ASample: Adaptive Spatial Sampling in Wireless Sensor Networks

Piotr Szczytowski, Abdelmajid Khelil, Neeraj Suri
DEEDS, TU Darmstadt, Germany
Email: {szczytowski, khelil, suri}@informatik.tu-darmstadt.de

***Abstract*—A prominent application of Wireless Sensor Networks is the monitoring of physical phenomena. The value of the monitored attributes naturally depends on the accuracy of the spatial sampling achieved by the deployed sensors. The monitored phenomena often tend to have unknown spatial distributions at pre-deployment stage, which also change over time. This can detrimentally affect the overall achievable accuracy of monitoring. Consequently, reaching an optimal (accuracy driven) static sensor node deployment is generally not possible, resulting in either under- or over-sampling of signals in space. Our goal is to provide for adaptive spatial sampling. The key challenges consist in identifying the regions of over- or under-sampling and in suggesting the appropriate countermeasures. In this paper, we propose a Voronoi based adaptive spatial sampling (ASample) solution. Our approach removes unnecessary samples from regions of over-sampling and generates additional new sampling locations in the under-sampling regions to fulfill specified accuracy requirements. Simulation results show that ASample significantly and efficiently reduces the mean square error of the achieved measurement accuracy.**

***Keywords*-Wireless Sensor Networks; Adaptive Spatial Sampling; Reconfiguration;**

I. Introduction

The monitoring of physical phenomena constitutes a key application of Wireless Sensor Networks (WSN). One of the key metrics to assess the quality of monitoring is the accuracy of the conducted measurements. The accuracy depends on several aspects, with the prominent facets being: *network* and *phenomenon* related. The main network related aspects are the spatial distribution of sensor nodes (SN), sensors accuracy and communication reliability. The central, phenomenon-related aspects, include the temporal and spatial distributions/dynamics of the phenomena of interest. The former aspects can be considered by an appropriate design and maintenance of the WSN. However, the latter aspects are provided as ranges and hard to forecast, which complicates the design and maintenance of the WSN. A worst-case-driven uniform deployment is expensive, and may waste valuable resources such as bandwidth and energy. Consequently, an efficient and flexible strategy is required to maintain the desired monitoring accuracy depending on phenomenon-related and network-related factors.

For specified accuracy requirements, such a strategy consists of achieving efficient reconfiguration and maintenance options in order to react to the varying physical phenomena. The network can be reconfigured by adjusting networks resources on the basis of gathered profiles [4]–[6]. As the accuracy of spatial monitoring strongly depends on the SNs spatial distribution, the appropriate generation of spatial samples through suppressing some [17]–[19] or adding some [12], [14] is possible. There are two possible approaches for such spatial sampling adaptation. The first approach aims at optimizing the use of the resources already present in the network (self-reconfiguration). The second approach exploits the possibility of supplementing the network with additional SNs (maintenance). The scope/extent of WSN reconfiguration techniques depends on the available resources. In practical terms the optimization of resource utilization entails repositioning of the mobile nodes.

The challenge this desired adaptive spatial sampling poses is to locally identify both over- and under-sampled regions. In the regions of under-sampling, new sampling locations must be chosen, to bring the WSN within the accuracy requirements, while minimizing required resources. The redundant nodes from the over-sampled regions can be relocated to these new sensing locations. If the available resources are limited, then new sampling locations should be selected so as to minimize the unavoidable inaccuracy.

In the literature we identify some excellent initial efforts to provide adaptive sampling. [17]–[19] focus on over-sampled regions. [4]–[9], [12] provide centralized techniques for handling under-sampled regions. As the monitored phenomenon evolves, so must the WSN in order to sustain the fidelity of its measurements. Failing to do so results in (a) waste of resources in over-sampled regions and (b) lack of accuracy in under-sampled regions. Consequently, there is a need for a holistic approach that can identify both types of regions to drive adaptive sampling, and perform it in a distributed manner for meaningful adaptive reactivity in evolving WSNs.

A. Paper Contributions & Structure

This paper develops an efficient distributed technique for identifying under- and over-sampled regions. Our proposed approach (ASample) utilizes current measurement and SNs placements, (a) to insert new sampling locations or (b) to remove redundant spatial samples, in order to exactly meet the accuracy requirements. Our approach is holistic as it is valid for both over-sampling and under-sampling profiling. Accordingly, it simplifies network-level decisions on moving nodes from the over-sampled regions to the under-sampled

This research is supported in part by DFG GRK 1362 (TUD GKMM), EU INSPIRE, EU CoMiFin, and CASED (www.cased.de)

ones and for tuning the network sampling resolution according to the changes in the monitored phenomena.

The paper is structured as follows. The related work is discussed in Section II. Following the system model in Section III, we present the problem formulation in Section IV. Section V details the proposed adaptive spatial sampling (ASample) technique as the paper's main contribution. The evaluation of ASample is presented in Section VI.

II. Related Work

The majority of the existing adaptive spatial sampling approaches assume centralized knowledge of the network deployment. [4]–[10] collect the measurement data from the network to model the monitored physical phenomenon. On the basis of the model and the topology information, the new sampling locations are chosen so that they reduce entropy of the models. These form useful foundations to achieve high monitoring accuracy. However being centralized they incur high communication and energy overhead on resource limited sensor nodes, in particular for handling evolving phenomenon.

