Self-Organized Inference of Spatial Structure in Randomly Deployed Sensor Networks

Neena A. George, Ali A. Minai
ECECS Department, University of Cincinnati,
Cincinnati, OH
georgena@ececs.uc.edu, aminai@ececs.uc.edu

Simona Doboli
Department of Computer Science
Hofstra University
Hempstead, NY
simona.doboli@hofstra.edu

Randomly deployed wireless sensor networks are becoming increasingly viable for applications such as environmental monitoring, battlefield awareness, tracking and smart environments. Such networks can comprise anywhere from a few hundred to thousands of sensor nodes, and these sizes are likely to grow with advancing technology, making scalability a primary concern. Each node in these sensor networks is a small unit with limited resources and localized sensing and communication. Thus, all global tasks must be accomplished through self-organized distributed algorithms, which also leads to improved scalability, robustness and flexibility. In this paper, we examine the use of distributed algorithms to infer the spatial structure of an extended environment monitored by a self-organizing sensor network. Based on its sensing, the network segments the environment into regions with distinct characteristics, thereby inferring a cognitive map of the environment. This, in turn, can be used to answer global queries about the environment efficiently and accurately. The main challenge to the

network arises from the necessarily irregular spatial sampling and the need for totally distributed computation. We consider distributed machine learning techniques for segmentation and study the variation of segmentation quality with reconstruction at different node densities and in environments of varying complexity. The eventual goal of the work presented in this paper is to obtain intelligent networks capable of autonomous reconfiguration based on their observations. The inference of spatial structure in monitored environments is clearly an essential first step for such self-reconfigurability.

1 Introduction

Wireless sensor Networks (WSN's) are interconnected groups of spatially distributed sensor nodes used to gather and process large amounts of data. Each sensor node is a miniature electronic device, characterized by limited power, sensing, communication and computation capabilities [Pottie 2000].Advances in hardware and communication technologies have led to the deployment of these nodes in large numbers for applications as diverse as habitat monitoring, border surveillance, contaminant tracking, and health monitoring [Chintalapudi 2003][Nowak 2003], giving rise to concerns such as scalability of processing algorithms and robustness of the network [Estrin 1999]. Also, traditional methods of processing where all data is transmitted to a central base station and processed there using powerful centralized algorithms is non-robust and inefficient with high communication overhead and poor response time. This has led to the development of decentralized algorithms, and such sensor networks can be regarded as self-organizing complex systems in which localized interactions between neighboring nodes helps to achieve a global objective.

Most environments monitored by sensor networks have spatial structure (e.g., tracks, or regions with different terrain) that significantly affects the observations made by the network. If the network can infer this spatial structure from initial observations, it can predict events in the environment much more intelligently than a naïve, uniformly organized network. For example, network performance in tracking applications is much better if the network can infer the layout of tracks (if any) or distinguish between hazardous and navigable regions. Since every node surveys only a small part of the environment, inference of the environment structure by the network requires intelligent collaboration between the nodes.

Segmentation can be defined as the identification of regions or clusters which have similar properties. This similarity could be similarity of attributes such as moisture, texture, composition, etc. or similarity in events occurring in these regions, e.g., regions with high/low traffic or regions with higher/lower event density. The goal of segmentation in a self-organizing network is to use the information obtained from local sampling to identify these regions, thereby inferring the spatial semantic structure of the environment.

Segmentation of the sensed environment can be seen as a powerful generic task that can serve several purposes including, but not limited to, the following:

1. Providing a cognitive map to interpret observed events.

2. Allowing better prediction of events (e.g., the direction of a moving entity).

3. Improving the quality of sensing by customizing sensing parameters to each region.

4. Improving network efficiency by allowing better sensor scheduling.

5. Facilitating task allocation among nodes, leading to better resource utilization.

Segmentation is widely studied in the fields of computer vision and image processing, and some of the methodologies used there have been applied in the context of sensor networks by considering sensor nodes as equivalent to pixels. However, significant differences exist between the two [Devaguptapu 2003]. In image processing, pixels are regularly spaced and each pixel has eight neighboring nodes. On the other hand, nodes in sensor networks are randomly deployed .The number of a node's neighbors is determined by how many nodes fall within its transmission and hence is variable in a random network. Also, due to random deployment, node density may not be completely uniform, leaving significant areas of the environment poorly monitored. This irregular sampling of the environment by the sensor nodes coupled with the lower density of deployment, makes it difficult to apply image processing techniques with equivalent accuracy in sensor networks.

In this paper we look at the problem of segmentation in environments of different complexities, propose to use machine learning techniques for the intelligent approximation of areas not covered by sensor nodes, and study the quality of segmentation under different node densities.

