

Fusion Driven Dynamic Space-Time Clustering for Sensor Networks

Shashi Phoha, Bharat B. Madan and John Koch
Information Science and technology Division/ARL
Pennsylvania State University, University Park, Pa. 16802
Email: sxp26@psu.edu, bbm2@psu.edu, jdk11@psu.edu

Abstract

Sensors require physical interaction with the sensed phenomena and are subject to a number of noise factors. Moreover, sensor data is highly correlated across a subset of sensors in the vicinity of a stimulus. To get reliable performance from individually less reliable sensors, time-critical collaborative inference in the vicinity of a stimulus is necessary to circumvent limitations of sensing, communications, power, and equipment faults. Dynamic space-time clustering (DSTC) is the ability of a sensor network to support such collaborative inferencing in the presence of physical stimuli. In this paper we present the DSTC algorithm for tracking events and targets by deploying a sensor field in the surveillance region. The computationally efficient DSTC algorithm leverages its performance by facilitating collaboration between sensors by way of sensor data fusion. To conserve network bandwidth required for sensor data fusion, we use a probabilistic finite state-machine model based on the symbolic dynamics theory to extract useful information from time series data that represents the raw sensor data. Such a model is capable of extracting maximum useful information in the form a probabilistic finite state machine. This paper describes protocols required to carry out DSTC and to adaptively reconfigure the network, in-situ, to capture the statistical characteristics of emerging change in the information dynamics. Building on this framework, we present an urban area application that requires adaptive sensor networks to dynamically cluster sensing, processing

Report Documentation Page			Form Approved OMB No. 0704-0188		
Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.					
1. REPORT DATE 2008		2. REPORT TYPE N/A		3. DATES COVERED -	
4. TITLE AND SUBTITLE Fusion Driven Dynamic Space-Time Clustering for Sensor Networks				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHOR(S)				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Pennsylvania State University, University Park, PA 16802				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)	
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release, distribution unlimited					
13. SUPPLEMENTARY NOTES The original document contains color images.					
14. ABSTRACT					
15. SUBJECT TERMS					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT UU	18. NUMBER OF PAGES 20	19a. NAME OF RESPONSIBLE PERSON
a. REPORT unclassified	b. ABSTRACT unclassified	c. THIS PAGE unclassified			

and communications resources in space-time neighborhoods of emergent *hotspots* for progressively fine grained sampling and prediction, and collaborate with other dynamic clusters for event tracking.

1 Introduction

A sensor network operates on an infrastructure of sensing, computation, and communication, through which it perceives the evolution of physical dynamic processes in its environment. Sensors require physical interaction with the sensed phenomena and are subject to a number of noise factors. Sensor data gathered by a group of sensors in the vicinity of a phenomenon therefore exhibits high mutual correlation. Since individual sensors tend to have limited capability in terms of quality of data gathered by individual sensors, higher situation understanding may be obtained by resorting to time-critical collaboration between networked sensors. Information fusion is an effective way to support such collaboration [1, 2]. Sensor fusion over a group of sensors in the vicinity of a stimulus can effectively circumvent limitations of sensing, communications and power limitation of individual sensors. Aggregation of data, information or both from multiple sensors has been shown to be effective in building better intelligent systems [3] as shown by Wong et al. [4] for improving the performance of 3-D face recognition using multiple sensors. Sensor fusion need not be limited to stationary sensors. Qi et al. [5] extend the idea of sensor data fusion to sensor networks utilizing mobile agents that can migrate to different sensors lying in the vicinity of a phenomenon being sensed to collect the sensed data. To support sensor data fusion, a network of static or mobile sensors has to address the following additional problems:

1. The sensors in the vicinity of a stimulus generate more data than the communication network can transport to a fusion center.
2. Communication paths, in addition to capacity limitations, are often not reliable due to limited transmitter power, queue overflows and interference .
3. Relative significance of data generated at a particular location is highly variable depending on the current tactical situation.

To deal with these problems, we extend the basic concept of multi-layer sensor fusion [2] by making the sensor network adaptive to changes in the sensed

