development, we assume that the nodes are globally time-synchronized up to some reasonable (e.g., sub-second) accuracy and communication between nodes are relatively reliable, though the group management scheme is designed with some robustness against occasional packet losses. We also assume nodes are aware of their one-hop neighborhood.

2.1. Target detection on individual nodes

In many practical applications, activity within the network is sparse, and most of the sensor nodes spend the majority of their time in the low-duty-cycle detection mode. Only when a target enters the region does the sensor field become actively involved in tracking. Since detection is the most frequent mode, the detection algorithm must be light-weight in terms of computation and communication. This helps maintain the longevity and stealth of the sensor network.

In distributed sensor networks, one can combine the measurements of multiple sensors to reach a detection decision, as done in (Li et al., 2002), but such approaches require communication between multiple sensor nodes all the time. The communication cost is significant because of the frequency of the detection operation. Here we take a simple standalone target detection approach, where each individual sensor detects independently of the others. The group collaboration scheme takes effects only after interesting phenomena have been detected. Much of the benefit of multi-node detections can also be realized by a two-stage detection process, with the first stage being single-node detections set conservatively to minimize missed targets, and the second stage verifying these detections using a multi-node process to minimize false alarms.

In single node detection, each node needs a decision rule to decide whether a target is present within some pre-specified detection range R_{detect} of itself. For this task, a common approach is LRT, which compares two hypotheses:

H_0: target not present, or outside of the detection range, i.e., $d(x, L_{sensor}) \geq R_{detect}$, where x is the target location, L_{sensor} is the sensor location, and $d(\cdot,\cdot)$ measures the Euclidean distance. The possibility of target not present is equivalent to $d(x, L_{sensor}) = \infty$, hence is included in this hypothesis.

H_1: target present in the detection range, i.e., $d(x, L_{sensor}) < R_{detect}$.

Assuming the two hypotheses are equally probable (i.e., no prior knowledge about whether the target is present or not), the decision rule takes the form:

$$p(z|H_0) \underset{H_1}{\overset{H_0}{\gtrless}} p(z|H_1), \tag{1}$$

i.e., detection is declared if the presence hypothesis is more likely than the absence hypothesis. This decision rule guarantees the smallest probability of error. Besides the standard LRT, other decision rules are also applicable and have similar forms. Depending on the application requirements, one might use the Neyman-Pearson rule (see, e.g., (Poor, 1994)), which maximizes the detection probability while keeping the false alarm probability below a specified value, or the minimax decision rule (Poor, 1994), which is more conservative and optimizes for worst case scenarios.

Let us illustrate a LRT using a sensing model similar to that in (Liu et al., 2003). Assume a sensor network consists of microphone-based acoustic sensors. The sound wave received at the microphone takes the form:

$$f(t) = \frac{S(t-\tau)}{d(x, L_{sensor})} + n(t). \tag{2}$$

where t indexes time, $S(t)$ is the wave emitted by the target, τ is the wave propagation delay, and $n(t)$ is the measurement noise. The model is justifiable from the physics of wave propagation, assuming it is lossless and isotropic (Kinsler et al., 1999). We further assume that signal and noise are statistically independent. The signal has energy denoted as E_s, and the noise sequence $n(t)$ is white with zero mean and some known variance σ_n^2.

Acoustic energy sensors compute the sound energy $z = \frac{1}{N}\sum_{t=1}^{N} |f(t)|^2$, where N is the buffer size. Based on z, the sensor decides whether a target is present. Under the sensing model described above, by the Central Limit Theorem (see, e.g., (Stark and Woods, 1994)), the observation $p(z|x)$ is approximately Gaussian with parameters:

- mean= $E_s/d^2 + \sigma_n^2$.
- variance $= 2\sigma_n^4/N + 4 \cdot (E_s/d^2) \cdot \sigma_n^2/N$.

Under this observational model, the decision rule (1) boils down to a simple comparison of z to a decision threshold τ: if $z > \tau$, the sensor declares a target detection; otherwise no detection. The decision rule formalizes the intuition that when the perceived sound is loud enough, there is probably a target nearby.

The threshold τ is the dividing point which satisfies

$$p(\tau|H_0) \ = \ p(\tau|H_1). \tag{3}$$

It is computed numerically. From $p(z|x)$, one can compute the likelihoods $p(z|H_0)$ and $p(z|H_1)$ by numerical integration. Fig. 2 plots the likelihoods. It can be shown that $p(z|H_0)$ and $p(z|H_1)$ intersect at only one point, which is τ. Although the computation of τ is nontrivial, it only needs to be computed infrequently – at deployment time if the observation model is stationary, or during idle periods for background noise levels which change slowly. During detection, each sensor node periodically checks for detection simply by taking a measurement and comparing to the precomputed τ.