Other approaches target adapting the distribution of SNs to the distribution of the monitored phenomenon [12]–[14], or increasing the sensing capabilities in the area of the increased phenomenon activity [11]. [12]–[14] assume all network nodes to be mobile. In [12] each node tracks in the centralized manner the position of all other nodes. Nodes plan their movement trajectory based on the principle of attracting forces. A detected event is given an attracting potential and nodes travel a distance proportional to the "attracting" force. [13], [14] discuss the case of fleets of mobile robots where clustering is used to collect the local measurements and calculate the model of a local phenomenon. The model is used to determine the new sampling locations. The nodes move to the assigned positions and the process is repeated. As the decision is taken locally it is likely not optimal and requires an iterative multi-step movement. That incurs a high energy overhead. Our technique also takes local decisions, but the iterative movements are virtual until final positions are settled. The approach of [14] does not give accuracy guarantees.

[15] proposes a solution for the problem of the one-dimensional adaptive spatial sampling for water temperature. The goal of moving SNs on a line is to find the depth at which water temperature changes rapidly. Each SN is given a section to measure the variance of measurements at its ends. If the variance is above a specified threshold, the section is divided and the sub-sections are further evaluated. As the solution applies only for one-dimensional spatial scenarios, it has a limited application to the more common two-dimensional physical WSN environments considered in our work.

[17]–[19] consider the case of over-sampling. [17], [18] try to build the prediction models of the neighboring nodes and if they are successful, the modeled nodes are put in the sleep mode to save energy. [19] collects the data from a small subset of sensors nodes at the sink. This information is used to estimate the environment conditions of the monitored area. On the basis of this information, the sink selects the regions where additional nodes should be activated. This class of algorithms does not consider the under-sampled regions. While the reconfiguration options (putting SNs in sleep mode) are, at an abstract level, valid for our approach, they cannot be easily integrated with the under-sampling detectors. We provide an approach valid for both over- and under-sampled regions.

[24]–[32] also present a variety of adaptive sampling protocols, whose main objective is resource conservation by adaptive sampling in the time domain. These works are not directly related to the discussed problem.

The problem we are pursuing differs form the problem of assuring sensing coverage [20]–[23], which assumes that each SN can sense/detect an occurrence of some event in the given sensing radius. The goal therefore is to deploy the WSN so that each point of the area of interest is within the sensing radius of at least one SN. These works handle the network-related and ignore the phenomena-related aspects. On the contrary, we consider the scenarios where the phenomena change gradually over the monitoring area. The measurement at a given sampling location corresponds exclusively to the location where measurement was taken and has no defined radius of its validity. The phenomenon between two points can only be interpolated, thus bounding the difference between the closest sampling points helps decrease the interpolation error.

III. System Model

Conforming to the established WSN models, we assume a WSN consisting of a large number n of resource constrained SNs and a sink. The SNs have finite capacity battery power. The communication range r is limited, and two neighboring SNs can communicate only if the Euclidean distance between them is smaller than r. The communication dominates the imprint on the energy depletion of the SNs. We consider an arbitrary spatial distribution of SNs in the deployment area. We only assume that the full area is covered by these nodes and the resulting WSN is connected. The SNs know their position using on-board GPS receivers or alternative GPS-free techniques of localization [1]. We consider cases where (a) all nodes are static, (b) all nodes can move, or (c) a mix of static and mobile nodes. Our approach implicitly tolerates sensor node failures as this is equivalent to moving a node away from its position or putting it in sleep mode. Message loss will be explicitly considered while evaluating our approach. We investigate arbitrary continuous two dimensional signals in the sensor field.

IV. Problem Formulation and Objectives

For environmental monitoring applications it is crucial to reconstruct the spatial distribution of the signal with an acceptable accuracy. This may be transformed into specifying a desired maximally allowed value difference between any two neighboring sampling points Acc_{TH}, which we refer to by accuracy requirement. Therefore, it is important to bound the value difference between the measurements of any SN and its closest neighbors (in each direction on the plane) and consequently to reduce the possible variations of the signal.

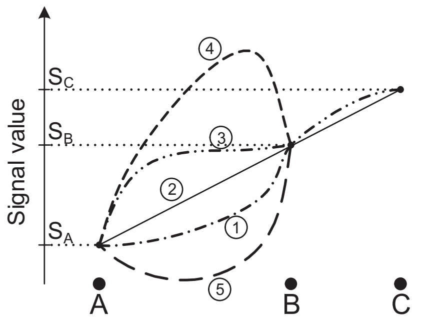

Fig. 1. Variants of signal distribution

Consider three SNs A, B and C, located in each others communication range (Fig. 1). Node A, B and C measure values S_A, S_B and S_C, respectively. We distinguish five, among infinitely many, interesting possibilities for the signal shape between A, B and C (curves 1 - 5 in Fig. 1).

If $S_B - S_A > Acc_{TH}$, then the accuracy requirement is violated by all five signals. This can be easily detected if nodes share their value with their closest neighbors. It is now crucial to select supplementary sampling locations where adding/relocating the SNs makes WSN adhere to the requirements of monitoring accuracy. The algorithm should minimize the number of resources needed for the reconfiguration.

If $S_C - S_A < Acc_{TH}$ the sample of Node B is redundant. Therefore, our holistic technique should help identify such redundant nodes, which can be used for the re-deployment in under-sampled regions.

If $S_B - S_A < Acc_{TH}$, then the accuracy requirements may still be violated by signals such as Signals 4 and 5. This situation can only be detected through additional measurements, which implies a more dense pre-deployment of sensor nodes. The detection of such local minima or maxima is not the focus of our work. However, our work is applicable if such local extrema can be detected.

Overall, our techniques react to the changes in the monitored area as the result of dynamics of the phenomenon.