phenomenon. Consider the problem of tracking a moving target by a sensor network. As the target moves in space and time, at different space-time epochs, a sufficient number of sensors need to be positioned in the vicinity of the path being traversed by the target to provide data/information to be fused. We refer to such dynamic changes caused in the sensor network topology and the associated data fusion operations as *dynamic space time clustering* (DSTC). The concept of DSTC in a large sensor network offers several benefits, such as, reduced network bandwidth requirements, power management, distributed processing and data fusion. Additionally, by dynamically optimizing the network structure, DSTC can improve network utilization and life time. In the context of the target tracking scenario, besides optimally selecting a cluster head, network optimization may entail optimizing cluster size and controlling the internal state of a sensor node. For example, if there are too many sensor nodes in a cluster, to reduce intra cluster traffic the cluster head may force some of these nodes to leave the cluster or instruct some nodes to hibernate, instruct some nodes to act only as routers, etc. Conversely, too few cluster nodes in a cluster may not deliver acceptable quality of fusion, requiring cluster head to request some nearby nodes to join the cluster. Much of the past research on sensor networks has either focussed solely on the networking issues or solely the fusion issues. In real scenarios (such as the one outlined above), these problems have to be dealt with together. This paper deals with developing a more holistic approach to sensor networking by integrating information, networking and fusion issues in an integrated framework. For ensuring brevity in communicating our ideas, we deal with a specific sensor network scenario that involves tracking a target moving through a field of randomly placed sensors. A subset of sensors close to the current position of the target form a cluster of resource constrained sensors. The topological attributes of a cluster keep changing dynamically to ensure best quality of fusion. We start by identifying a subset of sensors that appear to provide correlated information. Such an operation would require this subset of sensors to exchange large volumes of sensor data, thus overloading a resource limited sensor network. To minimize network overloading, our approach proposes to compress the raw sensor data by extracting semantic information via a probabilistic finite state model (PFSM) of the sensor data. This avoids having to send large volumes of raw sensor data, thus conserving network bandwidth [6]. The challenge then is to work with the PFSM model of the sensor data for identifying objects of interests, fusing information provided by multiple sensors, establishing relationships between

the objects of interest, etc. Computational sophistication required to generate and operate on the PFSM models provide us with a strong motive to split the problem at hand into two separate but interacting logical domains, namely, the Information Space (IS) and the Network Control Space (NCS), as shown in Fig. 1. The IS is responsible for dealing with all issues related to sensor data representation and associated information processing. In contrast, the NCS will be responsible for all networking issues, such as, protocols, network topology, information transportation, etc. In large sensor

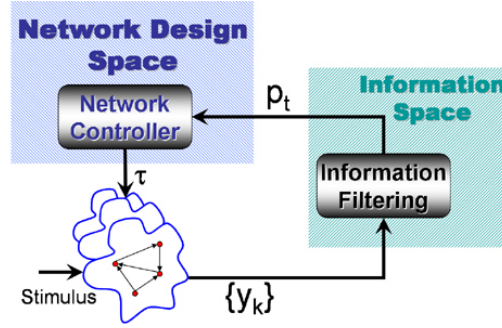


Figure 1: Solution Concept

fields used for persistent surveillance of urban areas for potential terrorist activities or undersea mine-hunting for ensuring safety of commercial shipping traffic, the sensor nodes are sparsely placed. Moreover, to conserve battery power, sensors are usually placed in the sleep mode until a particular stimulus in the operational environment alerts them to potential events of interest. Characterization of dynamic behavior of environmental stimuli under operational constraints requires tactful capture of both the coarse and the fine grained system dynamics. Such dynamics is crucial for ensuring persistent collaboration between distributed sensors. Human oversight and endurance for long duration sensor collaboration is neither possible nor practical in an amorphous and unpredictable networking environment. A large resource constrained sensor network must, therefore, dynamically switch from coarse to fine grained topologies in the vicinity of a stimulus in order to support progressively segmented analysis to localize emerging hotspots and events of interest.

This paper utilizes an analytical model of the abstract Information Space generated by asynchronous data streams emanating from the sensors [7]. The model enables us to compute the information fusion requirements in the

vicinity of a stimulus in terms of the cluster characteristics that will support the collaborative monitoring of the emerging hotspot. This paper formulates the DSTC protocol for reconfiguring the network topology as a function of changes in the information dynamics due to stimuli for the dynamical system served by the network. Thus, it also provides flexible configurations to accommodate uncontrollable fluctuations of the exogenous variables that generate stimuli to the sensors. The remainder of the paper is organized into 4 additional sections. In Sec. 2, we provide a brief description of the PFSM data modeling approach. Sec. 3, describes the DSTC technique and associated protocols used to derive a fusion driven cluster. In Sec. 4, we compare the performance of the DSTC technique with static clustering (i.e., when the sensor nodes are not mobile) and centralized clustering. Sec. 5 provides the concluding remarks and the future work being carried out.

2 PFSM Sensor Data Model

Sensors networks are typically constructed with low bandwidth wireless links. While individual sensor are resource limited, networking of sensors allows resource limited sensors to form a powerful collaborative system. The collaboration takes the form of sensor data fusion, which in turn requires all nodes in a cluster to exchange sensed data with each other, leading to large network bandwidth consumption. To minimize network bandwidth consumption, it is therefore imperative to avoid communicating redundant data. In this section, we describe a technique that models the raw sensor data as PFSM [7]. The basic concept used in deriving the PFSM model of a time-series data set is based upon the fundamental principles of finite state automata, pattern recognition, and information theory. It relies on the following two simplifying assumptions:

- The behavior of the time-series data is quasi-stationary at the fast time scale of process dynamics;
- Observable non-stationary behavior of the dynamical system can be captured with the help of parametric or nonparametric changes that evolve at a slow-time scale.