2.2. Group formation and leader election

The single node detection scheme described above ignores the correlation between sensor measurements. It is very likely that multiple nodes detect a single target at roughly the same time. However, the leader-based tracking algorithm suggests that a single leader should be sufficient for the tracking of any individual target. Multiple sensor nodes with detections cause contention for leadership. In this section, we describe a geographically-based group management scheme which resolves contention and elects a single leader via message exchange.

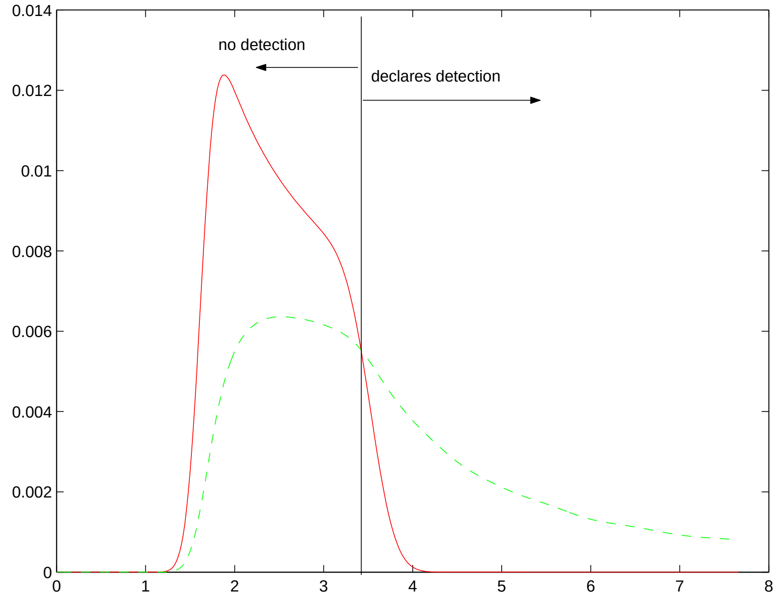


Figure 2. Single node LRT. The horizontal axis is the energy measurement z, and the vertical axis is the value of the hypothesis likelihood functions $p(z|H_0)$ (the solid curve) and $p(z|H_1)$ (the dashed curve).

First consider the ideal initialization condition: we have a sensor network covering a field in which the target has never appeared before. If sound propagates isotropically and attenuates monotonically with distance, the sensors physically closer to the target are more likely to detect than the sensors far away. One can compute an "alarm region", similar to a 3-σ region of a Gaussian distribution, such that most (e.g., 99%) of sensors with detections fall in the region. This is illustrated in Fig. 3. Sensor nodes are marked with small circles; the dark ones have detected a target. Assume the target is located at x (marked with a "+" in the figure), the alarm region is a disk centered at x with some radius R, where R is determined by the observation model. In practice, we use R_{detect} plus some moderate margin (to account for target motion during the sample period) as our choice of R.

Ideally, nodes in the alarm region should collaborate together to resolve their contention and elect a single leader from the region. However, the exact location of the alarm region is unknown since the target position x is unknown. Each node with a detection only knows that the target is within R distance of it, and a possible competitor could be another distance R from the target.

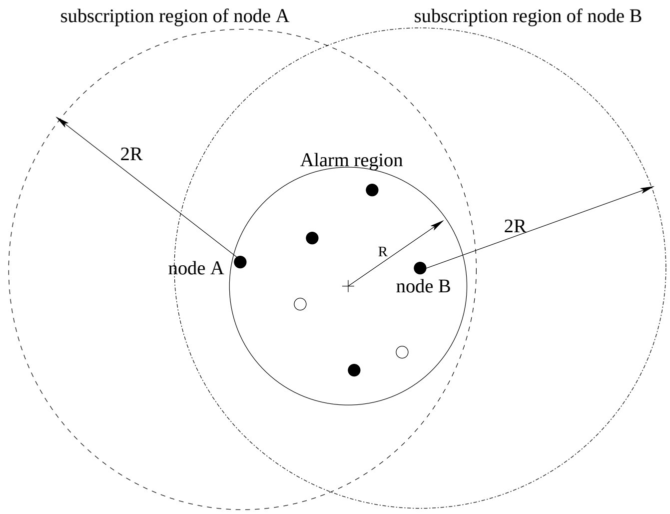


Figure 3. Detection and collaborative regions

Thus, in the absence of a "message center", a node notifies all nodes within a radius $2R$ of itself, which are potential "competitors" for leadership, of its detection.