V. The Adaptive Spatial Sampling Approach

We first describe the necessary preliminaries concerning the Voronoi diagrams as they form the basis for our approach. Subsequently, we present an overview of our approach, and then detail our profiling and reconfiguration techniques in order to provide for adaptive spatial sampling.

A. Distributed Voronoi Construction

As our goal is to bound the difference in values between the closest neighbors, we use the Voronoi diagram [2] approach to determine a set of closest neighbors of each node. Across the varied techniques for space tessellation, the Voronoi diagrams offer a simple and efficient approach. The overall space S occupied by SNs $s_i, 0 < i < n$ is divided into Voronoi cells (VC_i). Each (VC_i) contains SN and an area surrounding the node. Each point belonging to the Voronoi cell is closest to the SN s_i placed in cell than to any other SN (Eq. (1)), where δ is a distance function.

$$VC_i(S, s_i) = \{p \epsilon R^d \mid \delta(p, s_i) < \delta(p, s_j), i \neq j\} \quad (1)$$

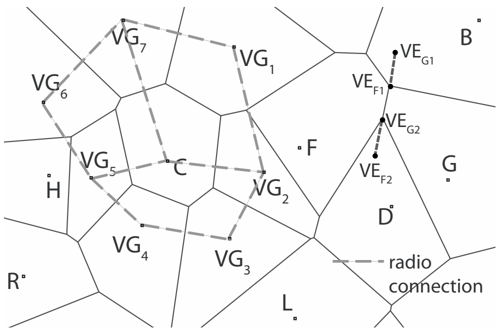

Fig. 2. Voronoi neighborhood

We define Voronoi neighbors (VG) as a set of SNs whose Voronoi cells share the border with the Voronoi cell of node s_i, which we call Voronoi edge. The Voronoi edge geometrically corresponds to a line segment (Fig. 2 Voronoi edge between Node C and Node VG_4) or a half line (Fig. 2 Voronoi edge between Node H and Node R) if the Voronoi cell is not completely bounded by other neighboring nodes. We also define a Voronoi polygon (VP_i) of node s_i as a set of its Voronoi edges.

The construction of the full Voronoi diagram requires $O(n \log n)$ operations [2] and may require also the global knowledge of the WSN topology by each SN. Such strict requirement makes construction of full Voronoi diagram impractical in the context of WSN. The cost of collecting the global topology information by each SN or by the sink renders this approach infeasible. Thus, for our algorithm we use a heuristic proposed by [3]. This heuristic determines Voronoi cell of each SN based on the information about the position of their 1-hop and 2-hop neighbors, instead of using the complete topology information. The use of only local information sacrifices some accuracy of the created Voronoi diagram for the reduced communication cost to construct it. Not all of the positions of 2-hop neighbors may be relevant for calculating the Voronoi diagram of a SN. To limit the amount of information transmitted, the SNs may pre-compute, which of their neighbors are relevant for submission as the 2-hop neighbors, e.g. Node VG_5 (Fig. 2) may omit transmitting the position of Node H to Node C. Node VG_5 knowing the positions of VG_4 and VG_6 may conclude that Node H cannot be a Voronoi neighbor of Node C.

B. Overview of our Approach

The first step towards an adaptive spatial sampling is for the SNs to locally check the fulfillment of the accuracy requirements. It is important to perform this in an efficient distributed manner as each redundant communication message reduces the battery lifetime of the involved SNs. This detection is based on the knowledge of SNs Voronoi diagram, which efficiently models each SN closest neighborhood. Each SN evaluates whether any of its Voronoi neighbors sensor readings violates the required accuracy. In such case the SN decides to place a virtual node (VN) at the Voronoi edge in order to separate the Voronoi neighbors whose values variance violates the required accuracy. The value of the VN is linearly interpolated and the Voronoi diagram is rebuilt including the new VN. As the approach is heuristic, the nodes are initially added as VNs, as some of them may be redundant. The redundant VNs can be removed so the SN proposes only the necessary sampling locations. The process iterates until the accuracy requirements are met by adding the VNs. For the situation of an evolving phenomenon, the SNs update their data to maintain accuracy.

C. The Proposed ASample Approach

We now develop two algorithms - Algorithm 1 for *adding virtual nodes*, and Algorithm 2 for *removing redundant nodes* that collectively constitute the proposed ASample technique.

1) Adding Virtual Nodes to Under-sampled Regions: After the Voronoi construction phase is completed, the SNs hold the list of their Voronoi neighbors. The SNs evaluate the list in search of nodes whose measurements differ from their own measurement by more than the requested accuracy Acc_{TH} (Algorithm 1, lines: 6-7). In case of detecting such disparity, both SNs agree on adding the VN. The placement of the VN is decided as follows. If the Voronoi edge separating neighboring Voronoi cells is a line segment (Fig. 2 - Voronoi edge of nodes C and VG_5), then the VN is placed in middle of this segment. The goal is to remove the Voronoi edge between the Voronoi neighbors violating the required accuracy. Placing the VN in the middle of the Voronoi edge maximizes the extent to which added VN separates the neighboring Voronoi neighbors (Alg. 1, line: 8). Otherwise, if the Voronoi edge is a half-line (Fig. 2 - Voronoi edge of nodes H and R) then a VN is added at the beginning of the half-line, as we cannot calculate the mid-point of a half-line. Both SNs sharing a common Voronoi edge will conclude the same decision of adding the VN. As we are adding a virtual node, then we require one of these both nodes to impersonate it. We refer to such a node as a *Proxy node*. The second node is referred to as a *Support node*. It has to be decided which node will assume the Proxy or Support role. The simple solution is to choose as the Proxy node the SN which has a lower measurement value. As both nodes know each other values, they agree on the roles to assume.