The main steps involved in arriving at such a model as given in [6] are:

- Symbolic dynamic filtering (SDF) to quantize the data block into a set of symbols $\{\sigma_1, ..\sigma_k\} \in \mathcal{S}$
- Probability distribution of the occurrence of different symbols. Simple histogram based methods have been shown to be quite effective in estimating these distribution.
- Derivation of the underlying hidden Markov model (HMM) using the D-Markov machine construction. The resulting D-Markov machine is the final PFSM model of a data block. Different data blocks of large time-series give a set slowly evolving set of such PFSM.

Complete algorithm used to obtain the PFSM model is beyond the scope of this paper and the readers can refer to the details in [6]. Such a PFSM model leads to a significant removal of redundant and predictable data present in a sensor data stream, thus compressing the sensor data. The ensuing data compression results in reducing the demand for network bandwidth. The overall problem of sensor data compression followed by sensor data fusion and network adaptation for efficient event tracking using the non-stationary statistics of the information dynamics to drive in-situ changes to the network space is depicted in (Fig. 1).

3 Fusion Driven Dynamic Clustering

To enhance quality of data fusion and resilience, a distributed sensor network needs to be adaptively reconfigured, where the network topology is dynamically updated based on the spatial-temporal information derived from the ensemble of sensor data. Clearly, it is not practical for all sensor nodes to exchange large volumes of raw sensor data with each other. To reduce the network traffic, we need to:

1. Choose a subset of sensor nodes that are capable of providing high quality sensor data. Such a subset of sensors is likely to be a cluster of nodes in the vicinity of the target or the phenomenon being sensed.
2. Extract only the useful information from the raw sensor data while discarding predictable information, thereby compressing the sensor data.

There are numerous data compression techniques available in the literature. In our previous works, we have suggested the discrete event dynamic system representation [7, 8] of a sensor data time series. Such a representation results in modeling sensor data as a PFSM. Furthermore, a fusion process is generally structured as a multi-layer process [2] with each higher layer delivering a higher level of abstraction. It has been proposed in [8] that multi-layered fusion can also be implemented by organizing sensors into a hierarchy of networking structures. This hierarchy starts with individual sensor nodes, then clusters of sensors and finally the inter-network of clusters that spans the entire sensor network. The PFSM modeling approach has the additional advantage that when applied iteratively for each fusion layer, it leads to a hybrid multilayered interacting probabilistic automata (IPA) [9] for each fusion layer. Continuously varying dynamics capture the physical processes at the lowest level of abstraction while discrete event models integrate sensing, computation and communication events in a formal language representation [9]. A formal language measure was also developed for measuring operational deviations from specified behavioral representations [10]. The PFSM and IPA data models lead to an appreciable reduction in network traffic. To further minimize the network traffic, we propose to create an off-line library of causal patterns consisting of PFSMs and IPAs. This library of patterns is initially created during the learning phase and is continuously updated during the operational phase by appending any new observed patterns. After converting the observed raw sensor data block into its PFSM model, we compare the generated pattern with the members of the pattern library using the language distance measure proposed in [9] and choose the pattern giving minimum distance d_{min} from a reference pattern. If the condition c_{min} defined by the inequality $d_{min} < d_t$ is true, then the sensor is selected as a potential candidate for being a member of cluster. We refer to this phase as the *pre-clustering* phase during which potential cluster members are identified. This is followed by the actual *cluster formation* phase, wherein these members select a cluster head that acts as proxy for the entire cluster of nodes. We have developed two heuristic algorithms for forming a cluster. The first algorithm follows *greedy clustering* strategy and has the advantage of being fast and requires less computing and communication. In contrast, the second algorithm is referred to as *patient clustering* that carries out a more detailed analysis of the distance measures to decide on a cluster structure and the cluster head. These two algorithms are described in the two subsections below.

3.1 Greedy Clustering Algorithm

The greedy clustering algorithm works by letting the first sensor node n_i that meets the condition c_{min} , as defined earlier in the previous paragraph, announces itself as the cluster head, along with the pattern p_j that it has observed and the lifetime of this cluster. During the life time of this cluster, the cluster head periodically broadcasts this information. Subsequently, if any other node n_k , $k \neq i$ sees the same data pattern p_j , it informs the cluster head of its intention to join the cluster. When the cluster head acknowledges this request from n_k , then n_k also becomes a member of this cluster. Prior to the expiry of the lifetime of this cluster, the cluster head has the option of renewing its designation as a cluster head. If the cluster head does not broadcast its renewal message, then at the end of cluster life time, the cluster is automatically disbanded. This algorithm may be summarized as below:

```

start:
forall sensor nodes n[k], k=1,...,N do
    collect m samples;
    compute PFSM pattern p[k];
    clusterhead = 0;
    for j = 1,...,|pattern_lib|
        if(distance(p[k], plib[j]) < threshold)
            clusterhead = j;
        endif
        if (clusterhead != 0)
            break;
    endfor;
endforall;
if (clusterhead == 0)
    goto start;
else
    broadcast (clusterhead, lifetime)
endif