Upon detection, each node broadcasts a `DETECTION` message to all nodes in this alarm region containing a time stamp recording when the detection is declared, and the likelihood ratio $p(z|H_1)/p(z|H_0)$. The higher this ratio, the more confident the detecting node is of its detection. We rely on the routing mechanisms to effectively limit the propagation of the detection messages to the specified region, a capability of critical importance for the algorithm to be scalable.

After sending out its own detection message, the node checks all detection packets received within an interval of t_{comm}. The value of t_{comm} should be long enough for all messages to reach their destination, yet not too long so that the target can be considered approximately stationary. These messages are then compared with the node's own detection. The node winning this election then becomes leader immediately, with no need for further confirmation. The election procedure is as follows:

– If none of the messages are timestamped earlier than the node's own detection, the node declares itself leader.

– If there are one or more messages with an earlier time stamp, the node knows that it is not the leader.

– If none of the messages contains earlier timestamps, but some message contains a time stamp identical to the node's detection time, the node compares the likelihood ratio. If the node's likelihood ratio is higher, the node becomes the leader.

Ideally, this algorithm will elect only one leader per target. In real networks, this algorithm is imperfect due to unreliable wireless links, and in some circumstances, multiple leaders may be elected. For example, if the `DETECTION` packet with the earliest detection time stamp fails to reach all the destination nodes, multiple nodes may find that they are the "earliest" detection and each may initiate a track. Since these tracks correspond to the same target, it is likely that they will collide with each other in the near future. This calls for methods to merge redundant tracks. Merging tracks is handled by the track maintenance scheme discussed in Sec. 3.

2.3. Suppression within the collaborative group

Once the leader is elected, it initializes a belief state $p(x^{(0)}|z^{(0)})$ as a uniform disk of radius R centered at its own location. The disk contains the true target location with high probability. This belief provides a starting point for the tracking algorithm.

The leader plays a key role in maintaining the collaborative group. As the target moves, the sensors which did not previously detect may begin detecting. These sensors are potential sources of contention. The system uses `SUPPRESSION` messages to minimize this. Basically, a `SUPPRESSION` message is a claim of group membership. The leader sends out `SUPPRESSION` messages to notify the recipient nodes to abandon detection and join the group. The message goes out to a region known as the suppression region, which should contain potential sources of contention. Assuming the actual target position is contained in the belief state, the suppression region should cover all

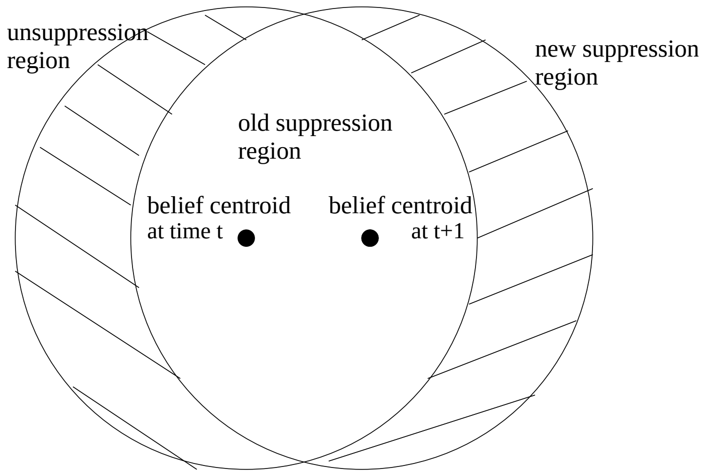


Figure 4. Suppression and unsuppression regions.

locations within R distance to the belief state. The actual implementation of this may depend on the belief representation used. In the case of the grid-based nonparametric representation used in (Liu et al., 2003), this region was found by starting with the bounding box containing all probability grids above a preset threshold level and adding margins of size R to all sides. Alternatively, one could identify a region containing the target with a specified probability and add margins of size R to that. The key factor is that the region must contain, with high probability, all nodes which might detect the target. In the special case of the original detection, as discussed in Sec. 2.2, the initial belief is a radius R disk centered at the leader, hence the suppression region is initially a concentric disk of radius $2R$.

