TASC: Topology Adaptive Spatial Clustering for Sensor Networks

Reino Virrankoski †, Dimitrios Lymberopoulos and Andreas Savvides
Embedded Networks and Applications Lab (ENALAB)
Electrical Engineering Department
Yale University, New Haven, CT 06520

Abstract— **The ability to extract topological regularity out of large randomly deployed sensor networks holds the promise to maximally leverage correlation for data aggregation and also to assist with other tasks such as sensor localization. This paper focuses on extracting such regular structures through the development of a distributed clustering scheme. The Topology Adaptive Spatial Clustering (TASC) algorithm is a distributed algorithm that partitions the network into a set of locally isotropic, non-overlapping clusters without prior knowledge of the number of clusters, cluster size and node coordinates. This is achieved by deriving a set of weights that encode distance, connectivity and density information within the locality of each node. The derived weights form the terrain for holding a coordinated leader election in which each node selects the node closer to the center of mass of its neighborhood to become its leader. The clustering algorithm also employs a dynamic density reachability criterion that enables the grouping of nodes according to their neighborhood density properties. Our simulation results show that the proposed algorithm can produce uniform sized isotropic clusters and it is tolerant to up to 30% measurement noise. In the cases of non-uniform distribution the network is partitioned according to the local density attributes. Using these behaviors we then illustrate how such clustering scheme can benefit ad-hoc localization.**

I. Introduction

The anticipation of large-scale sensor networks and experience from preliminary deployments has demonstrated the need for meaningful decomposition of large distributed sensor networks into a set of smaller subnetworks. Such decomposition should be conducted in a manner that facilitates sensor node coordination and enhances the feasibility network management and in-network processing and aggregation of sensor data. In this paper, we explore this issue of network decomposition through the development of a specialized distributed clustering scheme. The scheme we investigate tries to extract regularity from irregular network topologies by allowing the nodes to organize themselves into groups of locally isotropic (or regular) clusters without requiring the knowledge of node locations. Given the close coupling of sensors to the physical world we advocate that such classification of sensor nodes according to their spatial attributes would be beneficial from multiple aspects.

†Visiting assistant researcher from the Control Engineering Laboratory, Helsinki University of Technology, Finland

Besides the intuitive benefit of improving the ease of network management, the spatial grouping of nodes with respect to regions of close proximity and similar deployment density promotes efficient data aggregation and compression of sensor data. As pointed out by [11] spatial irregularity in sensor sampling can exacerbate the load and cost imbalance between different parts of the network. This is mainly because many of the existing distributed signal processing and compression algorithms assume spatially regular data samples. This also entails that the spatial grouping of nodes can help reduce the propagation of redundant data inside the network. This argument is further reinforced by the recent results presented in [16] that suggests the existence of optimal cluster sizes under certain conditions based on the correlation among measurements from multiple sensors and the routing costs of forwarding the data to the sinks.

In addition to improving sampling, the understanding of network structure and spatial clustering can benefit other important network tasks such as the formation of hierarchies and node localization. In node localization the organization of nodes into smaller groups reduces the required computation and helps limit redundant computations, something that is favorable for small resource constrained sensor nodes. Moreover, in the case where high precision locations are required, the organization of nodes in isotropic clusters can also help with limiting error propagation inside the network. In section VI we describe a study case for weight driven localization and outline a multistart scheme for limiting error propagation in distributed localization systems.

Despite the fact that clustering has been previously studied both theoretically and in the context of ad-hoc networks [1], [3], [4], [8], [10], its consideration in the context of sensor networks gives rise to a new problem setup. From this perspective, the uniqueness of this work lies in its ability to cluster large networks in to a set of small non-overlapping clusters in a distributed manner without any prior knowledge on the number of clusters and cluster sizes. The algorithm operates at each node and encodes local information on inter-node distances and routing information into a set of weights that form the terrain to elect a set of leaders and hence organize nodes into clusters. The size and density properties of each cluster are controlled with the introduction of a dynamic density reachability metric, a variation of the density reachability concept borrowed from data-mining [20], [26]. The advantages of this approach arise from the ability to capture the topology information within the locality of a node a single quantity - the weight of a node that can be used as a starting point for leader election and hierarchy creation.

The proposed distributed algorithm does not require node locations but it assumes that nodes are aware of their 2-hop neighborhood and that distances between nodes are known. We consider both assumptions reasonable. The former is a standard assumption for many neighborhood discovery algorithms whereas the latter is becoming a common feature of many sensor networks. Accurate distance measurements in the sensor network domain have been demonstrated using ultrasound in the system described in [28], the MIT Crickets [27] and in the Medusa MK-2 node [22] developed us. In the radio domain, ultra-wide-band ranging systems such as the one offered by Ubisense [29] have already demonstrated very accurate distance measurements in very small packages that will be suitable for sensor networks. From our simulation results we found that our algorithm is resilient to high levels of noisy measurements. This implies that it may be possible to use this techniques with radio signal strength distance measurements in environments We also note that the spatial clustering of the network before node localization is actually an advantage for ad-hoc localization. As we will describe in section VI, ad-hoc localization schemes such as [17], [22], [24] may actually benefit from the properties of our algorithm to eliminate computation redundancies and geometric error propagation.

The contributions of this paper include the development and characterization of a *Topology Adaptive Spatial Clustering Scheme* (TASC), through the development of weights and dynamic density reachability. In addition to this this work illustrates the application of TASC to node localization and enumerates a set of additional sensor network applications that can be assisted by TASC.

The paper is organized as follows. In the next section we highlight the related work. Section III describes the clustering problem requirements. Section IV provides the details on the creation of weights and describes the clustering algorithm. Section V explains the details of TASC and explains the election algorithm, Section VI illustrates how TASC can be applied in ad-hoc localization. Section VII discusses some additional attributes and section VIII states our conclusions and plans for future work.

II. Related Work

The idea of clustering is not new, and it has been studied in many different domains including computer theory, neural networks, astronomy and many others. In general it is considered a hard problem in terms of optimality, but many application specific algorithms exist. A brief overview of this work is provided here.

A. K-means clustering

A mathematical framework that has similarities with the network clustering problem is the k-means clustering [3]. In classical k-means, the number of cluster center points is set in advance. Initially, center points are placed to data either according to some a priori knowledge or randomly. Then, for each data point, the algorithm computes the cluster center that is closest to the data point based on specified nearness criteria. After that, each center point is moved to the location that is the average of the locations of those points that are closest to that center. One famous modification of the classical k-means problem is the fuzzy k-means clustering, where the same point can simultaneously belong to several clusters and have membership degree that is specific to each cluster.