```

A1: Greedy Clustering Algorithm

The working of this clustering algorithm is demonstrated via the target tracking example in Sec. 3.3. The greedy clustering algorithm has the advantage of being simple and efficient as far as cluster formation is concerned. However, it suffers from a serious drawback in that a greedily formed cluster head may be based on poor quality sensed information. It is quite plausible that over a small window of time, another sensor may have been able to observe the phenomenon more effectively. These limitations are addressed by the clustering algorithm described in the next section.

3.2 Patient Clustering Algorithm

In contrast to the greedy algorithm, the patient algorithm does not opt to choose the first observer of a phenomenon as the cluster head. Instead, each node periodically broadcasts a small number of data patterns it has observed (provided these data patterns meet the clustering condition c_{min}) along with their distance measure as suggested in [4] in the context of facial pattern recognition. The resulting fusion process ensures that the cluster members consult amongst each other to arrive at a consensus based election of the cluster head. The consensus criterion used to elect the cluster head is based on the distance measure associated with the PFSM models of the phenomenon being observed by the cluster members as summarized in the algorithm below:

```

start:
forall sensor nodes n[k], k=1,...,N do
    initialize <p[k]=NULL>
    collect m samples;
    compute acceptable PFSM patterns <p[k], d[k]>;
    clusterhead = 0;
    msgQ = NULL;
    if(p[k] != NULL) broadcast <p[k], d[k]> endif;
    forall received broadcasts
        append(d[k], msgQ);
    end forall;
    min = argmin(d[k]);
    clusterhead = min;
endforall;
if (clusterhead == 0)
    goto start;
else
    broadcast (clusterhead, lifetime)
endif

```

A2: Patient Clustering Algorithm

It is important that a cluster head once agreed upon by the cluster members fulfills its responsibility as a cluster head only for a finite time¹. After the expiry of the cluster head life time, a new cluster head should be elected. This ensures that the processing load associated with a cluster head is shared between a number of sensors. Otherwise, a single sensor acting as cluster head

¹this finite time is a design parameter that depends on the target dynamics and the battery power constraints

for a prolonged interval of time may end up exhausting its power source to the detriment of the entire cluster. On the flip side, time sharing cluster head duties introduces overheads by way of protocol overhead entailed in electing a new cluster head periodically.

3.3 Event Tracking using DSTC

Dynamic Space-time Clustering (DSTC) is a distributed processing algorithm based on the closest point of approach (CPA) [11] of the target to each of the sensors in the network [12–15]. DSTC is based on the concepts of a space-time neighborhood, a dynamic window, and an event. A space-time neighborhood centered $N(\bar{x}_0, t_0)$ around the space-time point $(\bar{x}_0, t_0)$ at which the event is assumed to have occurred is given by a set of space-time points,

$$N(\bar{x}_0, t_0) = \{(\bar{x}, t) \mid |\bar{x} - \bar{x}_0| \leq \Delta x, |t - t_0| \leq \Delta t\}$$

The quantities Δx and Δt define the size of the neighborhood in space and time, respectively. The space-time window contains all the data that was sensed within a distance Δx around $\bar{x}_0$ within the time interval Δt around time t_0 . This allows us to define a dynamic window around a moving point, $\bar{g}(t)$, as

$$w(t) = \{(x, t) \mid |\bar{x}_0 - \bar{g}(t_0)| \leq \Delta x, |t - t_0| \leq \Delta t\}$$

Ideally, if $\bar{g}(t)$ was the known trajectory followed by the event, we would analyze time-series data or its PFSM model provided by a subset of sensors in the window, $N_c = w(t_e)$ to determine information about the event at time t_e . In reality though, the event trajectory is not known. It is, in fact, what we want the sensor network to estimate. We therefore look at the closest point of approach (CPA) events that occur within a single space-time neighborhood. A CPA event e_{ij} is defined as being associated with the event i (i.e., the PFSM model of the i^{th} sensor's time-series data has had a closest match with a stored PFSM pattern) at the CPA time t_j . The space-time coordinates of the event are $(\bar{x}_i(t_j), t_j)$ where $\bar{x}_i(t_j)$ is the trajectory of event i . A subset of sensor $\mathcal{S}_i$ nodes that are capable of observing the event i effectively within a given space-time window form a cluster. The subset $\mathcal{S}_i$ is computed dynamically, based on the PFSM distance criterion described in [7]. The cluster head is selected using either the greedy or the patient clustering algorithm as described in previous subsection. In either case, the cluster head is responsible for computing the location of the event. In the