As leadership moves in the network and the belief state is refined by successive measurements, the suppression region changes, and the group membership needs to be updated. Fig. 4 shows the two suppression regions at time t and $t + 1$. The two regions are not identical, but overlap. We can further reduce network traffic by only notifying the delta–regions, that is, the regions containing nodes which are added to or removed from the group. Three geographical regions need to be handled separately:

- The unsuppression region which contains nodes who are suppressed at t but not $t + 1$. This is pictured in Fig. 4 as the crescent-shaped region on the left hand side. The leader sends to this

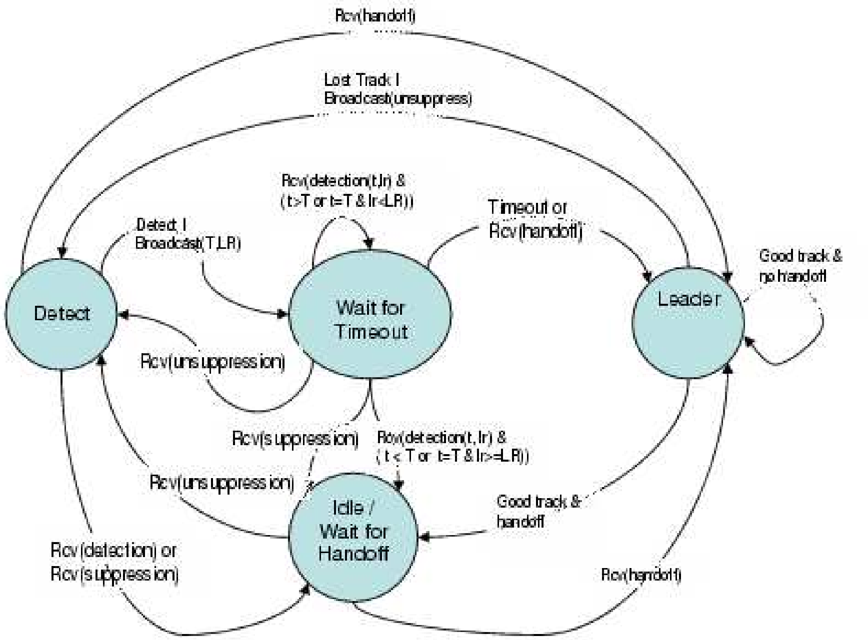


Figure 5. Finite state machine on nodes

region an UNSUPPRESSION message which basically reverses the effect of the SUPPRESSION message, equivalent to a dismissal from the collaborative group. Nodes receiving the UNSUPPRESSION message are freed, and go back to detection.

- The new suppression region which contains nodes who were not suppressed at t but suppressed at $t+1$. This is pictured in Fig. 4 as the crescent-shaped region on the right hand side. The leader sends SUPPRESSION message to claim membership.

- The region which sees no changes in group membership. No messages need to be sent to this region. The nodes in this region remain suppressed.

2.4. Group management process on nodes

The protocol described in Secs. 2.2 and 2.3 can be implemented on each sensor node using a finite state machine. Fig. 5 shows the group management process on each node. The actual implementation was more complicated in order to be robust against packet loss and out-of-sequence message arrivals.

The node has four states:

- *Detecting*: the node is not in any collaborative group, and periodically monitors its measurement for detection of possible targets.
- *Leader*: the node takes measurements, updates the track and the collaborative group.
- *Idle*: the node belongs to a collaborative group, and is passively waiting for a handoff from the leader.
- *Waiting-for-time-out*: intermediate states waiting for potential detections to arrive from other nodes.

There are four types of messages in the process. In addition to `DETECTION`, `SUPPRESSION`, and `UNSUPPRESSION` messages described in previous sections, `HANDOFF` messages carry the belief state used in tracking. They contain a time stamp, a belief state, the sender and the receiver's IDs, and a flag indicating if the track is successful or lost. A track is considered successful if the uncertainty of the track is under some specified tolerance level, and lost if otherwise. All nodes in the collaborative group corresponding to a lost track dismiss their membership and restart detection immediately.

3. Distributed Track Maintenance

With collaborative group management, each group associated with tracking of a single target. The co-existence of multiple tracks in the network can be readily handled as long as the the tracks are far apart and the collaborative regions are non-overlapping.