Due to the heuristic nature of the Voronoi construction, the Proxy and Support nodes may result in different views on a common Voronoi edge. In this case, the Proxy node sends to the Support node its proposition of the VN placement, with its view on the Voronoi edge. If the views differ (Fig. 2 - Node F view is segment $VE_{F1}VE_{F2}$, Node G view is segment $VE_{G1}VE_{G2}$), then a new view is chosen so that it consist of a section of line segments shared by both nodes. In this case that corresponds to the view of segment $VE_{F1}VE_{G2}$as the edge.

Adding a single VN between two Voronoi neighbors may not be sufficient to fulfill the accuracy requirement. This situation may arise from two reasons. The first results from the fact that the discrepancy between measured value may be greater than the double value of the accuracy threshold. In that case, placing the VN optimally between the Proxy and Support nodes, such it assumes a measurement value exactly between the values of Voronoi neighbors, still cannot resolve the inaccuracy problem. The second situation arises when the added VN still does not geometrically fully separates the Voronoi cells of the Proxy and Support nodes.

Algorithm 1 Adding Virtual Nodes

```
 1: var VoroNeighs=CreateVoronoi(self, 2_Hop_Neigh);
 2: while (!CheckAccuracyOK());
 3:
 4: function CheckAccuracyOK() : boolean
 5: var VirtualNodes;
 6: for all VN in VoroNeighs do
 7:     if abs(v(this) - v(VN)) $>$ $Acc_{TH}$ then
 8:         VirtualNodes.Add(middle(this.pos, VN.pos));
 9:     end if
10: end for
11: if VirtualNodes.Count $>$ 0 then
12:     VoroNeighs=CreateVoronoi(self,VoroNeighs $\cup$ VirtualNodes);
13:     return FALSE;
14: end if
15: return TRUE;
```

Consequently, the algorithm continues to iteratively add VNs until the accuracy requirements are met (Alg. 1, line: 2). After adding the VNs, the Proxy and Support nodes include the positions of the added VNs to recalculate the Voronoi cells (Alg. 1, line: 12). The complexity of recalculating the Voronoi cells may be reduced by considering only the positions of SNs that (at the previous step of iteration) were already Voronoi neighbors. An additional VN can only reduce the area of Voronoi cell. In order to recalculate the inaccuracy, the VNs also require to have some measurement value assigned. As the VN is added at equal distance between the Proxy and Support node, for this purpose their mean value is used.

It is also evident that the Voronoi neighbors of the Proxy and Support nodes should be informed about added VNs. Both Proxy and Support nodes calculate, which neighboring node's Voronoi neighborhood will be affected by adding VN and only these SNs are notified. To further limit the amount of transmitted information, the Proxy and Support nodes inform only their set of closest common Voronoi neighbors. The notification sent to selected Voronoi neighbor nodes includes the position and the interpolated value of VN.

2) Removing Redundant Nodes from Over-sampled Regions: As the described solution is based on a heuristic, it may add more VNs than required. To minimize the use of resources and the reconfiguration costs, it is desirable to find the redundant VNs. In the case of a network composed of only mobile nodes, it is important to find SNs which could move freely while not violating the accuracy requirements of the WSN. We describe here an algorithm that allows each node to determine its redundancy status locally.

The candidate Node C to become redundant, iterates through the list of its Voronoi neighbors (VG_i) (Alg. 2, line: 2). From Fig. 2 it is evident that by removing Node *C*, each of the VG_i nodes Voronoi polygon will grow in size and gain new Voronoi neighbors. Concluding from the property of the Voronoi diagram, the new edges of extended polygons will be created with the current neighbors of *C*. Therefore, for each Voronoi neighbor, *C* calculates local Voronoi polygons using the positions of the rest of the Voronoi neighbors (Alg. 2, line: 4). Next, for each Voronoi neighbor it checks whether any of its new neighbors sensing measurement diverges more than the required accuracy (Alg. 2, line: 5). If it is the case then Node *C* cannot be removed (Alg. 2, line: 6). In case that the removal of Node *C* does not cause local inaccuracy, further evaluation takes place. Each node informs its Voronoi neighbors about the possibility of its removal.

It may happen that, apart from Node *C*, some of its Voronoi neighbors may conclude that they are also potentially redundant nodes. At that point it is necessary to decide which node ultimately will be designated redundant. We propose here a criteria based on the estimated size of Voronoi cell of each candidate node. We justify this approach on the basis that the smaller the area of the Voronoi diagram (SN has to have many closely placed neighbors), the less representative is the sample and hence smaller is its impact on the overall measurement accuracy. Therefore, each candidate node sends, along with the notification about its potential redundancy, the information about its Voronoi cell area. After sending the notification, each node waits a pre-defined time t_{wait} for the response from the neighboring nodes in the case any of them declares itself redundant. If any node responds or the received responses carry smaller area value then the node becomes redundant status. It communicates to its Voronoi neighbors that it will leave the network and should not anymore be considered for calculating the Voronoi diagram. The nodes receiving this information recalculate their Voronoi polygons. If the node was the candidate node but got suppressed by a neighbor with a smaller Voronoi cell area, then it recalculates its chances to become redundant node. The process progresses as long as any of the nodes are eligible for becoming a redundant node.