case of the greedy algorithm, the cluster head is responsible for estimating the spatial and temporal coordinates of an event as explained below. Each sensor within a cluster estimates the coordinates of the event with a certain degree of confidence. This level of confidence is given by the PFSM distance measure [7] between the PFSM patterns stored in a sensor's pattern library and the observed pattern computed by each sensor within a cluster. The occurrence of the CPA at each sensor is established based on when this distance measure attains minimum value [15]. Assume that a cluster of sensor node $\{n_1, n_2, \dots, n_k\}$ has been formed. Without loss of generality, let us assume that n_1 is the cluster head. The associated minimum PFSM distance measure set is given by $\{d_1, d_2, \dots, d_k\}$. This implies that the event coordinates reported by the sensor n_i , $i = 1, \dots, k$ lies within an uncertainty region given by a circle with radius d_i as shown in Fig. 2. Consequently, the coordinates of the event based on the fusion of estimates reported by all the cluster members will be given by the intersection of these k uncertainty circles. In the literature, such intersection problems have been shown to belong to the *Maximal Covering Location Problem* class and this class of problems has been shown to be NP-complete [16, 17]. It is therefore not feasible to optimally determine the location of an event or target, based on circular uncertainty region associated with each of the cluster's sensors. This is more so for sensor nodes that are characterized by having limited computing and communication resources. Keeping these limitations in mind, the cluster head uses a very simple heuristic for fusing together the estimates reported by various cluster members. Our main design goal for designing a

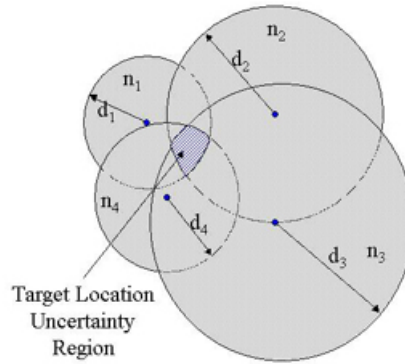


Figure 2: Location Estimate Fusion

heuristic algorithm has been the simplicity of the algorithm. Using extensive Matlab simulations, we decided to use linear weighted average heuristic for fusing individual estimates at the cluster head. Let s_i denote the cartesian coordinates of the location of the i^{th} sensor and as before, let d_i denote the distance measure of the PFSM pattern from a standard library pattern at this node. The spatial and temporal coordinates s_e and t_e of the the observed event are then computed as the weighted average of the individual estimates. The weights are taken to be the inverse of the distance measure that acts as a performance measure of an estimate computed by a particular sensor.

$$s_e = \frac{d_1^{-1}s_1 + d_2^{-1}s_2 + \dots + d_k^{-1}s_k}{d_1^{-1} + d_2^{-1} + \dots + d_k^{-1}} \quad (1)$$

$$t_e = \frac{d_1^{-1}t_1 + d_2^{-1}t_2 + \dots + d_k^{-1}t_k}{d_1^{-1} + d_2^{-1} + \dots + d_k^{-1}} \quad (2)$$

here, t_i represents the time of occurrence of the CPA event at the i^{th} sensor. After forming a cluster and establishment of the cluster head (using protocols A1 or A2 defined earlier in Secs. 3.1 and 3.2, respectively) each member of the cluster sends its estimates s_i 's and t_i 's to the cluster head that computes s_e and t_e . Fig. 3 shows the generation of an event track from the estimated position and velocity of the event as it moves through a sensor field. Our

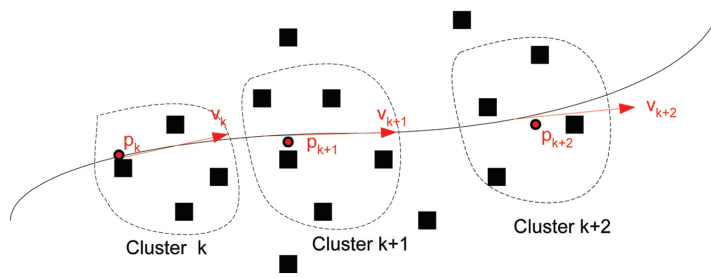


Figure 3: DSTC for Event Tracking

experimental results show that connectionless and best effort User Datagram Protocol (UDP) communication from each cluster member to the cluster head has acceptable reliability. Compared to connection oriented and reliable TCP connection, UDP has two limitations, 1) it is less reliable, and 2) it does not offer in-sequence delivery. However, UDP incurs substantially less protocol overhead as compared to the TCP, thus saving precious network bandwidth.