In practice, however, collisions between tracks are possible. For example, as briefly discussed in Sec. 2.2, redundant tracks corresponding to the same target are very likely to collide as the tracking algorithm advances. In tracking of multiple targets, targets crossing each other's path will cause the collaborative regions to collide. In these cases, nodes in the overlapped collaborative regions need to resolve the ambiguity of which leader to follow, especially when the multiple leaders dictate conflicting actions. There are numerous ways collisions can be handled. Here we describe a simple method for maintenance and management of multiple tracks.

First, in order for the tracks to be distinguishable, each track is assigned a unique ID. A simple choice is the time stamp (in microseconds) when the track was initiated. This choice does not require global knowledge shared throughout the network beyond rough time synchronization. The chance of multiple tracks being assigned the same ID is very small. The ID is carried along with the track and shared among the nodes in the collaborative group. All messages originating from the group are tagged with it. When a node receives a message, by examining the ID, it knows which group (and hence which track) the message refers to.

Now consider a node which belongs to multiple collaborative groups. Each node keeps track of its multiple membership based on the received `SUPPRESSION` and `UNSUPPRESSION` messages. A non-leader node ("follower" in the group) can be suppressed by any leader, but freeing it requires `UNSUPPRESSION` messages from all the local leaders. In other words, a node is free only when no leader claims ownership over it.

For a leader node, a received `SUPPRESSION` message with a different ID than its own is a clear indication of multiple groups colliding. Without the help of a suitable target classification method, the nodes cannot tell whether the collision is due to multiple targets crossing over, or redundant tracks for the same target merging. Furthermore, the maintenance of overlapping tracks requires source separation and data association, which are in general notorious inverse problems and hard to implement in distributed networks. In view of the difficulties, we propose a simple track merging

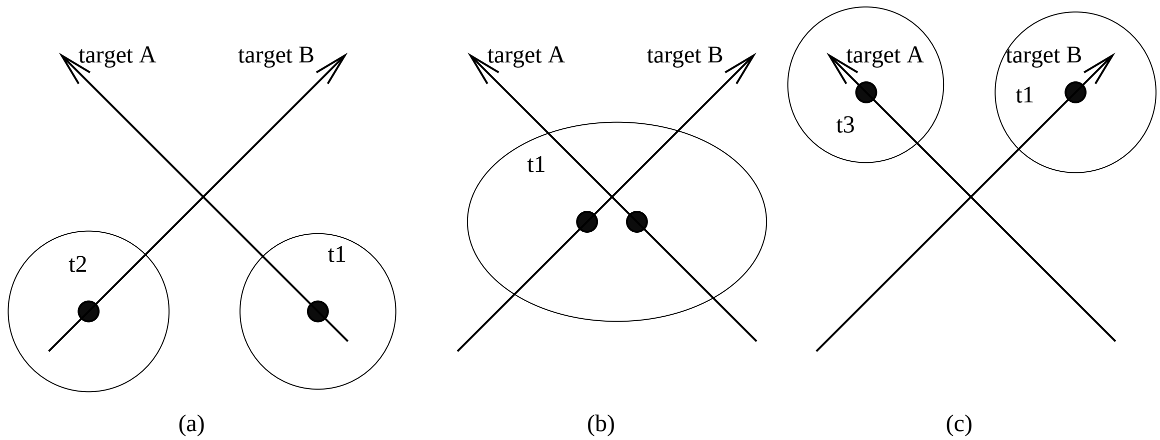


Figure 6. Track merging and splitting when multiple targets cross over. The collaborative groups are plotted as circles or ellipsoids. The labels $t1$, $t2$, and $t3$ are track IDs.

approach: One of the tracks survives; others are dropped. The collaborative groups merge together into a single group.

To decide which track to retain, each leader compares the ID of the incoming `SUPPRESSION` message, $t_{suppression}$, with its own, t_{leader}. We refer to the track corresponding to the incoming `SUPPRESSION` message as the incoming track. Between the incoming track and the track the leader currently has, the older one is retained. This is based on the intuition that an older track has already incorporated many measurements, hence is in general more accurate and reliable. The leader performs a comparison:

– If $t_{suppression} < t_{leader}$, i.e., the incoming track is older, the leader drops its own track, and relays the incoming `SUPPRESSION` message to its collaborative group, then gives up leadership. By this message, the two collaborative groups merge into one, obeying the leader of the incoming track.

– If $t_{suppression} \geq t_{leader}$, the leader's track survives. The leader sends a `SUPPRESSION` message to the leader of the incoming track.