The idea of figuring out the clusters based on some incomplete a priori knowledge is similar to ad hoc clustering. Furthermore, some k-means clustering modifications that are interesting in ad hoc clustering point of view are recently presented in [4], [7], [8]. What makes the setting we investigate different from others is the fact the amount of prior knowledge is smaller than typical k-means applications. Nodes are only able to measure distances to their one hop neighbors, positions are unknown and the network architecture does not offer the centralized knowledge needed for basic k-means algorithm applications.

B. Ad Hoc Network Clustering

From the ad-hoc networking perspective, Gupta et al [9] present a way of finding optimal sensor cover to a query that targets in a specific region in the network area. In their application, each query has a time window and a geographical window associated with it. Each piece of data generated in the sensor network has a timestamp and a location stamp. The algorithm uses the redundancy of the network to select a small subset of sensors that defines a connected sensor cover. The main novelty is savings achieved in power consumption. Both centralized and discrete versions of the algorithm are presented [9]. Chen and Liestman [10] present a zonal algorithm to find weakly connected dominating sets. The algorithm consists of three phases. First, an input graph representing the ad hoc network is partitioned into regions of approximately size x. Then, the distributed algorithm for weakly connected sets is run in each region and finally some additional region border vertexes are added. Basagni in [1] uses weight-based criteria in a distributed ad hoc clustering algorithm. Each node has a weight, and nodes are grouped based on their own weight and the weight of the nodes in their one hop neighborhood. Beyond its distributed nature, the proposed algorithm can deal with changing topology. When nodes are moving, one can set node weights to be inversely proportional to their velocity and then set the nodes having lowest velocity to be cluster centers.

In our approach, local variations in network topology and density are taken into account so that we are able to locally split anisotropic network into a set of locally more isotropic clusters. The clustering parameters are able to adapt to different network types without the requirement of manually changing them. The final number of clusters depends on network topology.

III. Clustering Objectives

When considering a clustering scheme for ad-hoc sensor networks one first needs to define the meaning of a *good* cluster. In the trivial case of uniform deployment and known node coordinates, the network can be partioned using a grid construct. In [16] an optimal cluster size is computed with respect to the energy cost of forwarding data from a set of sources to the sink. When considering irregular ad-hoc sensor deployments, the question of determining the optimal cluster size while considering network densities still remains an open question. Some theoretical work by Jain and Dubes [13] suggest techniques to estimate cluster validity in terms of compactness and isolation, but the general problem of defining a baseline on which results should be compared remains open, even though one suggested solution is the use of studies based on Monte Carlo simulations.

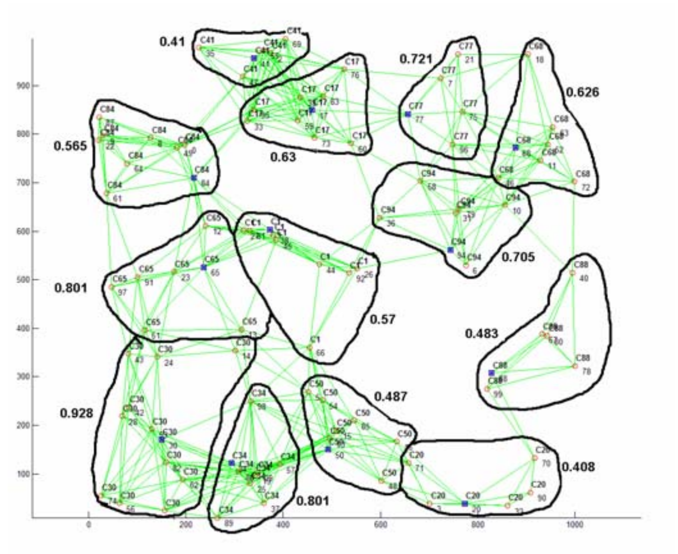


Fig. 1. An example of the clustering outcome

Our clustering approach is motivated by the requirements of the sensor network domain. More specifically, a clustering algorithm should partition the network so that the nodes inside each cluster have high correlation in sensor measurements and are evenly spaced in order to maximize gains and reduce errors due to ill geometric positioning as in the case of node localization.

In contrast to uniform deployments, one cannot dictate a fixed number of clusters or use a grid construction since that would diminish the exploitation of correlation properties. Instead of requiring a fixed number of cluster sizes and in order to avoid the creation of single node clusters, our algorithm requires only the minimum number of nodes in a cluster. *The problem we pursue aims to partition networks with density non-uniformities, into a set of smaller locally isotropic clusters by grouping nodes with similar density attributes*. We consider this to be the analogous of grid partitioning in deterministic deployments that will enable the clustering of nodes in non-uniform deployments, such as deployments along winding corridors and other non-isotropic patterns.

IV. Distributed Leader Election

A. The Leader Election Algorithm

The distributed leader election algorithm builds on two main components, *node weights* and *density reachability* and takes place in two phases nomination and voting followed by a merging phase. During the first phase, each node considers the weights of nodes in its 2-hop neighborhood, nominates the node with the maximum

```
Inputs: 2-hop neighborhood, inter-node
distance measurements, minimum cluster size, MinPoints
Output: Leader node
------------------------------------------------------
$weight$ = ComputeWeight();
BroadcastToNeighborhood($weight$);
If all $weights$ received:
  Select heaviest density reachable node
  as $nominee$;
  BroadcastToNeighborhood($nominee$);
EndIf
If all nominations have been received:
  Select the closest $nominee$ as $leader$;
  BroadcastToNeighborhood($leaderID$, $nodeID$);
EndIf
If this node is $leader$:
  Wait until election timeout;
  BroadcastToNeighborhood($clustermembers$);
EndIf
If cluster size is received:
  If $clustersize <$ minimum cluster size:
    select the closest neighbor for which
    $clustersize \geq$ minimum cluster size
    and join its cluster;
  EndIf
  BroadcastToNeighborhood($leaderID$, $clustersize$)
EndIf
```

Fig. 2. Leader Election Algorithm

weight as an election candidate and notifies the nodes in its neighborhood of this nomination. In the second phase, each node elects the closest candidate as its leader. Nodes that end up in clusters that are smaller than a pre-specified minimum cluster size are dismantled and their node members join bigger existing clusters. The pseudocode of the leader election algorithm running on each node is given in Figure 2. A detailed description of weights and density reachability is provided in the next subsections. We recommend the reader to revisit the leader election algorithm after reviewing those sections.

B. Weight Computation: Discovering Local Network Structure

The computation of node weights tries to achieve the reverse effect of greedy forwarding in geographic routing [5], [6]. In greedy forwarding, a node found on the path to a packet destination, forwards the packet to its neighboring node with location closest to the location of the destination. Instead of trying to forward traffic the neighboring node that is closest to destination, TASC all-pairs-shortest path routing based on distance measurements to extract information about the network topology. More specifically, a node weight is a measurement of two key quantities 1) the frequency a node is found on the shortest path between pairs of nodes and 2) the distance contribution of that node with respect to the total length of the path. In the basic case of uniform deployment in a circular field, the node found closer to the center of mass of the network will have the highest weight.