For the problem at hand, both these drawbacks are not very serious. We are able to alleviate the in-sequence delivery problem by packing both s_i and t_i into a single UDP datagram payload at the i^{th} sensor node. Moreover, fusion of individual estimates as given by (1) and (2) has inherent redundancy. Consequently, lack of reliability of UDP communication does not cause a significant drop in performance while computing the fused estimates (i.e., for reasonably large cluster, failure of a small fraction of sensors to successfully send their estimates to the cluster head does not cause a serious drop in performance). As explained earlier in the clustering protocols, a cluster has a finite and predefined life time, unless this lifetime is extended by the cluster head. Prior to the cluster being disbanded, the cluster head periodically broadcasts estimated spatial and temporal coordinates s_e and t_e , respectively for use by the next cluster. These broadcasts are terminated either after some fixed time or after the next cluster head informs the previous cluster head to abort its broadcast. These broadcasts ensure that the sensor network is able to track a target for a long time without long interruptions.

3.4 Network Reconfiguration

To respond to constantly changing sensed environment and the network state, the network structure needs to be constantly reconfigured to adapt to such changes for ensuring acceptable quality of sensor data fusion. To affect adaptation and control, we use a formal control model in which the NCS is primarily treated as the plant to be controlled and the IS as the controller or the supervisor. There are instances when the NCS also acts as a controller for itself. Different protocol layers in a network protocol stack have their own control plane. Examples of such autonomous control situations are, the Transmission Control Protocol (TCP) layer controlling its congestion window and retransmission timer, a battery powered wireless node switching off its routing (forwarding) functionality to conserve power and bandwidth, etc. While such autonomous control is important, fortunately there is a rich body of research material dealing with the autonomous control issues that can be applied to our situation. Consequently, our focus will be on exercising external control on a sensor network to enhance the quality of fusion, using one of the available control algorithms. In the proposed sensor network architecture, since the IS is primarily responsible for the sensor data fusion operation, we seek to arrive at a control structure in which IS makes control decisions. These control decisions are used to reconfigure the net-

work structure consisting of network topology, link rates, MAC layer backoff timer, radio transmission power, node mobility, prioritization of the traffic, etc. Such reconfiguration needs to be done in a manner that preserves the statistical characteristics (predictability) of the ensemble of original sensor data at each level of fusion (Fig. 4) as suggested in [8]. Since our primary focus is on sensor networks, we do not intend to go deeply into the control algorithm for controlling large networks. Instead, our discussion will be limited to describing the network controller architecture. We treat the control algorithm as a black box that generates control messages, such as, *transmit power on/off*, *change link data rate*, *move sensor*($\Delta x, \Delta y$), *adjust back-off timer by Δt* , etc. It should be designed to allow changing network parameters reactively in response to the sensed environment, the present network state and the current quality of fusion. The network controller should reconfigure a sensor network's topology and operation in a manner that the fusion quality is enhanced. In the proposed control architecture shown in Fig. 5, the controller logically resides in the IS of the CH of a cluster, while current and future cluster members act as plant(s) to be controlled. Sensor data quality and consequent fusion results computed by the IS are used as inputs by the control algorithm. A sample mapping of the data data fusion results to appropriate control messages is shown below:

Sensor Data/Fusion Quality	Control Message
Node 4, SNR=-6dbm	Node 4: Increase battery power 2dbm
Node 3, SNR=-20dbm	Node 3: Power off
Node 7	Node 7: Join cluster ID = 23
...	...

Since IS is responsible for all sensor data & information related functions, as mentioned previously, the CH's IS is also responsible for making information driven control decisions. A control decision is sent to the CH's NS via the InfoNet to be delivered to the target node(s). The sensor network delivers control messages to the NS of the target node(s) identified in the control message and each node implements the control decision. Sensor node executing a control function sends its response back to the CH.

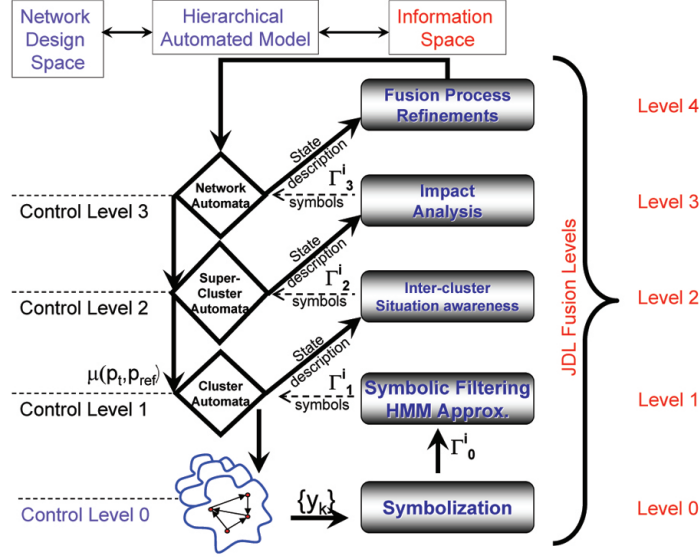


Figure 4: Hierarchical Network Control

4 Comparative Analysis and Results

To validate the effectiveness of the proposed DSTC techniques, we consider the problem of tracking a target passing through a network of sensors. We evaluate the relative performance of three different clustering techniques using the Matlab simulation codes for these three techniques,

CC Tracking: Centralized Tracking is a form of Static Cluster Tracking where there is only one Static Cluster Head to which all the sensor nodes send their CPA's, and track estimation is performed by signal strength weighted linear curve fitting of the reported CPA positions, constantly updated as new CPA's arrive.