This mechanism works well in merging multiple tracks corresponding to a single target. In the case where two (or more) targets approach each other closely, it basically tracks the superposition of the two targets as if the two targets could be regarded as a single “virtual” target. Without an accurate source separation scheme in place, the tracking algorithm is unable to tell the two targets apart. Once the targets separate, the second target will be re-detected as a new target. Fig. 6 illustrates this merging and splitting of tracks. As targets A and B approach each other, their groups merge, and then separate again. This example shows that track merging and splitting enables the tracking of multiple targets, but cannot maintain the identities of either targets.

Alternatively, we can assign a new group ID when multiple groups merge into one. A time-contiguous series of location estimates with a consistent identity is considered as a “tracklet”. For example, Fig. 6c contains four tracklets, two before the merging and two after. We can reacquire the target identity of each tracklet using classification schemes, and assemble tracklets into complete tracks.

The distributed track initiation and management scheme, combined with the leader-based tracking algorithm described in (Liu et al., 2003), forms a scalable system. The system works well in tracking multiple targets when the targets are not interfering (i.e., far apart), and can recover from inter-target interference once the targets move apart.

4. Experiment

We built a sensor network for multi-target tracking using the group management scheme. The sensor nodes in the experiment consists of 17 WINSNG 2.0 sensor nodes designed and manufactured by Sensoria Corp. Each node is essentially a Hitachi SH-4 based Linux PC with acoustic sensor inputs. Two type of sensors are used:

- Acoustic energy sensors. These output sound energy over a 256-sample window and estimate target distance based on the physics of sound attenuation.