The rest of the paper is organized as follows. Section 2 looks at the related work and background of the algorithms used .In Section 3 we present the approaches developed in our research. Section 4 deals with the summary of results obtained, Section 5 with the analysis of results, and Section 6 concludes with future directions of work.

2 Related Work

Most of the work done in the area of spatial analysis of data in sensor networks has been on edge/boundary detection and has relied on methods used in image processing. In [Devaguptapu 2003] and [Chintalapudi 2003], the authors tackle the problem of distributed edge detection by applying filter-based schemes. A linear classifier-based approach and a statistical thresholding scheme in addition to the Prewitt Filter is used in [Chintalapudi 2003], and the three different methods compared, of which the linear classifier approach is found to perform the best. Nowak and Mitra examine a hierarchical method for boundary detection in sensor networks [Nowak 2003]. Edge reconstruction using platelets is studied [Willet 2003].

While boundary detection is of importance for determining the extent of phenomena, a method is also needed to identify semantically significant regions in the environment. This would not only involve extracting information from the

locations at which sensors are present, but an estimation of the environment at locations not covered by sensors. Though some work has been done which deals with estimation of homogenous fields using interpolation methods [Nowak 2003], scenarios involving abrupt spatial changes are likelier to occur in a variety of applications and distributed, energy efficient methods are needed to address the problem of estimating these inhomogeneous fields [Nowak 2003].

As segments are comprised of regions with similar attributes, there is a natural analogy between finding segments and finding clusters. We use a method of region growing proposed by Panangadan and Sukhatme for region tracking [Panangadan 2005] .The basic idea behind this algorithm, as used in image processing, is that if two adjacent pixels are similar in intensity, they are merged into a single region.While they use it purely in the context of tracking a specific region and in a single attribute case, we investigate the use of the algorithm in finding all regions in environments of different complexities , with multi-attribute specifications. We examine the use of non-hierarchical, distributed methods to estimate inhomogeneous fields and use it in conjunction with the region growing algorithm to get a comprehensive global estimate of the segments. We also use two machine learning algorithms, namely the Inverse distance weighted (IDW) algorithm and the K-nearest neighbors (KNN) algorithm for data imputation, making intelligent guesses about un-sensed regions. We also evaluate the performance of these methods with different node densities.

For simulation purposes, we use cellular environments, but argue that these methods can be used in a continuous framework too.

3 System Design

Environments of different types and complexities are used to test the performance of the algorithms. Each environment is modeled as an $n \times n$ cellular region, which is divided into segments of different types. Regions or segments can be defined as connected subsets of the environment which have similar values for all or some attributes. Every cell location (x, y), is characterized by N_a attributes, $a_i(x, y)$, where $i=1, 2,....N_a$. There are N_s segments in the environment, drawn from a fixed number of *segment types*. Attribute a_j is distributed over segment S_k, in a range $[a_{jk}^{min}, a_{jk}^{max}]$, with mean m_{jk} and variance σ_{jk}. Each segment type has a characteristic mean and variance for each attribute, and the means for different segment types are well-separated on at least one attribute.

3.1 Environment Models

In this paper, we use the following specific types of environments:

1. ***Track Environment*:** The environment has two different piecewise linear tracks of different widths and is characterized by two attributes. The tracks are similar to each other on one attribute, but differ on both attributes from the background segment.

2. ***Patch Environments*:** Two different types of patch-like segments are distributed over the background, and are characterized by two attributes. The segments differ from each other on one attribute, and from the rest of the environment on both attribute types. Note that the patch segments imply a non-convex background segment by default.

3. ***Spiral Environments*:** These environments consist of two intertwined spiral segments. The segments differ from each other on both the attributes. This is a very complex segmentation problem, and is used primarily as a benchmark because its complexity can be controlled systematically by varying the width of each spiral. It has been widely used in the classification literature.

3.2 Segmentation Algorithm

A total of N_v sensor nodes are randomly distributed over the environment. Each node is equipped with some sensing, communication, and computation capabilities. It also has some limited memory associated with it. Each node has a unique randomly generated identifier and is capable of sensing all attributes of interest. Every node is assumed to sense the attributes within the cell in which it falls. Nodes also have a transmission radius, t_r, within which they can broadcast and receive messages. Nodes that fall within this radius around a specific node are called its *1-hop neighbors*, or just neighbors. Every sensor uses information from its neighbors to infer whether it is part of a larger region.

Segmentation is based on the detection of similarity in sensed attributes for neighboring nodes. Using H(R) to denote homogeneity for region R, the basic property defining a region, R_i, is [Petrou 1999]

$$H(R_i) = \text{TRUE},$$

i.e. every region is homogenous within itself. Also

$$H(R_i \cup R_j) = \text{FALSE}, \forall i \neq j,$$

where R_i and R_j are adjacent regions. Homogeneity can be found by comparing all attributes or a subset of attributes.