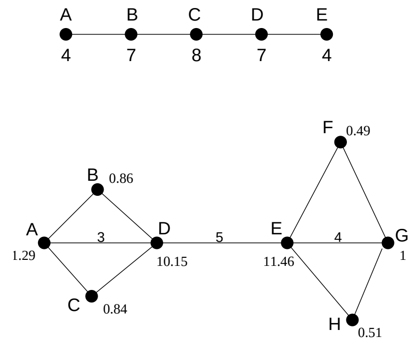


Fig. 3. Weights example

Consider the network in Figure 3a. If we define the weights to be the number of times a node is found on a the shortest path then we can compute a weight for each node. Node A for instance can be found on the paths AB, AC, AD and AE hence it will have a weight of 4. Node C is found on eight different paths hence it receives a weight of 8. To construct a proof of this behavior we use the principle of optimality [2]:

If S is the shortest Euclidean path between two nodes, it includes all shortest paths between all pairs of nodes that are located in path S.

Definition 1: Each node in the sensor network gets weight +1 each time the shortest Euclidean path between any pair of nodes in the network crosses that node or ends at it. Paths are assumed undirected in weight computation.

Theorem 1: Let S be the shortest Euclidean path between two nodes, and let $2n+1$ be the total number of nodes in path S. When computing all shortest paths between each pair of nodes in path S and assigning weights to each node in S as presented in Definition 1, the node that is from equal hop distance from both endpoints of path S, gets the biggest weight.

Proof: Observe path S having total number of $2n+1$ nodes, and let c be the node located from equal hop distances from both ends of the path S. Since the total number of nodes in path S is $2n+1$, there are n nodes on both sides of node c. Based on basic routing theory, S includes shortest paths between all pairs of nodes located in S. Thus, the weight of node c is equal to the total number of shortest paths crossing node c and ending at node c:

$$W_c = n \cdot n + 2n = n^2 + 2n \tag{1}$$

Pick then a node g from the path S so that there are $k < n$ nodes from the other side of that node. In that case there are $n + (n - k)$ nodes on the opposite side. The weight of node g is:

$$W_g = k(n+(n-k))+k+n+(n-k) = 2nk+2n-k^2 \tag{2}$$

When comparing the weights (1) and (2) we get

$$\begin{aligned} W_g < W_c \Leftrightarrow 2nk + 2n - k^2 < n^2 + 2n \Leftrightarrow \\ n^2 - 2nk + k^2 > 0 \Leftrightarrow (n-k)^2 > 0 \end{aligned} \tag{3}$$

That holds always when $0 < k < n$. ■

Corollary 1: If there are 2n nodes in the path S discussed in Theorem 1, two nodes in the middle get both equal biggest weight values.

Proof: The result follows from equations (1)-(3), when total number of $2n$ nodes are used. ■

Theorem 2: When weights in network graph are computed like presented in Definition 1, the node or nodes closest to the network center get the biggest weights.

Proof: The proof is a generalization of the discussion presented in equations (1)-(3). Observe M shortest paths that are crossing each other in one node, and mark $N = 2M$. In the symmetric case, the number of nodes in each path is $2n+1$ and all paths are crossing each other in the midmost node c. When all shortest paths between pairs of nodes located in paths M are taken into account in weight computation, the weight of the node c is

$$\begin{aligned} W_c = n \cdot (N-1) \cdot n + n \cdot (N-2) \cdot n + ... \\ +n \cdot n + N \cdot n = n^2 \sum_{i=1}^{N-1} i + Nn \end{aligned} \tag{4}$$

Observe next asymmetric case, where one of the paths has $n+k+1$ nodes, where $1 \leq k < n$, when the rest of paths have still $2n+1$ nodes, and paths are crossing each other in node g so that in path M_j there are n nodes on the one side and k nodes on the opposite side of node g. For other paths $M_{i,i \neq j}$, g is still the midmost node. In that case the weight of node g is:

$$\begin{aligned} W_g = n \cdot (N-2) \cdot n + kn + n \cdot (N-3) \cdot n + kn + ... \\ +n \cdot n + kn + kn + (N-1)n + k \\ = n^2 \sum_{i=1}^{N-2} i + (N-1)kn + (N-1)n + k \end{aligned} \tag{5}$$

When comparing weights W_c and W_g, we get

$$\begin{aligned} W_g < W_c \Leftrightarrow \\ n^2 \sum_{i=1}^{N-2} i + (N-1)kn + (N-1)n + k < n^2 \sum_{i=1}^{N-1} i + Nn \\ n^2 \sum_{i=1}^{N-2} i + (N-1)kn + (N-1)n + k < ... \\ < n^2 \sum_{i=1}^{N-2} i + n^2(N-1) + Nn \\ (N-1)kn + (N-1)n + k < (N-1)n^2 + Nn \\ (N-1)kn + k < (N-1)n^2 + n \end{aligned}$$

which is true under assumption $1 \leq k < n$. ■

This is enough to show that always the node that tends to be the midmost related to all shortest communication paths (in terms of hops) gets the biggest weight. The result of Corollary 1 generalizes this result so that if some of the paths have even number of nodes, there can be several nodes with equal biggest weights in the middle. An illustrative example of the weight behavior for a deterministic grid deployment and a non-deterministic deployment is shown in Figure 4.

1) Including Distances in Weight Computation: Although this method of computing weights would decide the central node, it does not give enough information in the cases where the paths are symmetric such as the paths in the example network shown in Figure 3b. To handle this problem, we augment the weight computation to incorporate distance information. Instead of incrementing the weight by one each time a node is used in a path, we increment the weight as a function of the distance a node contributes to the path. If a node k is found on the path from node i to node j in between nodes a and b, then the weight increment of node k is given by equation 6 where $l_{a,k}$ and $l_{k,b}$ are the lengths of the edges between nodes a and b and node k respectively and $l_{i,j}$ is the length of the whole path from node i to node j.

$$w_{ij} = \frac{l_{a,k} + l_{k,b}}{l_{ij}} \tag{6}$$

An example of this method of weight computation is shown in Figure 3b. According to equation 6 the weight contribution from path AG to node D is $(3+5)/12$. In the same figure we also note that the use of distances in weights helps to identify the node closest to the center of mass of the network. Node D and E have symmetric path configurations and the consideration of routing only would have resulted in equal weights for both nodes. The consideration of distances and shortest routes helps identify the node that is closest to the center of mass, node E in this case.

C. Density Reachability: Grouping Similar Densities

While the information from node weights can be used to identify local centers, we would still like to construct clusters by grouping nodes in regions with similar density attributes. To achieve this goal, in addition to considering weights we need to consider additional means of pulling areas with high node densities towards the center of a cluster. To be able to do so using only distance measurements, we must define a group specific to each node in which nodes have similar or higher densities to the node under consideration. This can be achieved with a modified version of density reachability traditionally applied in data clustering [20], [26] to cluster spatial data in the presence of obstacles.