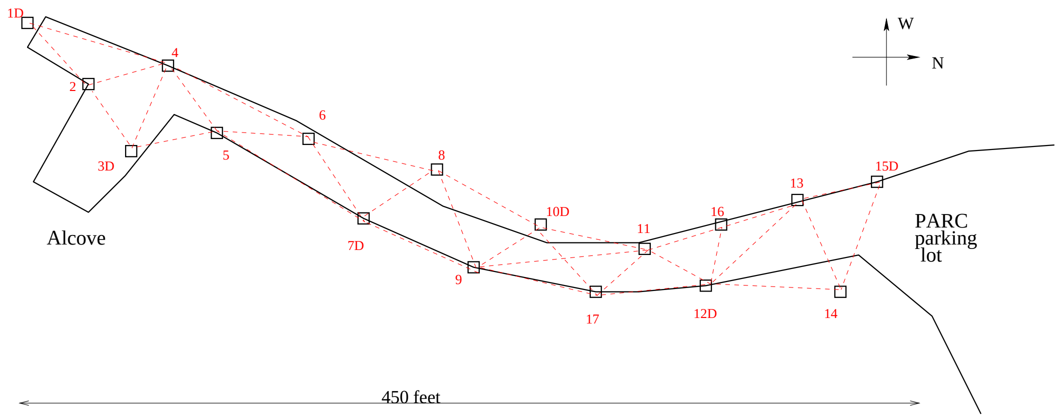


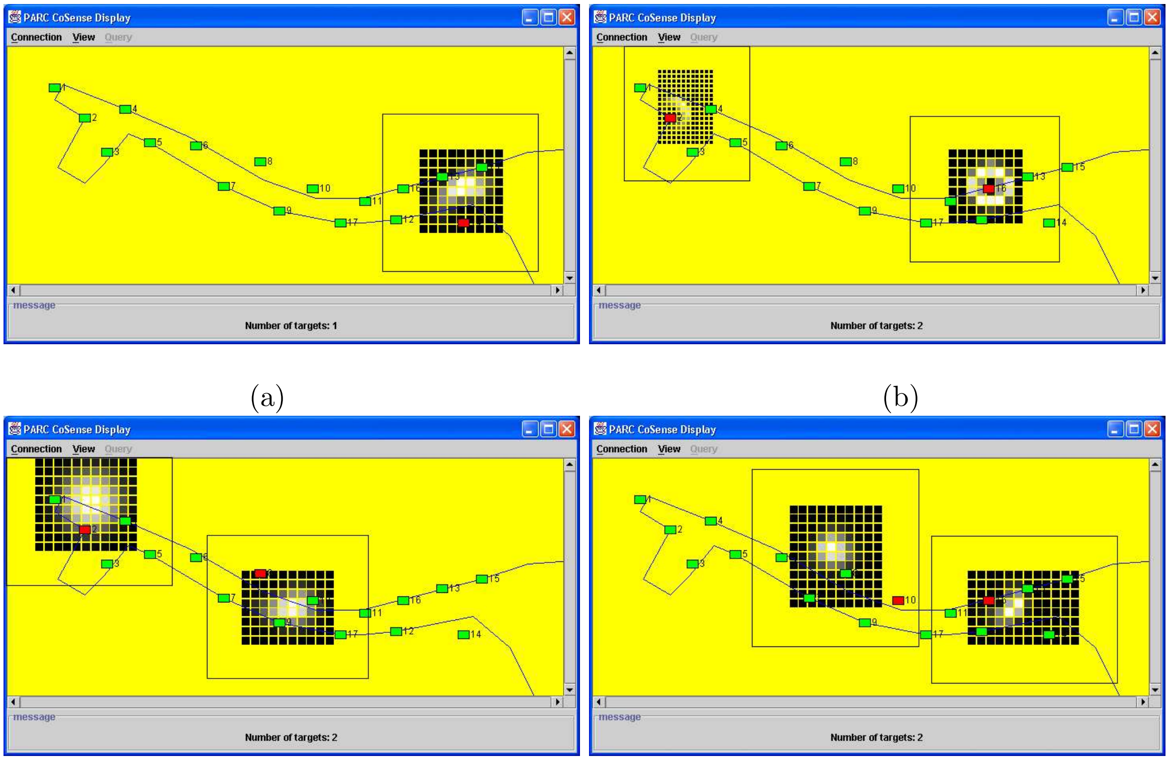
Figure 7. Sensor network layout. The solid line plots the service road. The sensor nodes are marked with small squares. Labels with a "D" are DOA sensors. The dashed lines picture the connectivity between nodes.

– Direction-of-arrival (DOA) sensors. They are arrays of 4 microphones attached to a single node, and use beamforming techniques (Chen et al., 2001) to determine the bearing to the target.

The nodes in our experiment included 6 DOA sensors and 11 energy sensors. This diversity in sensing modality helps to balance the systematic biases of individual sensors to obtain accurate target location. The nodes are placed along a service road outside the PARC building, as plotted in Fig. 7. Neighboring nodes talk to each other via 802.11b-based wireless links.

The geographically-based group management is built on top of the directed diffusion (USC/ISI, 2002) network protocol. To avoid unnecessary flooding of network packets, we use the GEAR (Geographic and Energy Aware Routing) protocol, which is an implementation of geo-casting (Ko and Vaidya, 1999) in directed diffusion. The protocol allows data to be sent to a specified geographic region, and limits message propagation outside of the destination region by optimizing the routes to gradually approach the region. Only the destination region is flooded with data. To cope with the constraint in GEAR that the geographic regions has to be specified as rectangles, we use the rectangular bounding boxes. In our experiment, the leader-election time-out t_{comm} is set to 1 second, and the detect range $R = 45$ feet.

The tracking and group management algorithms run on the node in real time. Each node runs a process similar to that described in Sec. 2.4 to decide which sensing mode to use. Two non-



(a) (b) (c) (d)

Figure 8. Snapshots of tracking. The rectangular boxes correspond to the collaborative regions. Nodes in red are leaders.

interfering targets are tracked. One is a military truck, and the other is a speaker playing a recorded sound of an amphibious assault vehicle (AAV). The ground truth of target locations was measured using differential GPS, which reports an average accuracy of 6 — 10 feet. To measure the tracking performance, we have computed the displacement between the location estimates produced by the tracker and the GPS-measured ground truth. The standard deviation averaged over a complete run is about 19 feet. Given that GPS measurement error and tracking error are independent, the tracking accuracy of our system is actually better than the reported 19 feet.

Fig. 8 shows a few snapshots of the tracking result. The belief states are pictured using greyscales. A bright cell indicates that a target is very likely to be on the cell location, and a dark cell suggests otherwise. In Fig. 8a, the first target is detected as it enters the sensor field from the parking lot. The second target has not appeared yet. The rectangular box enclosing the belief state represents the suppression region. The nodes inside the region (nodes 12 — 16) form a collaborative group, led by node 14. The rest of the network is doing standalone detection. Fig. 8b tracks the first target as it moves along the road (south bound) to the alcove end. Its collaborative region contains nodes 1 – 5. The second target has just been detected. Nodes 11 – 14, 16, and 17 form a collaborative group.