Each node, v_i , senses the attributes within its sensing range, and stores the data in a measurement vector μ_i. It also transmits to its neighbors a message packet with its identifier and measurement vector. Communication is assumed to be omni-directional and instantaneous over a 1-hop neighborhood. Every node checks the similarity of the values that it senses with those received from each of its neighbors. Similarity is said to obtain if the two nodes sense values which fall within a certain percentage of each other. If a match is found, the node marks this in a match table, and the process continues over several sensing cycles. Once the number of times the node finds a match with a neighbor (as a fraction of the number of trials) exceeds a pre-specified

threshold, it deems its neighbor as similar to itself and hence lying in the same region. The requirement of multiple matches even in static or slowly-varying environments ensures robust segmentation in the presence of measurement noise. The node then sets its identifier to the minimum identifier among all its similar neighbors. Every node continues to compare itself to its neighboring nodes until no further similarities are found over a sufficiently long interval. At this point, all the nodes in a region with similar readings have converged to the same identifier value.

Even at high densities, sensor nodes do not sample every part of the environment, and complete segmentation requires assignment of segment identifiers to un-sensed cells through inference. For this, we use two very similar algorithms, the KNN algorithm and IDW algorithm, both implemented by the nodes in a distributed manner. Each node has a cell map of a pre-specified size $m \times m$, which represents the node's subjective estimate of a limited region beyond its area of sensing. In the version of the KNN algorithm implemented, every node finds the k-nearest neighbor nodes for each cell within its cell map and decides the label or identifier of the cell based on the plurality of these. The algorithm is tested using different sizes for the cell map and by varying k. By increasing the size of the cell map, a more comprehensive approximation of the environment can be made at even lower densities.

A slightly modified version of the KNN algorithm is used in the inverse distance weighting method in which each of the k neighbors is weighted inversely with its distance to the cell in deciding the label. The weight, w_j of a node label is calculated as

$$w_j = d^{-(k+1)}$$

where d is the distance of the node to the unsensed cell and k is the number of neighbors. The label with the greatest cumulative label is assigned to the cell.

4 Results

The quality of segmentation is studied for each of the environment types. We define segmentation metrics based on the number of correctly classified cells found after the running the algorithm.

1. Track Environment: In this environment configuration, the number of track cells that are classified as background segment cells and the total number of correctly classified track segment cells (including type 1 and type 2), are found for 3 runs of the algorithm at a specified density .The average accuracy of classification is then calculated and the procedure repeated for different node densities .A comparative plot of the KNN and IDW algorithms at different node densities is plotted in Figure 3 for the track environments. Accuracy is defined as the percentage of track 1 and 2 cells classified correctly.

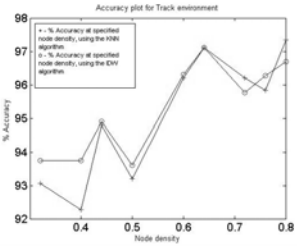

Figure. 3: Accuracy plot for Tracks

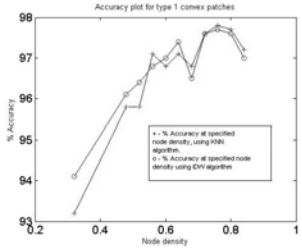

Figure. 4: Accuracy plot for patches

The figures below (Figures 5 – 7) depict an example of the reconstruction.

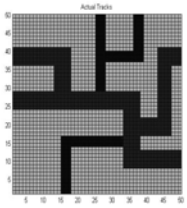	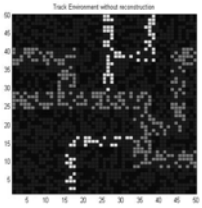	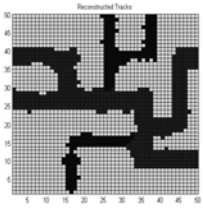
Figure. 5: Actual Track Environment	**Figure. 6**: Tracks before reconstruction	**Figure. 7.**:Tracks after reconstruction at 0.5 node density

2. Patch Environment: The number of incorrectly and correctly classified convex segments are found, averaged over 3 different runs and plotted for different node densities. Figure 4 shows the error plot for one of the patch segments. Similar results are obtained for the second type of patch segment. Accuracy is defined as the percentage of patch cells classified correctly.