When density reachability is used, each node can further limit the number of nodes that it can potentially nominate by considering only density reachable nodes as nomination candidates. The determination of density reachabilily is based on a dynamic distance metric called density range defined as follows:

Definition 2: The density range r_i of node i with respect to the minimum number of nodes $MinPoints$ is the smallest disk centered at i that covers $m-1$ other nodes in the vicinity of i.

Based on this we define a node to be density reachable as follows:

Definition 3: A node j is density reachable from i if there is a path from i to j where the length of every hop l satisfies the constraint: $l \leq r_i$.

Figure 5 shows an example network when $m = 3$. Nodes j, k and the black nodes are density reachable from node i since the hop length to reach each of these nodes from i is smaller than r_i. Note that for the purposes of our clustering algorithm, density reachability can only expand within the 2-hop neighborhood of each node.

V. Evaluation of Cluster Properties

To characterize the properties of the clustering algorithm, we run a set of simulations on a suite of 100 random scenarios. In each scenario, 100 nodes are deployed on a square deployment field of size 1000 by 1000. Each scenario is used five times for five different measurement ranges 200, 250, 300, 350 and 400. For most cases, the minimum number of nodes per cluster is set to 4. The simulation also assumes that the distance measurement range of the node is equal to the communication range. In practice, we expect that the communication range is greater than the measurement range, so this assumption does not violate the fundamental properties of our clustering algorithm. Our simulations are implemented with an in-house version of NeslSim [15], which is implemented in PARSEC. The main role of the NeslSim environment in our work is the enforcing of a distributed implementation of our clustering algorithm. The computation of shortest paths is done using the Floyd-Warshall algorithm running at each node. The measurement noise model is modeled as additive noise following a white gaussian distribution that the standard deviation of which is entered as a percentage of the maximum measurement range. Cluster characterization is based on one main metric, the estimation of the area of a cluster from a discrete set of points, the coordinates of each node. The

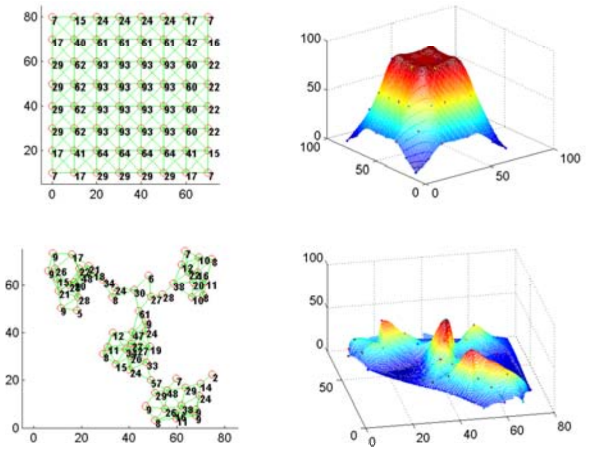


Fig. 4. 64 nodes having range 10 scattered in 80x80 area. In first case nodes are placed in regular grid and in second case node placement is random. Each node computes its weight based on shortest paths in its two hop environment. In the case of grid placement there is no changes in local topology and because of that the weight distribution is smooth increasing from borders to the middle. In random placement case maximum weights indicates centers of local topology structures.

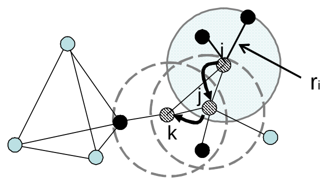


Fig. 5. Selecting density reachable nodes. Node i select its density reachable nodes when min nodes = 3

area of a cluster is approximated by the area of an ellipse that contains at least 99% of the nodes that form the cluster. The major axis of the ellipse is the straight line regression of the y coordinates on the x coordinates and the center of the ellipse is the the center of mass of the cluster (the mean of the x and y coordinates of the cluster members). The ratio of the major to minor axes is the ratio of the standard deviation of the node distances to the cluster center of mass, projected onto the major and minor axes. The area of this ellipse is used as an estimate of the cluster area. Although the area of the ellipse will always be larger than the actual cluster area, the relative ratios of cluster areas remain the same. An alternative method to estimate cluster areas is to compute the sum of the areas of the Delaunay triangles included in the convex hull of the cluster. We chose to use an ellipse fitting instead since this metric also allows us to estimate the *axial ratio* (roundness) of a cluster. Using the estimated cluster area, our evaluations also make use of the term *cluster density* in nodes per meter square.

1) Cluster Isotropy: The first experiment was to evaluate the cluster isotropy by computing the average node separation in each cluster, when the density reachability parameter MinPoints and the minimum cluster sizes are set to 4. This is the average distance from each node in the cluster to its closest neighbor. Figures 6a-c shows the distribution of average distances to each node within each cluster plotted against the density of their cluster for three different measurement ranges 200, 300, 400. Each dot in the figure, is the average separation per clusters for each of the clusters resulting from our scenario suite. From figure 6 we observe that the majority of the dots fall within 40 and 100. This result shows that the clustering algorithm creates clusters by packing nearby nodes together thus producing locally isotropic configuration. This property is also illustrated in Figures 9 and 10. Figure 9 shows the average axial ratios for all scenarios in our test suite. Figure 10 is a snapshot of the axial ratios for the resulting clusters in the example network in Figure 1.

2) Cluster Sizes and Density Reachability: The trends in cluster sizes are shown in Figures 6a-c and 7a-c. Intuitively, one would expect the average cluster size increase with increasing measurement range since the area of the 2-hop neighborhood increases. Instead, the average cluster size remains constant between 7 and 9 nodes in each of the tested 300 cases. This is enforced by the density reachability parameter, which prohibits nodes from electing a leader in from a region of the network that does not have similar properties than region the node is in.