Fig. 8c and d tracks two targets simultaneously. The respective collaborative regions are plotted. The sensors are organized into independently coordinated groups, which enables the co-existence of multiple tracks and the simultaneous tracking of multiple targets. Tracking of two targets which are occasionally interfering has also been tested in our experiment. The tracking system can successfully recover from mutual interference via track merging and splitting. Overall, by restricting message broadcasting to be within the collaborative region (rather than flooding the whole network), the network traffic is reduced by 40% in this experiment. The traffic reduction will be more prominent for larger sensor networks.

5. Discussion

The paper focuses on the group management method for track initiation and maintenance in target tracking applications. While we have experimentally validated the basic structure of the group management algorithm, through extensive simulations and field experiments on sensor nodes, a number of important theoretical and experimental characterizations remain as immediate future research tasks. For example, we have not characterized how the performance of the algorithm, measured as the frequency of spurious track initiation or track loss, varies as target speed increases or parameters such as t_{comm} changes. The current implementation is also limited in its capability to maintain information about multiple targets once they closely approach each other. It might, however, be viewed as a key stepping-stone towards future systems with these capabilities. Modules to perform data association, classification, and source separation may be added, and different sensor selection approaches may be chosen, but the concept of organizing sensors in local collaborative groups to control information propagation is essential to each of these modules, and hence fundamental to the scalability of the entire system.

Collaboration between sensors is especially important in cases where individual sensors are of limited capability, for example, on the Berkeley motes. Individually motes can only perform simple tasks, and only through collaboration can more sophisticated sensing tasks be accomplished.

A similar group management method is described in (Fang et al., 2003), where a group is defined as those nodes that satisfy a membership predicate conveyed by message passing among neighbors. Using a target counting problem as an example, sensors in a network attempts to determine the number of distinct targets (i.e., sources of signals) in an area using strictly local measurements. The collaborative groups correspond to local regions dominated by energy peaks induced by the target sources, and the groups are formed based on local communication and simple detection amplitude comparison. The interesting feature of this method is that the geographic groups are only implicitly defined, i.e., no geographic regions need to be specified, and a geographic group is self-organized as a result of the underlying signal field the network is sampling. An interesting future research direction is to generalize the geo-specified and physics-specified group management methods to the more generic attribute-based group management protocols.

References

Chen, J., R. Hudson, and K. Yao: 2001, 'Joint Maximum-Likelihood Source Localization and Unknown Sensor Location Estimation for Near-Field Wideband Signals'. *Proceedings of the SPIE* **4474**.

Chu, M., H. Haussecker, and F. Zhao: 2002, 'Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks'. *International Journal of High-Performance Computing Applications* **16**(3).

Fang, Q., F. Zhao, and L. Guibas: 2003, 'Couting targets: building and managing aggregates in wireless sensor networks'. In: *Proc. 4th ACM Int'l Symp on Mobile Ad Hoc Networking and Computing (Mobihoc'03)*. Annapolis, MD.

Kinsler, L. E., A. R. Frey, A. B. Coppens, and J. V. Sanders: 1999, *Fundamentals of Acoustics*. New York, NY: John Wiley and Sons, Inc.

Ko, Y.-B. and N. H. Vaidya: 1999, 'Geocasting in mobile ad hoc networks: Location-based multicast algorithms'. In: *Proc. IEEE Workshop on Mobile Computer Systems and Applications*. New Orleans.

Li, D., K. Wong, Y. H. Hu, and A. Sayeed: 2002, 'Detection, classification, and tracking of targets'. *IEEE Signal Processing Magazine* pp. 17–29.

Liu, J., J. E. Reich, and F. Zhao: 2003, 'Collaborative in-network processing for target tracking'. *EURASIP, Journal on Applied Signal Processing* **2003**(4), 378–391.

Poor, V.: 1994, *An Introduction to Signal Detection and Estimation,* second edition. New York, NY: Springer-Verlag.

Stark, H. and J. W. Woods: 1994, *Probability, Random Processes, and Estimation Theory for Engineers,* second edition. Upper Saddle River, NJ: Prentice Hall.

USC/ISI: 2002. SCADDS Diffusion 3.1.2, developped by USC Information Science Institute, available on http://www.isi.edu/scadds/software.

Yu, Y., R. Govindan, and D. Estrin: 2001, 'Geographical and energy aware routing: a recursive data desemination protocol for wireless sensor networks'. UCLA Computer Science Department Technical Report UCLA/CSD-TR-01-0023.

Zhao, F., J. Shin, and J. Reich: 2002, 'Information-driven dynamic sensor collaboration'. *IEEE Signal Processing Magazine* **19**(2), 61–72.