Algorithm 2 Removing Redundant Nodes

```
1: function CheckRedundancy() : boolean
2: for all VN in VoroNeighs do
3:     VN_NewVoroNeighs=CreateVoronoi(vn,VoroNeighs\this\VN);
4:     for all VNh in vn_NewVoroiNeighs do
5:         if abs(v(VN) - v(VNh)) > $Acc_{TH}$ then
6:             return FALSE;
7:         end if
8:     end for
9: end for
10: return TRUE;
```

The relocation of SNs, taking only the accuracy constraints under consideration, may lead to the break-up in the network connectivity. To eliminate this threat, each SN also checks whether its removal will disrupt the connectivity among its Voronoi neighbors. As each SN already holds the list of Voronoi neighbors and their locations, such connectivity check is easy to execute. In Fig. 1 Node C can be safely removed as all other Voronoi neighbors VG_i can communicate.

We also notice that in some boundary cases the repetitive removal of the SNs may lead to the removal of the whole region of the network. This happens when the SN to be removed is in an area of local maximum or minimum of the measured value. To avoid/prohibit this situation we do not allow such nodes to be removed.

D. Coping with Dynamic Phenomena

The fulfillment of the accuracy requirements depends directly on the monitored phenomenon. Consequently, it is evident that as the phenomenon is changing, the algorithm requires a scheme to keep the data updated. We propose here a simple approach which allows keeping the data consistent and minimizes the SNs communication effort.

The accuracy requirements may be violated only when the measurement difference between two Voronoi neighbors (C, VG_i) exceeds Acc_{TH}. The constant notifications upon changes in the measurement value (eg. measurement oscillation) would incur large communication costs. The SNs know each other's measurement values when for the first time they collect the measurement and execute the evaluation. In order to limit the amount of exchanged data, SNs only need to inform about the substantial changes in the measurement values. On the other hand, our scheme should avoid the situation when the update does not take place despite the fact that accuracy Acc_{TH} was already reached. Assume that $v(C) < v(VG_i)$ and the current difference in the measurement values equals $\delta_v = |v(C) - v(VG_i)|$. Node C sends the update to VG_i when it decreases for $\gamma = 0.5 \times (Acc_{TH} - \delta_v)$. That is the highest possible value of γ for which we can be sure that even if the $v(VG_i)$ increases for the value of γ it would not require a reconfiguration. This prevents the case where the threshold Acc_{TH} is exceeded without having C or VG_i send a notification to each other. In most of cases the δ_v will just narrow with each update causing further updates. In order to prevent this effect, γ_{min} can be defined. Reaching the value of γ_{min} would already trigger a process of adding a new VN.

It is also possible that $v(C)$ will increase while $v(VG_i)$ decreases. At first this process will reduce the δ_v but after reducing it to zero δ_v will start growing. In this case γ should be adjusted to $\gamma = 0.5 \times (Acc_{TH} + \delta_v)$.

When searching for the redundant SNs, it is not necessary to use active data update as described. The active redundant SNs will not violate accuracy requirements. Therefore, the update can be relaxed to a desired frequency.

E. Reconfiguration Scenarios

The results produced by the ASample technique consist of a set of points selected for additional sampling. Along with the sampling locations, the trajectory connecting the Proxy and the Support nodes where a sampling location was added in between, can also be provided. At the given sampling point, the interpolated value may be different from the real one. To find the right position the mobile node may move along the trajectory and stop upon reaching the position where the measured value of the phenomenon equals the one which was calculated by algorithm for the VN.

The reconfiguration requires dissemination of gathered information, which can be discussed only in context of usage scenarios. At present we foresee three classes (Class 1 - 3) of usage scenarios where ASample can come into play.

Class 1 assumes no mobility in the network. For this purpose the SNs that detected violation of accuracy requirements should send the gathered information to the sink using existing in the network routing protocol.

Class 2 involves a WSN, where apart from static SNs, mobile SNs (more powerful nodes e.g.: robots) are present and navigate within the network in order to collect the measurements. The mobile SNs equipped with their own sensors can use the proposed sampling locations to optimize path planning and increase sampling resolution. Mobile SNs alternatively could also pick-up redundant static SNs and place them in the proposed locations. The static SNs do not need to route the locations information through the network, but only to piggy-back it when transmitting sensor measurements. In this scenario the ASample algorithm can be executed on the mobile robots, instead of the resource limited SNs. The robot collecting the measurements already holds the necessary information to calculate each SN Voronoi diagram and to find new sampling locations.

Class 3 involves network where a subset or all of SNs can move. That kind of WSN can adapt its spatial resolution in self organizing manner without involvement of outside operator. Such a scheme requires from the redundant nodes to announce their availability and from SNs detecting under-sampling to request new sampling locations. For this purpose adapted version of the [20] can be used. SNs use virtual quorum to announce and request resources.

Additional problem may arise when the number of redundant nodes is insufficient to provide for all requested sampling locations. The problem may be resolved using adapted version of the bidding algorithm proposed in [22]. Using this algorithm the SNs requesting new sampling locations place their bids. The value of the bid corresponds to the size of the area of the VNs on behalf of which the SN acts as a proxy.

VI. Performance Evaluation

In this section, we describe the evaluation settings and the metrics we have chosen for the evaluation. We evaluate the algorithms complexity concentrating on their implementation feasibility and the communication costs its execution incurs on the WSN. We also present the evaluation results and discuss their implications.