The properties of density reachability are better illustrated in Figures 12a-c. The three cases show how cluster sizes change for three different values of $MinPoints$ and the minimum cluster size set to 4. The parameter $MinPoints$ sets the density range of a node to be the distance to the $(MinPoints-1)^{th}$ neighbor or the furthest neighbor if the $(MinPoints-1)^{th}$ neighbor is not within measurement range. The first case of $MinPoints=2$ limits the density reachable nodes to the distance of the closest neighbor. This is very small so the effect of density reachability is overshadowed by the merging process of the leader election algorithm. With

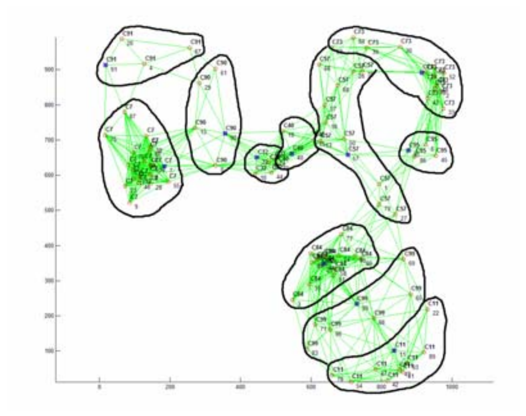


Fig. 8. Clustering outcome on an anisotropic network

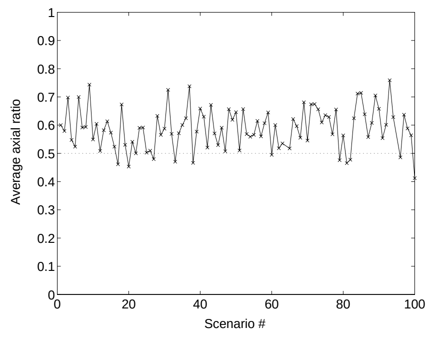


Fig. 9. Average axial ratios for all scenarios

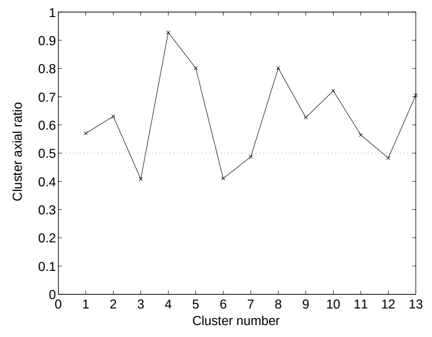


Fig. 10. Axial ratios for the clusters of the network in Figure 1

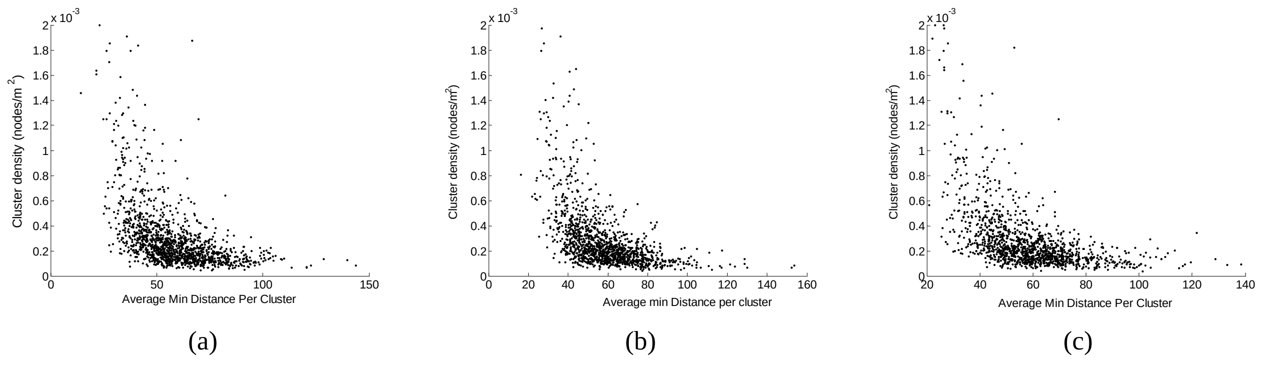


Fig. 6. Average distance to closest node in each cluster. Measurement range = a)200, b)300, c)400

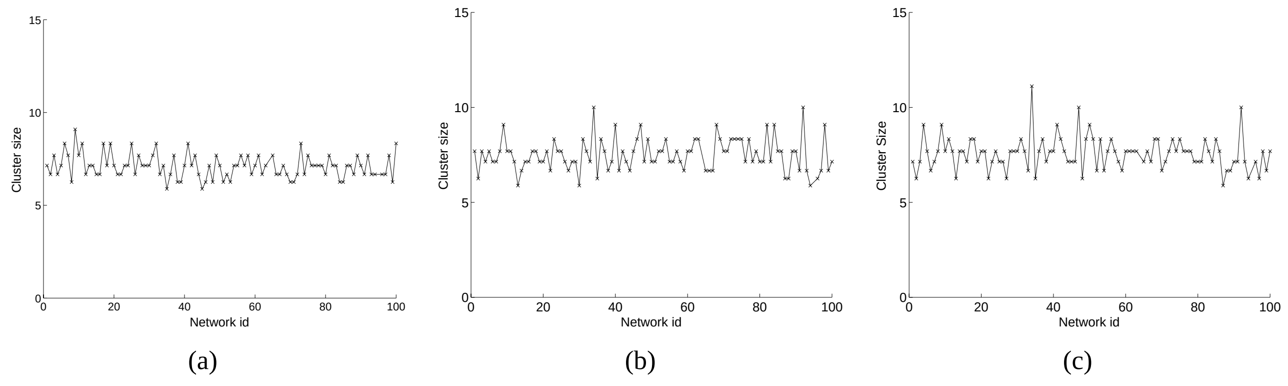


Fig. 7. Average Cluster Sizes. Measurement range = a)200, b)300, c)400

no control on the eventual cluster density properties, the cluster sizes increase with measurement range, and at high measurement ranges clustering results in a single cluster that has the size of the whole network. The case of $MinPoints = 4$ (Figure 12b) represents the most desirable cases where the majority of the cluster sizes stays within 4 and 10 nodes. The effects of density reachability begin to diminish again for high values of $MinPoints$ as shown in the c part of the figure. As the density range begins to approach the maximum measurement range of the node, the effect of density decays to the point where it cannot differentiate among density variations in the vicinity of the node. This behavior entails that to obtain reasonable cluster sizes, density reachability should be tuned with respect to the maximum measurement range and deployment area.

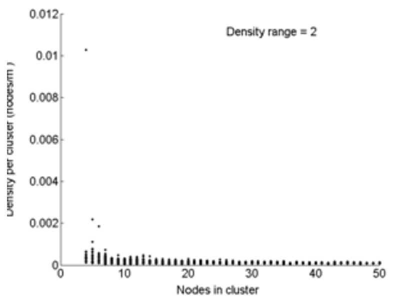
A. Behavior with anisotropic networks and noise

Figure 8 shows the clustering outcome on a highly anisotropic network that includes a winding path, holes and exhibits large density variations. This pictorial representation best describes the outcome of TASC for such networks, and demonstrates it resilience to density perturbations. We also note that since TASC operates using local information and weights, the clustering outcome will not be affected by the existence of wholes inside the network.