DSC Tracking: Static Cluster Tracking is very similar to DSTC, with the exception that the cluster heads are pre-defined and clusters do not disband

DSTC Tracking: DSTC is relatively more complex as compared to the previous two techniques. The DSTC technique can be informally summarized in the following steps:

1. Sensor nodes wait for sensory input for an approaching target.
2. When a sensor node detects a rising input signal that then falls, and the peak signal is higher than a cutoff threshold, a Closest Point of

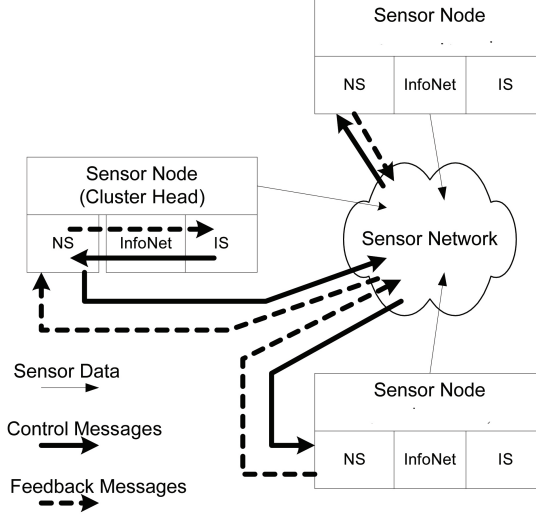


Figure 5: Information Driven Network Control

Approach (CPA) is declared at the time of the peak.

3. If the Node is not already part of a cluster, it sends out a cluster formation message to its neighbors, all of whom join the cluster if they are not already part of a previous cluster.
4. If the node is already part of a cluster, it sends its CPA, including signal strength, time of detection and the node's position to its Cluster Head
5. Once the Cluster Head has received enough CPA's (3 or more) it estimates the position and time of the actual detection as the signal weighted averages of the received CPA positions and times. The Velocity vector of the target is estimated as the signal weighted linear fit of the changes in CPA position over the changes in CPA times. This combined information forms a track point.
6. If the Cluster Head had not previously any such track points from other Cluster Heads, it originates a Track Message composed of its own track point. This Track message is then sent out to the sensor nodes along the target's estimated path.

7. If the Cluster Head had previously received any such track points from other Cluster Heads, they are compared with the local track point to find the incoming track most likely to be associated with the local point. If any matches are found, the best one is combined with the local track point to form a Track Message that is then sent out to the sensor nodes along the target's estimated path. If no matches are found, the Cluster Head proceeds as step 6.
8. Once the Track Message is sent, the Cluster Head disbands the cluster and informs all the cluster members.

The performance of these tracking algorithms is compared in Table 1 using the following metrics:

- Track Variance: the variance of the positions of the estimated track points versus the actual ground truth
- Track Coverage: the percentage of the target's path through the sensor field that was tracked
- Broadcast Data: number of bytes that need to be broadcast to disseminate CPA, Clustering and Tracking messages to other nodes
- Broadcast Energy: the amount of energy used to distribute the necessary CPA, Clustering and Tracking messages

These results validate the intuitive observation that CC method suffers from the drawback of poor energy efficiency. We have restricted our simulations to single hop communication. The poor energy efficiency can be attributed to the fact that nodes that are far away from the designated centralized cluster head node have to use much larger transmission power to communicate their sensed data to this central node. However, as is to be expected it provides very good location and velocity estimation performance since the cluster head receives sensed data from all other sensor nodes. In contrast to CC, the DSC method uses much less power since the average communication radius of a static cluster is much smaller than a single large cluster implied in CC. However, due to the static nature of the clusters, the tracking performance is rather poor. This is attributable to the fact that if the target follows a trajectory that is in close proximity to the location of static clusters, only then individual clusters provide better location and velocity estimates. However,

Comparison Metrics	DSTC (Patient)	DSTC (Greedy)	Static	Central
Track Variance (m^2)	1.743	0.431	1.048	0.446
Track Coverage (%)	88.617	90.868	90.863	75.188
Broadcast Data (byte)	1904	2384	2848	1520
Broadcast Energy (mJ)	3.828	5.696	4.897	21.656

Table 1: Tracking Algorithm Comparison Table

if the location of static clusters is far away from the target tracks then the sensed signal strength is reduced due to increased propagation losses. This translates to poor signal to noise ratio (SNR), which is the main performance determining factor. Finally the DSTC method is able to deliver advantages of both the CC and DSC methods since small clusters are dynamically formed around the current location of the target. This reduces transmission power and ensures higher SNR and hence better tracking performance.