A. Evaluation Studies

Our evaluations settings are based on the reconfiguration scenario classes introduced in Section V-E. We designed five experiments. The first three correspond to the static WSN scenario where no form of mobility is provided. These experiments foresee supplementary deployment of additional SNs in the locations proposed by the ASample algorithm. The first experiment (*Add all*) deploys as many new SNs as ASample suggests. It is to evaluate the efficiency of ASample when sufficient resources are available. The second and third experiments (*Add 10%* and *Add 20%*) add only additional 10% or 20% new SNs (compared to the total number of SNs). If the computed number of the required sampling locations is higher than the available additional SNs, then the sampling locations with the larger Voronoi cell area are deployed first. The forth experiment (*Move only*) assumes a network where all SNs are mobile. In this experiment, we move only the redundant nodes to proposed locations. This evaluates the self-sustainability of the WSN. The fifth experiment (*Move & add*) allows for both moving of the redundant SNs and adding the additional SNs if still some sampling location are not occupied. This scenario tries to limit the amount of resources necessary to restore the network fidelity. We use simulated model of physical phenomena where its intensity $p = f(d)$, depends on the distance d to the center of the phenomenon. In our scenario, we model the phenomena using several distributions: exponential, pareto, normal and linear. Their parameters are adjusted that in the network area they assume values in the range between 0 and the peak value of 100 units in the center of phenomenon. We provide also an evaluation for the adaptability of ASample to the changes in the phenomenon activity. For this purpose we simulate the changes in the peak value of the phenomenon between 30 and 100 units. Throughout all experiments (if not stated otherwise) we fixed the deployment area to the size of 250×250m, the radio range to 25m and the number of SNs to 300. The parameters were chosen so as to generate a connected WSN [33]. We use our stand-alone implementation to simulate the ASample efficiency.

B. Evaluation Metrics

In order to quantify the accuracy enhancement of the phenomenon monitoring, we use a mean square error (MSE) based metric that we calculate as follows. We put a virtual grid over the network deployment area. At the coordinates of each vertex of the grid, we calculate the real value induced by the simulated phenomenon and subtract from it the value measured by the closest SN. We assume that the SN closest to the sampling point gives the best estimate of the value. For each simulation and proposed algorithm we calculate two MSE values. The first is performed as described above, the second (MSE30) applies only for the points where the value of the measured phenomenon is higher than 30% of its peak value. The reasoning behind the second value is to evaluate the considered algorithm in the areas of the phenomenon activity.

C. Complexity Analysis

We consider both the computational and the message complexity involved in the execution of ASample. The computation overhead is incurred primarily by the calculation of the Voronoi polygon on basis of the set of neighbors locations. For each location a bisectional line between the location and SN is calculated - a relatively simple algebraic operation. For each line then the possible intersection is checked against the segments and the semi-segments constituting till now calculated Voronoi polygon. In the worst case scenario there are $\frac{u^2}{2}$ operations, where u denotes the number of neighbors of SN. On average there will be $u \times VG_{cnt}$ operations, where VG_{cnt} is the final number of Voronoi neighbors. It should be noted that for most of SNs (not detecting adherence to the accuracy requirements) the calculation of Voronoi diagram takes place only at the initialization.

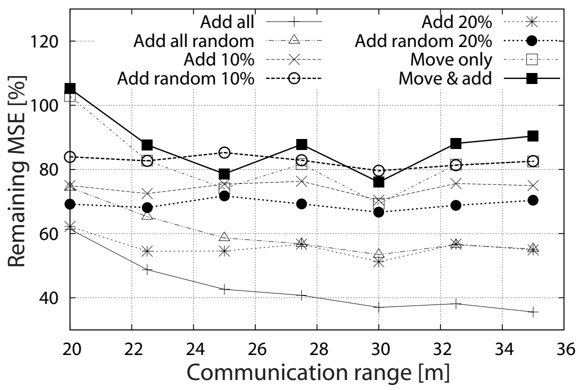

Fig. 3. Impact of communication range

Regarding the message complexity for the construction of the Voronoi diagram, each SN locally broadcasts twice 1-hop message. During the first broadcast the SN announces its ID. During the second one, it sends the list of neighbors which may be relevant for constructing its neighbors Voronoi diagrams. If the list of neighbors is too long it may be split into smaller ones containing the neighborhood information concerning only particular nodes and then directly addressed to proper nodes. SNs repeat this step only if a new SN is added or removed in their neighborhood.

When agreeing on new sampling locations, only up to 2 messages need to be exchanged between Proxy and Support nodes. Additional messages are sent only when the measured values change such at the notification of neighbors is requested to check for accuracy requirements.

D. Evaluation Results

Fig. 3 presents the results for all five described experiments (Sec. VI-A) and additionally three random deployments (the deployment of additionally 10%, 20% and the number of nodes added in the first experiment) for the reference. The results are presented for varied communication ranges, which have a direct impact on the accuracy of the created Voronoi diagram.

The best performing is the first experiment (*Add all*) as it is the only adding SNs and is not limited by their number. The reference experiment *Add all random*, which adds the same number of SNs but randomly, delivers the results, that in most of cases offer 20% less MSE reduction. Increasing the number of sampling points must lower the MSE value. As results show ASample utilizes the additional SNs more efficiently than the random approach. Pairs of experiments *Add 10%, Add random 10%* and *Add 10%, Add random 10%* show how efficiently ASample utilizes limited resources. Each pair shows that the reconfiguration executed using ASample preforms better by providing lower MSE values. In case of the first pair the results are approximately 10% and in case of the second pair up to 15% better than random deployment.

The strategies aimed at reducing the resource costs of reconfiguration *Move only* and *Move & add* also reduce MSE value but this is less significant as the number of sampling points is lower.