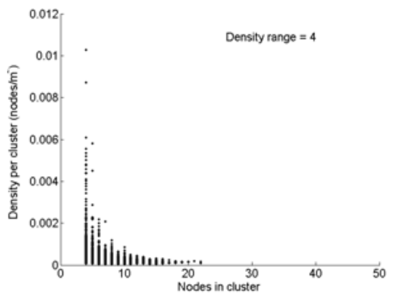
From our simulations we also found TASC to be resilient to measurement noise. With up to 30% measurement error we were able to obtain consistent cluster sizes. The effect of noise is shown in Figure 11. Between 30% and 60% measurement error, most of the clusters are still uniform, but some non-uniform clusters exists. When error level increases to or above 60%, most of the clusters become non-uniform. Even though our algorithm is able to produce clusters in the presence of 30% error level, the effect of density reachability weakens rapidly between 20% and 30% error levels and thus the algorithm is not able to take local density variations into account with the same accuracy as in lower noise levels. As a result of this, the trend shown in Figures 6 and 12 becomes weaker.

VI. Clusters and weights in node localization

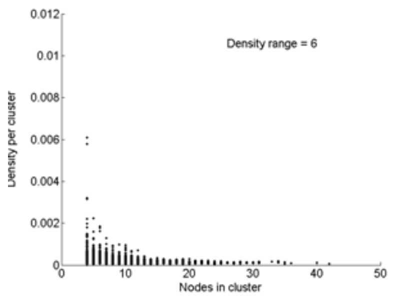
We now illustrate how the topology information extracted from weights and the existence of clusters and leaders can be used to drive computation in distributed node localization for three main reasons. First, by starting location computation at regions of higher density (denoted by the cluster heads) one can limit error propagation inside the network. Second, the existence of clusters keeps location computation tractable on resource



(a)



(b)



(c)

Fig. 12. Clustering behavior on 100 scenario suite at different density ranges set at the level of a)2, b)4, c)6 neighbors

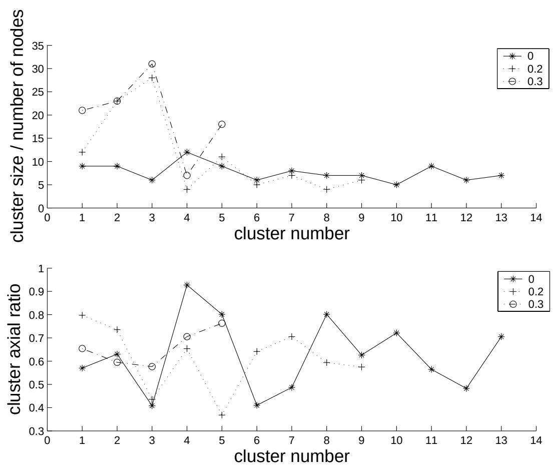


Fig. 11. Clustering outcome of the example network shown in Figure 1 without noise and with noise levels 20% and 30%

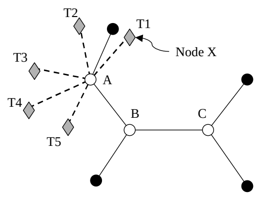


Fig. 13. Geometry error propagation

constrained sensor nodes and third, it reduces the redundancy in location computations.

The role of TASC on error propagation is motivated with the example network in Figure 13. The black nodes are beacon nodes and nodes A, B, C are unknown nodes. Positions $T1$-$T5$ represent different possible positions for a second beacon T attached to node A. Table I shows the error predicted by the Cramèr Rao bound for each of the five possible beacon positions under the assumption of white gaussian measurement error. The Cramèr Rao bound result is obtained from our previous work in the analysis of error inducing parameters in multihop localization described detail in [23]. From the table we can see that in this particular example, the error in the position estimate of node C can increase up to 87%. This is an artifact of bad geometric configurations of beacon node T located three hops away. This geometric dilution of precision effect (GDOP) [23], [25] can result in the propagation of a significant GDOP error component in multihop localization. To reduce the GDOP accrued error, one could initiate localization at the high-density regions of the network using the topology information encoded in the node weights. By starting localization in *locally isotropic* regions seeded with the elected leaders, localization can simultaneously start at multiple points inside the network, consuming the best geometric configurations first. Instead of providing the complete details of how node localization and TASC can be combined we illustrate this with the following two examples. A more detailed description of ad-hoc localization integrated with TASC will be presented in a subsequent paper.

TABLE I
BOUNDS ON LOCATION ERROR

Test Position	Node A	Node B	Node C
T1	1.48	2.17	2.49
T2	1.36	1.62	1.94
T3	1.34	1.48	1.4
T4	1.43	1.95	2.00
T5	1.5	2.24	2.63

A. Study case 1: Exploiting isotropy and meeting sensor node computational constraints

In addition to error accumulation to demonstrate the need for meeting the computational constrains, we draw from our experience in the design of an ad-hoc localization system based on the location aware Medusa MK-2 recently demonstrated at Sensys 2003. Our current efforts are focused on the design of a new distributed localization stack using a new generation sensor node the XYZ node designed by the authors to experiment with location aware systems [1].

The XYZ node, shown in Figure 14, is built around the OKI ML67Q500x series of ARM THUMB processor supporting a variable clock speed between 2 and 58MHz, and the IEEE 802.15.4 and Zigbee compliant radio from Chipcon. Our primary focus in the implementation of XYZ is the development of a versatile location stack described by Figure 15. The memory consumption of our first round implementation is shown in Table II. Although this implementation still requires substantial code optimizations we use the memory footprints of our current implementation status to provide some insight into the the memory limitations of sensor nodes. From our current implementation we found that the XYZ node has enough memory to form a local coordinate system for a group of 15 nodes. Our beaconless localization implementation consists of an enhancement of three main phases as originally outlined in [22]. In the first phase, the algorithm identifies a uniquely localizable configuration of nodes [2]. The second phase obtains a set of initial node position estimates. For this phase we have implemented the MDS algorithm as described in [24]. Finally, refinement is implemented with a beaconless version of collaborative multilateration we have previously presented in [22]. With our implementation with found that TASC shares several common features with the rest of the localization algorithm so its implementation does not pose a significant memory overhead. Two of the main challenges is to minimize error by exploiting local isotropy and to compute node locations while dealing with the platform memory constraints. Figure 16 illustrates how the weights obtained by TASC can be used to drive computation during a beaconless localization process that forms a relative coordinate system for all the nodes. For readability purposes in this example, distance information is not included in the weight computation. The example network demonstrates the use of weights on a small network with one large cluster and a set of smaller peripheral clusters. The nodes marked with a triangle denote the leaders of each cluster after the completion of the leader election, and the numbers next to each node represent the computed weights.

[1]The XYZ node is available to other researchers from Cogent Computer Systems at the beginning of August 2004

[2]This is a requirement we have identified in our prior work. Although this is not completely defined in our implementation, we are aware of the existence of an algorithm based on rigidity theory [12] and work in progress in [17] that can identify uniquely localizable nodes. This however is not the main topic of our discussion in this paper.