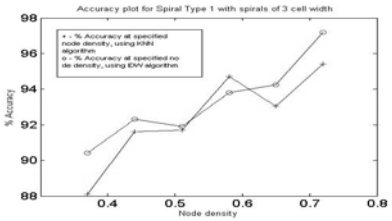

Figure. 8: Accuracy plot for type 1 spiral (3-cell width)

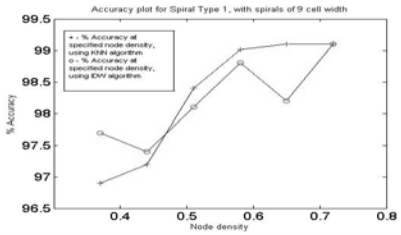

Figure. 9: Accuracy plot for type 2 spiral (3-cell width)

3. Spiral Environment type: Here the number of spiral cells that are classified correctly and misclassified are calculated, averaged over 3 runs and plotted at different node densities (Figures 8– 11).

For each environment (3-cell width, 6-cell width, 9 cell width), taken into consideration, the percentage accuracy of each spiral type is plotted, where accuracy is defined as the percentage of spiral cells classified correctly.

The plots for the environment type with spirals of widths 3 cells (Figure 8 and Figure 9) and 9 cells (Figure 10 and Figure 11) are shown .

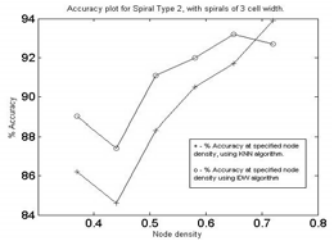

Figure. 10: Percentage spiral type 1 correctly classified (9 cell widths).

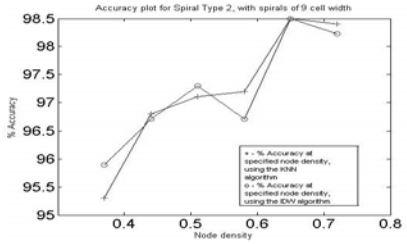

Figure. 11: Percentage spiral type 2 correctly classified (9 cell widths).

5 Analysis of Results

Both the KNN algorithm and Inverse Distance Weighted algorithm show comparable results, with high accuracy even at lower node densities. As expected, the percentage accuracy of classification improves at higher node densities for all environment types, with minor variations, which can be attributed to the variation in node-distribution over the segments. Some configurations might show better accuracy levels at lower density values which are very close, but significant increase in accuracy is seen at higher densities.

The IDW algorithm is found to perform much better than the KNN algorithm at low node densities (typically < 0.5), though at higher densities the two algorithms have very similar performance but with greater variation. This might be because at lower densities, nodes that sense differently are more likely to lie further from the segment edges than at higher densities. At higher densities, though the overall accuracy would increase, the performance of the two algorithms would undoubtedly depend on the node distribution due to random deployment.

Clearly, both the KNN and IDW algorithms can be used with sufficient accuracy for environment inference in unsensed regions, over a wide range of complexities, with the IDW algorithm showing better performance at lower densities of deployment.

6 Future work

Distributed classification of a monitored environment into comprehensive segments is only the first step in understanding the sensed environment. The next logical step

would be to enable the network to answer global queries about the environment .The network should also be able to correlate the events in the regions to the characteristics of the region. Work on these issues is currently in progress.

Bibliography

[1] Chintalapudi, K.K, & Govindan, R., 2003, Localized Edge Detection in Sensor Fields, IEEE Workshop on Sensor Networks Protocols and Applications

[2] Devaguptapu, D., & Krishnamachari, B., April 2003, Applications of localized image processing techniques in wireless sensor networks, SPIE's 17th Annual International Symposium on Aerospace/Defense Sensing, Simulation, and Controls, (Aerosense '03), Orlando, Florida.

[3] Estrin, D., Govindan, R., Heidemann, J., & Kumar, S.,1999, Next Century Challenges: Scalable Coordination in Sensor Networks, Proc. MOBICOM, Seattle, 263-270.

[4] Nowak, R. & Mitra, U., 2003, Boundary Estimation in sensor networks: Theory and Methods, Proc. IPSN'03, 80-95.

[5] Nowak, R., Mitra, U., & Willet, R., 2004, Estimating Inhomogeneous Fields Using Wireless Sensor Networks, IEEE Journal on Selected Areas in Communications, Vol. 22 , No. 6 (Aug.), pp. 999-1006.

[6] Panangadan, A., & Sukhatme, G.S., 2005, Data Segmentation for Region Detection in a Sensor Network, CRES Technical Report, 05-005.

[7] Petrou, M. & Bosdogianni, P., 1999, Image Processing, the Fundamentals, John Wiley and Sons.

[8] Pottie, G., Kaiser, W., Clare, L., & Marcy, H., 2000, "Wireless Integrated Network Sensors". Communications of the ACM**,** 43: 51 - 58.

[9] Willet, R., & Nowak, R., 2003, Platelets:A multiscale approach to recovering edges and surfaces in photon-limited imaging, IEEE Trans. Med. Imaging, 22:332–350.