Fig. 4 shows ASample reaction to different accuracy requirement settings. As expected lower bound on difference in the values between neighboring SNs results in lower MSE values. It comes at the cost of increased resource usage. Adding the same number of SNs but randomly leads to higher MSE values. Using the strategy of moving redundant SNs significantly reduces the need for the new resources. When the accuracy bound is set to a large value, ASample puts up to 40% of the existing nodes in the redundant mode. This results in higher MSE values, which are consequence of lower accuracy requirements. We also show the impact of message losses on the efficiency of ASample. The last two experiments *Add all (msg loss)* and *# Add all (msg loss)* show the MSE value reduction and number of nodes added in environment where 20% of messages were lost. As a result, the accuracy of constructed Voronoi diagrams suffers and hence some violations of accuracy requirements are not detected. Consequently, less VN are deployed thus providing lesser reduction in MSE value compared to the lossless message scenario.

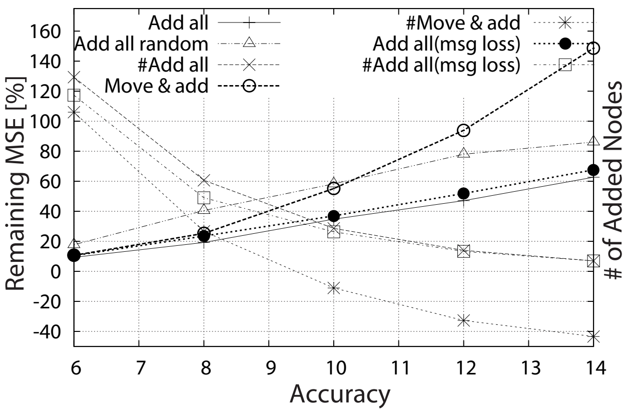

Fig. 4. Changing accuracy requirement

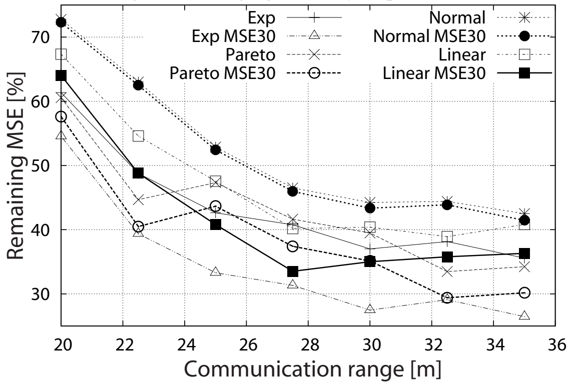

Fig. 5. Various phenomena models

Fig. 5 illustrates four pairs of experiments that show how ASample handles different types of phenomena. Each pair represents a different phenomenon distribution model (exponential, normal, pareto and linear) and reduction of the MSE values for the whole deployment area and the area where activity of the phenomenon reached at least 30% of the peak value. The curves of each pair are very similar, they differ only by the fixed offset. The MSE values for the activity area are better as the majority of SNs are added there. The exponential distribution shows the most significant difference and the reduction in the MSE values. This is because it represents

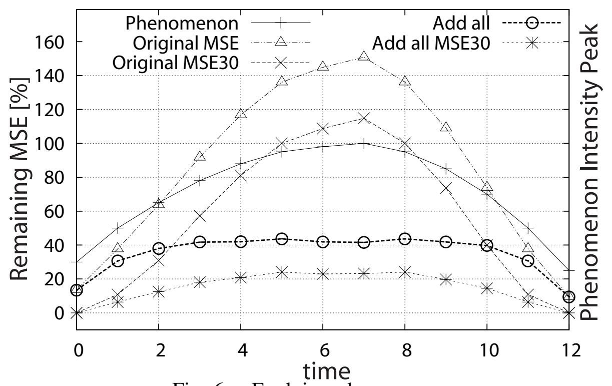

Fig. 6. Evolving phenomenon

the environment where the changes in the value show the largest volatility. The exponential distribution demands the highest number of SNs to add. On the other extreme, for the normal distribution the difference between the MSE values for the whole area and the activity region are negligible as the phenomenon changes gradually. Still, the case of normal distribution shows that ASample is able to significantly reduce the MSE value of the network.

Fig. 6 shows how ASample adjusts to the changes in the phenomenon. The *Phenomenon*-plot shows how the intensity of the phenomenon changes over the course of the simulation. The two curves *Original MSE* and *Original MSE 30* present how the absolute values of MSE change with the phenomenon. It can be seen that for the low intensities of phenomenon the original deployment may be sufficient, but as the phenomenon develops and its volatility increases the accuracy of measurements suffers. The two plots *Add all* and *Add all MSE30* show the MSE value with the additional sampling points proposed by ASample. Initially, the MSE values slightly rise but then saturate quickly and remain on a steady level. It demonstrates the capability of the algorithm to maintain the required accuracy.

VII. Conclusions and Future Work

In this paper, we have presented ASample, an efficient distributed adaptive spatial sampling technique. Using only local views, ASample is capable of detecting the regions of under- and over-sampling. In case of the under-sampled regions it proposes a set of new sampling positions and for the over-sampled regions it discovers redundant sensors. Both operations are executed in a unified manner. Such an holistic approach allows for self-sustainability of WSN. As our simulations show, our approach effectively and exactly maintains the required accuracy of the measurement and allows fulfillment of the stated accuracy requirements.