The localization process is initiated at each cluster leader after a backoff period that is inversely proportional to the weight of the node. This implies that computation will start from a *locally isotropic* view point, at the leader with weight 22. First the leader identify the set of uniquely localizable nodes in its cluster and compute an initial coordinate system using the three-phase scheme described above in the XYZ node software implementation. The leader then transfers control to the next heaviest (but already localized) node found on the perimeter of the cluster. This node repeats the same process in finding a uniquely localizable component, but this time, the node seeks an overlapping set of already localized nodes in the cluster it belongs to and a set of unlocalized nodes outside its cluster (Figure 16b). This set of nodes is localized and control is transferred to the next heaviest node to continue the process until all the nodes are localized (Figures 16c,d). This incremental localization scheme has the three desirable effects. First, localization starts at the center of the network, fully utilizing isotropy and reducing error propagation. Second, this weight driven localization helps the formation of a *moving cloud of computation inside* the network. This limits the number of nodes a computing node has to consider to localize other nodes in its vicinity, thus making localization feasible on memory constrained nodes. The third desirable feature is that it reduces computation redundancy. For instance, in the distributed MDS-based scheme (MDS-MAP(P,R)) described in [24]. Each node computes a local coordinate system using information in its 2-hop neighborhood, followed by a coordinated transformation. If the same localization process is driven by weights then only a small fraction of the nodes need to perform the local coordinate system. This also limits the amount of coordinate system transformations required to establish a common coordinate system among all the nodes. This study case hints the use of a *multistart* TASC driven localization strategy outlined in the next subsection.

TABLE II
MEMORY REQUIREMENTS ON XYZ NODE

Code Component	FLASH(KB)	RAM(KB)
Basic OS	10.3	14.24
Floating Point Libraries	49.13	2.216
Beaconless Localization	22.272	4.1
Clustering	1.9	1.2

B. Study case 2: A Multistart Localization Strategy

The existence of locally isotropic clusters suggests a *multistart ad-hoc localization strategy* to reduce error

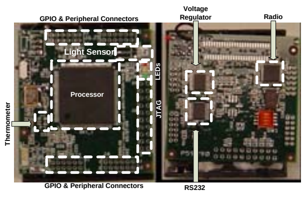


Fig. 14. The XYZ sensor node.

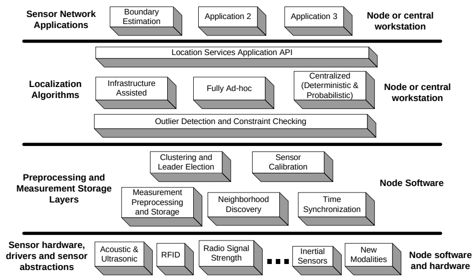

Sensor Network Applications
Boundary Estimation
Application 2
Application 3
Node or central workstation
Location Services Application API
Localization Algorithms
Infrastructure Assisted
Fully Ad-hoc
Centralized (Deterministic & Probabilistic)
Node or central workstation
Outlier Detection and Constraint Checking
Clustering and Leader Election
Sensor Calibration
Preprocessing and Measurement Storage Layers
Node Software
Measurement Preprocessing and Storage
Neighborhood Discovery
Time Synchronization
Sensor hardware, drivers and sensor abstractions
Acoustic & Ultrasonic
RFID
Radio Signal Strength
Inertial Sensors
New Modalities
Node software and hardware

Fig. 15. Layered view of localization protocol stack under development on XYZ

propagation. Localization can be initiated by cluster leaders in the denser and more isotropic parts of the network that are less likely to suffer from bad geometries. After its initialization, localization will begin to expand in multiple local coordinate systems beyond the boundaries of each cluster. Two local coordinate systems can merge with each other each time there is an overlap of three nodes. Our simulation has shown that this approach will work for uniform cases, but there are still issues that remain to be addressed to achieve efficient merging. At the same time, the merging of coordinate systems is a least square fit computation that may also induce errors due to bad geometry. The implications of error in coordinate merging are still not well understood and it is subject to further research.

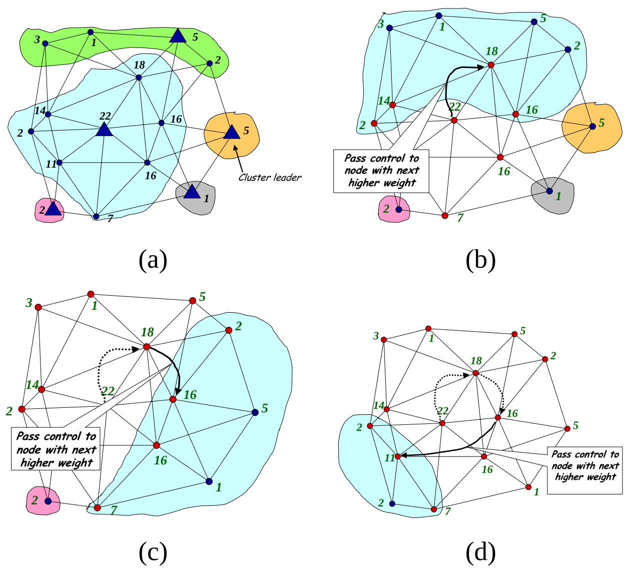


Fig. 16. An example of weight driven localization to limit error propagation

VII. Discussion

Despite the encouraging results on the behavior of TASC, we acknowledge that there are multiple issues to consider in realistic deployments. First the parameters of task (radius of neighborhood used, minimum cluster size, $MinPoints$ and the computation of weights) should be adapted to fit the particular application needs. The option of a node running multiple instances of TASC with different parameters is worth exploring. Second, the timing parameters of the algorithm should be more rigorously defined to comply with an actual deployment. For some systems where incremental deployment makes sense the leader election mechanisms need to be adapted to support the addition and subtraction of nodes from the network. Based on our experience from the simulation behavior and our efforts to build a scalable sensor network testbed, we believe that these changes are possible.

In addition to the features described here, weight computation in TASC can reveal important properties of a network topology that should be further investigated. For instance, in one particular implementation of weight computation, we found that we could discover with very high certainty the nodes that are found on the boundary of a network. Finally, one of the main drawbacks is the assumption that every node in the network is capable of performing distance measurements to its neighbors. An interesting extension would be to study how clustering behaves in the case of heterogenous measurement technologies with different measure confidence levels as well as the absence of distance measurements in some parts of the network.