5 Conclusions

This research presents issues to be dealt with in the design of energy-efficient and fusion driven reconfigurable sensor networks used for conducting surveillance in urban areas. Use of compressed sensing in which a sensor node compresses the raw sensed data using a PFSM sensor data model, leads to significant reduction in use of network resources. Using four different simulated sensor network topologies, *viz* CC, DSC, DTSC-Greedy and DSTC-Patient, we show that the working of the proposed sensing technique for tracking a target moving through a sensor field. These simulation studies show that DSTC-Patient has the best energy performance without any significant loss of tracking performance but at the expense of much more complex clustering protocol than others. We are also in the process modifying the four afore mentioned clustering to include fusion driven dynamic sensor network adaptation. We expect to report these in a another paper in the near future.

Acknowledgment

This work was supported under the eSensIF MURI Award No. W911NF-07-1-0376 sponsored and administered by the Army Research Office. Any opinions, findings, and conclusions or recommendations expressed in this

publication are those of the authors and do not necessarily reflect the views of the sponsor.

References

- [1] B.V. Dasarathy. *Decision Fusion*. IEEE Computer Society Press, 1994.
- [2] D.L. Hall and J. Llinas. *Handbook of Multisensor Data Fusion*. CRC Press, 2001.
- [3] R.C. Luo and M.G. Kay. Multisensor integration and fusion in intelligent systems. *IEEE Trans. Systems, Man and Cybernetics*, 19(5):901–931, Spet./Oct. 1989.
- [4] K.C. Wong, W.Y. Lin, Y.H. Hu, N. Boston, and X. Zhang. Optimal linear combination of facial regions for improving identification performance. *IEEE Trans. Systems, Man and Cybernetics (Part B)*, 37(5):1138–1148, October 2007.
- [5] S. S. Iyengar H. Qi and K. Chakrabarty. Multi-resolution data integration using mobile agents in distributed sensor networks. *IEEE Trans. Systems, Man and Cybernetics (Part C)*, 31(3):383–391, August 2001.
- [6] A. Ray. Symbolic dynamic analysis of complex systems for anomaly detection. *Signal Processing*, 84(7):1115–1130, 2004.
- [7] I. Chattopadhyay, A. Ray, and S. Phoha. Mathematical modeling of sensor network dynamics for control & stability. Technical Report W911NF-07-1-0376, Prepared for ONR, September 2008.
- [8] S. Phoha and A. Ray. Dynamic information fusion driven design of urban sensor networks. In *of the 2007 IEEE International Conference on Networking, Sensing and Control, London, UK*, pages 1–6, 2007.
- [9] A. Ray and S. Phoha. Signed real measure of regular languages for discrete-event automata. *Int. J. Control*, 76(18):1800–1808, 2003.
- [10] A. Ray, V. Phoha, and S. Phoha. *Quantitative measure for discrete event supervisory control*. Springer, 2005.

- [11] F.M. Dommermuth. The estimation of target motion parameters from cpa time measurements in a field of acoustic sensors. *J. Acous. Soc. Am.*, 83(4):1476–1480, 1988.
- [12] S. Phoha, N. Jacobson, D. Friedlander, and R. Brooks. Sensor network based localization and target tracking through hybridization in the operational domains of beamforming and dynamic space-time clustering. In *Proceedings of Globecom 2003*, pages 2952–2956, 2003.
- [13] D. Friedlander, C. Griffin, N. Jacobson, S. Phoha, and R. Brooks. Dynamic agent classification and tracking using an ad hoc mobile acoustic sensor network. *Eurasip J. on Applied Signal Processing*, 4:371–377, 2003.
- [14] S. Phoha, N. Jacobson, and D. Friedlander. Sensor network based localization and target tracking through hybridization in the operational domains of beamforming and dynamic space time clustering. In International Journal of Distributed Computing, editor, *Proceedings of the 2003 Global Communications Conference*, San Francisco, California, Dec 1-5 2003.
- [15] S. Phoha, E. Grele, C. Grffin, J. Koch, and B. Madan. Space-time coordinated distributed sensing algorithms for resource efficient narrowband target localization and tracking. *International Journal of Distributed Sensor Networks*, 1(1):81–99, 2005.
- [16] R. Schmalensee and J.-F. Thisse. Perceptual maps and the optimal location of new products: An integrative essay. *International J. of Research in Marketing*, 5:225–249, 1988.
- [17] Y. Crama, P. Hansen, and B. Jaumard. Complexity of product positioning nad ball intersection problems. *Mathematics and Operation Research*, 20(4):885–894, Nov. 1995.