The Voronoi abstraction used for our techniques is a powerful abstraction, which can be used beyond the presented two-dimensional network deployment. In our ongoing work, we investigate how with a limited effort our techniques can be extended for application in the three-dimensional network deployments. Hereby, the neighboring SNs instead of sharing common edges, share common planes. Therefore, the additional sampling points can be found using the planes' center of gravity instead of a middle point of Voronoi edge.

References

[1] He, T. et al.: Range-free localization and its impact on large scale sensor networks, In: ACM Trans. on Embedded Computing Sys., 4(4):877–906, 2005
[2] Aurenhammer, F.: Voronoi diagrams - a survey of a fundamental geometric data structure, In: ACM Comput. Surv., 23(3):345–405, 1991
[3] Alsalih, W. et al.: Distributed Voronoi diagram computation in wireless sensor networks, In: SPAA, pp. 364, 2008
[4] Krause, A. et al.: Near optimal Sensor Placements: Maximizing Information while Minimizing Communication Cost, In: IPSN, pp. 2–10, 2006
[5] Guestrin, C. et al.: Near-Optimal Sensor Placements in Gaussian Processes, In: ICML, pp. 265–272, 2005
[6] Popa, D. et al.: EKF-based Adaptive Sampling with Mobile Robotic Sensor Nodes, In: IROS, pp. 2451–2456, 2006
[7] Bin, Z., Sukhatme, G. S.: Adaptive Sampling for Estimating a Scalar Field using a Robotic Boat and a Sensor Network, In: ICRA, pp. 3673–3680, 2007
[8] Rahimi, M. et al.: Adaptive Sampling for Environmental Robotics, In: ICRA, pp. 3537–3544, 2004
[9] Rahimi, M. et al.: Adaptive Sampling for Environmental Field Estimation using Robotic Sensors, In: IROS, pp. 3692–3698, 2005
[10] Popa, D. et al.: Adaptive sampling algorithms for multiple autonomous underwater vehicles, In: IEEE/OES AUV, pp. 108–118, 2004
[11] Wang, Y.C. et al.: Exploring Load-Balance to Dispatch Mobile Sensors in Wireless Sensor Networks, In: ICCCN, pp. 669-674, 2007
[12] Butler, Z., Rus, D.: Event-Based Motion Control for Mobile Sensor Networks, In: IEEE Pervasive Computing, 2(4):34–42, 2003
[13] Huguenin, K., Rendas, M.: Distributed adaptive sampling using bounded-errors, In: RoboComm, pp. 54, 2007
[14] Bin, Z., Sukhatme, G. S.: Controlling Sensor Density using Mobility, In: EmNetS-II, pp. 141–149, 2005
[15] Bin, Z. et al.: Adaptive Sampling for Marine Microorganism Monitoring, In: IROS, pp. 1115–1122, 2004
[16] Popa, D. et al.: Optimal Sampling using Singular Value Decomposition of the Parameter Variance Space, In: IROS, pp. 3131–3136, 2005
[17] Gedik, B. et al.: ASAP: An Adaptive Sampling Approach to Data Collection in Sensor Networks, In: IEEE TPDS, 18(12):1766–1783, 2007
[18] Arici, T., Altunbasak, Y.: Adaptive Sensing for Environment Monitoring using Wireless Sensor Networks, In: WCNC, pp. 2347–2352, 2004
[19] Willett, R. et al.: Backcasting: Adaptive Sampling for Sensor Networks, In: IPSN, pp. 124–133, 2004
[20] Wang, G. et al.: Sensor Relocation in Mobile Sensor Networks, In: INFOCOM, pp. 2302–2312, 2005
[21] Shih, K. et al.: PALM: A Partition Avoidance Lazy Movement Protocol for Mobile Sensor Networks, In: WCNC, pp. 2484–2489, 2007
[22] Wang, G. et al.: Proxy-Based Sensor Deployment for Mobile Sensor Networks, In: MASS, pp. 493–502, 2004
[23] Wang, G. et al.: Movement-Assisted Sensor Deployment, In: IEEE Transactions on Mobile Computing, 5(6):640–652, 2006
[24] Kho, J. et al.: Decentralised adaptive sampling of wireless sensor networks, In: AAMAS, pp. 55–62, 2007
[25] Zhou, J. et al.: Adaptive Sampling and Routing in a Floodplain Monitoring Sensor Network, In: WiMob, pp. 85–93, 2007
[26] Alippi, C. et al.: Adaptive Sampling for Energy Conservation in Wireless Sensor Networks for Snow Monitoring Applications, In: MASS, pp. 1–6, 2007
[27] Virrankoski, R., Savvidees, A.: TASC: topology adaptive spatial clustering for sensor networks, In: MASS, pp. 605–614, 2005
[28] Jain, A., Chang, E.: Adaptive sampling for sensor networks, In: DMSN, pp. 10–16, 2004
[29] Ting H., Zafer, M.: Adaptive sampling for transient signal detection in the presence of missing samples, In: MASS, pp. 760–765, 2008
[30] Bandyopadhyay, S. et al.: Spatio-temporal sampling rates and energy efficiency in wireless sensor networks, In: IEEE/ACM Trans. Netw., 13(6):1339–1352, 2005
[31] Lin, J. et al.: Accuracy Based Adaptive Sampling and Multi-Sensor Scheduling for Collaborative Target Tracking, In: ICARCV, pp. 1–6, 2006
[32] Ermiş, E., Saligrama, V.: Adaptive statistical sampling methods for decentralized estimation and detection of localized phenomena, In: IPSN, pp. 19, 2005
[33] Bettstetter, C.: On the minimum node degree and connectivity of a wireless multihop network, MobiHoc, pp. 80–91, 2002