VIII. Conclusions and future work

This paper presented a topology adaptive clustering algorithm for sensor networks. Our evaluation has shown that the novel combination of weights and density reachability achieves the desired behavior that is, it can decompose large networks into smaller locally isotropic clusters node locations are computed. The preliminary TASC implementation on a wireless sensor node has shown that the algorithm has a small memory footprint, and a lightweight implementation. This is because from the implementation perspective, the network level functionality required by TASC has a significant overlap with neighborhood discovery services; the inter-node distance measurements needed for distance based localization and the neighborhood coordination features a sensor node is likely to have to be able to collaborate with other nodes in its vicinity. We anticipate that such an algorithm would provide a useful service for sensor networks. The potential usefulness of TASC with node localization has been described in this paper. Other potential applications of TASC include the use of clustering to favor data aggregation as well as non-uniform spatial sampling. The distribution of weights inside a network can also be used as an indicator for spatial regularity in a specific deployment. One possible research avenue would be to develop and algorithm for making localized decisions on how nodes should reposition themselves to improve sampling uniformity. Another possibility is to repeat the weight-based election process to construct hierarchies. The initial results are encouraging and suggest the more rigorous evaluation of TASC needs in more realistic deployment settings. As part of our future work, we plan to test TASC in the context of our 3-D testbed comprised with XYZ nodes. The two immediate uses of TASC in our 100-node testbed is to assist with ad-hoc node localization and in radio frequency allocation through the meaningful, spatial decomposition of a dense Zigbee network.

ACKNOWLEDGMENT

The authors would like to thank Johnny Yeh and Praneeth Wanigasekera for their help with porting the embedded operating system and the implementation of the algorithms on the OKI THUMB processor.

REFERENCES

[1] S. Basagni, *Distributed Clustering for Ad Hoc Networks*, International Symposium of Parallel Architectures, Algorithms and Networks (I-SPAN'99), Fremantle, Australia, June 23-25, 1999.

[2] D. Bertsekas, *Dynamic Programming and Optimal Control*, vol 1, Athena Scientific, 2000

[3] McQueen, J. B., *Some Methods for Classification and Analysis of Multivariate Observations*, Proceedings of the Fifth Symposium on Math, Statistics and Probability (pp. 281-297), 1967.

[4] Kanugo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., Wu, A. Y., A *Local Search Approximation Algorithm for k-Means Clustering*, Proc. of the 18th Annual ACM Symp. on Computational Geometry, 2002, 10-18.

[5] GPSR B. Karp and H.T. Kung, *GPSR: Greedy Perimeter Stateless Routing for Wireless Networks*, Proceedings of Mobicom 2000

[6] LAR Y.B Ko and N. Vaidya, *Location Aided Routing (LAR) in Mobile Networks*, Proceedings of ACM/IEEE MobiCom, pp. 66-75, October 1998

[7] Klein, D., Kamvar, S. D., Manning, C. D., *From Instance-level Constraints to Space-level Constraints: Making the Most of Prior Knowledge in Data Clustering*, The Nineteenth International Conference on Machine Learning (ICML-2002), Sydney, Australia, July 8-12, 2002.

[8] Ghiasi, S., Srivastava, A., Yang, X., Sarrafzadeh, M., *Optimal Energy Aware Clustering in Sensor Networks*, Sensors 2002, 2, 258-269.

[9] Gupta, H., Das, S. R.,Gu, Q., *Connected Sensor Cover: Self-Organization of Sensor Networks for Efficient Query Execution*, MobiHoc'03, Annapolis, Maryland, USA, June 1-3, 2003.

[10] Chen, Y. P., Liestman, A. L., *A Zonal Algorithm for Clustering Ad Hoc Networks*, International Journal of Foundations of Computer Science, 14(2):305-322, April 2003.

[11] D. Ganesan, S. Ratnasamy, H. Wang and D. Estrin, *Coping with irregular spatio-temporal sampling in sensor networks*, in Proceedings of Second Workshop on Hot Topics in Networks (HotNets-II), Novemeber 2003

[12] T. Eren, D. Goldenberg, W. Whiteley, Y. R. Yang, A. S. Morse, B. Anderson and P. Belhumeu, *Rigidity, Computation and Randomization in Network Localization*, Proceedings of IEEE INFOCOMM, Hong Kong, March 7-11, 2004

[13] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*,1988, isbn = 0-13-022278-X, Prentice-Hall, Inc.

[14] Christofides, N., *Graph Theory: An Algorithmic Approach*, Academic Press London, New York, San Fransisco, 1975.

[15] NeslSim Website http://www.ee.ucla.edu/ saurabh/NESLsim/

[16] S. Pattem, B. Krishnamachari and R. Govindan, *The Impact of Spatial Correlation on Routing with Compression in Wireless Sensor Networks*, Proceedings of the Third International Symposium on Information Processing in Sensor Networks (IPSN'04), April 26 - 27, 2004, Berkeley, California, USA

[17] N. B. Priyantha, H. Balakrishnan, E. Demaine, S. Teller, *Anchor-Free Distributed Localization in Sensor Networks*, LCS Tech. Report 892.

[18] Ji, X., Hongyuan, Z., *Sensor Positioning in Wireless Ad-hoc Sensor Networks Using Multidimensional Scaling*, IEEE Infocom, March 7-11, 2004.

[19] Kelley, J. L., *General Topology*, Van Nostrand Reinhold, New York, NY, 1955.

[20] Ester, M., Kriegel, H-P., Sander, J., Xu, X., *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases With Noise*, 2nd International Conference on Knowledge Discovery and Data Mining (KDD'96), Portland, Oregon, 1996.

[21] S. Ratnasamy, B. Karp, S. Shenker, D. Estrin, R. Govindan, L. Yin and F. Yu, *Data-Centric Storage in Sensornets with GHT, A Geographic Hash Table*, Mobile Networks and Applications (MONET), Special Issue on Wireless Sensor Networks, Kluwer August 2003

[22] A. Savvides, H. Park and M. B. Srivastava, *The n-hop Multilateration Primitive for Node Localization Problems*, Proceedings of Mobile Networks and Applications, 8, 443-451, 2003

[23] A. Savvides, W. Garber, R. Moses and M. B. Srivastava An Analysis of Error Inducing Parameters in Multihop Sensor Node Localization, to appear in the IEEE Transactions on Mobile Computing, 2004

[24] Y. Shang, W. Ruml, *Improved MDS-Based Localization*, Proceedings of IEEE INFOCOMM, Hong Kong, March 7-11, 2004

[25] M. A. Spirito*On the Accuracy of Cellular Mobile Station Location Estimation*, IEEE Transactions of Vehicular Technology, Vol. 50 No. 3, May 2001

[26] Zaane, O. R., Lee, C-H., *Clustering Spatial Data in the Presence of Obstacles: a Density-Based Approach*, Sixth International Database Engineering and Applications Symposium (IDEAS 2002), Edmonton, Alberta, Canada, July 17-19, 2002.

[27] N. B. Priyantha, A. Chakraborty, H. Balakrishnan, *The Cricket Location-Support System*, Proceedings of 6th ACM Mobicom, Boston, MA, August 2000.

[28] A. Harter and A. Hopper, *A New Location Technique for the Active Office*, IEEE Personal Communications, vol. 4, No. 5, October 1997, pp.42 47.

[29] Ubisense website, http://www.ubisense.net