An Adaptive Field Estimation Algorithm for Sensor Networks in Dynamic Environments

Amanda Prorok, William C. Evans and Alcherio Martinoli

Abstract— **The efficiency of distributed sensor networks depends on an optimal trade-off between the usage of resources and data quality. This workshop paper addresses the problem of optimizing this trade-off in self-configured distributed sensor networks. In our case-study example, we investigate a quadtree network topology and describe how we integrate a fully distributed node controller and field estimation algorithm. In a further step, we present a variant control algorithm, which continuously adapts network sampling and node activity to match spatio-temporal field variability. Realistic simulations are performed on the e-puck robot platform, and show that the proposed sampling strategy potentially economizes 20% of resource usage.**

I. Introduction

Since the beginnings of research on sensor networks in the 1970s, the monitoring of environments and habitats has become one of its major application fields [3]. Technological advances in embedded systems, such as the development of reliable wireless communication, and miniaturization and improved efficiency of microcontrollers and sensors have have answered key needs, and encouraged an increasing deployment of wireless sensor networks as a main tool to monitor spaces [8]. Still, one of the challenges presented with the deployment of sensor networks is the accurate estimation of fields with unpredictable environmental phenomena, while simultaneously addressing the critical issues of resource usage such as local memory, communication and processing constraints.

With networks often consisting of a considerable number of sensor nodes, the necessity of limiting energy consumption as well as bandwidth requirements increases. Research in the domain of ad hoc wireless routing has produced a range of algorithms which propose solutions for these problems. Improved routing algorithms have been developed which aim to accomplish in-network load balancing and an increased system lifetime, employing techniques that are mostly based on system information such as remaining energy levels and routing capacities.

A. Spatial & Temporal Suppression

There are two main approaches to optimizing the energy consumption of sensor networks. In temporal suppression schemes, each node uses its own history of measurements to determine if a new value can be inferred by the network sink instead of being transmitted, or even to avoid sampling and local processing entirely. A simple example would be transmitting measurements only when they differ from the previous value. Typically these approaches make use of much more complex models, often providing bounded error.

The Probabilistic Adaptable Query (PAQ) system is one notable such scheme based on time series forecasting [23]. It uses autoregressive models maintained locally per sensor node in order to keep from sending data directly to the sink. Instead, nodes communicate model parameters as necessary in order to keep the sink's predictions within some defined error bound. Tulone and Madden extend this work with their Similarity-based Adaptive Framework (SAF) [24], adding robustness to quick changes in data trends as well as a location-independent clustering technique that allows the detection of redundant nodes.

On the other hand, spatial suppression exploits spatial correlations between nearby sensor nodes in order to reduce communication load. Many spatial suppression algorithms attempt to detect and deactivate sets of redundant nodes. Arici and Altunbasak propose using a first-order model to determine the predictability of particular nodes [1]. They define some of the nodes in the network as *macronodes* which attempt to fit a plane over their neighbors' positions and data, commanding easily predictable nodes to stop reporting measurements for some period of time. Similarly, Willett et al. define the idea of a *fusion center* that is responsible for estimating a field based on received sensor measurements and then directly deactivating redundant nodes [26].

Chu et al. propose the use of replicated dynamic probabilistic models between the sink and *disjoint cliques* of data sources [4]. The sink then uses these models to predict future sensor data. If the *root* of a clique observes data inconsistent with the sink's current prediction model, a subset of the clique's recent observations are sent and the sink's model is updated as necessary.

B. Motivation

In our work, we address the problem of designing distributed sensor networks for surveillance and monitoring. It is clear from [14] that self-configuration is a necessary element for effective as well as efficient performance of such networks. The proposed design paradigm suggests hierarchical topologies, following a top-down control and bottom-up reconfiguration principle. Here, we build upon this design rule, implementing a distributed, multi-layer tree-based routing algorithm and combining it with a threshold-

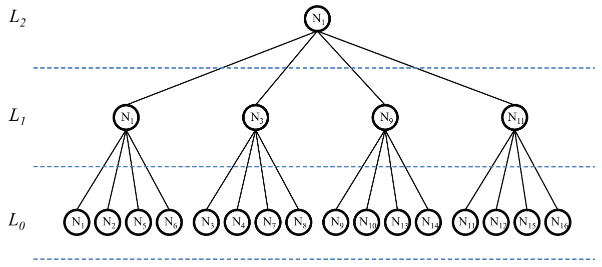
Amanda Prorok, William C. Evans and Alcherio Martinoli are with the Distributed Intelligent Systems and Algorithms Laboratory of the Ecole Polytechnique Fédérale de Lausanne. This work was partially sponsored by the Swiss National Science Foundation (Contract Nr. PP002-116913). William C. Evans is currently sponsored by the NCCR Stabilization Measures Initiative (project "Tamperproof Monitoring Solution for Weather Risk Management"). `firstname.lastname@epfl.ch`

based clustering strategy which is adaptive to the state of the field being estimated. Our algorithm leans on established field estimation methods described in [18] and [26]. The approach is similar to the one described by Arici et al. in [1], which describes an adaptive sensing method also based on a tree-like, hierarchical network structure. Their method exploits the fact that a manual deployment of sensors may offer more information than necessary (over time and space) to reconstruct an accurate field estimate. They propose a self-configuration algorithm which will put nodes into passive mode when their measurements become 'predictable'. Here, also motivated by previous research in the domain of distributed sensor node controllers as presented in [7], we develop a fully distributed node controller that is easily implemented on resource constrained and noisy hardware, which aims to optimize system performance by finding a trade-off between use of resources and data quality. In contrast to the methods described in [2, 13, 27], we base our clustering strategy on field data, rather than on system information. Also, our resulting data aggregation method follows a multi-layer bottom-up principle, which enables global abstraction of the target field, different from the local collaborative processing methods of [15, 28]. Lastly, in contrast to [18] and [26] we focus on the whole system rather than only on communication and routing activities, and our work in [19] demonstrates the approach on real hardware by comparing the performance to theoretical predictions.

The method in this work especially targets heterogeneous sensor-networks, given its non-homogeneous communication constraints. This allows for the deployment of large numbers of cheap sensor nodes to increase granularity, while more expensive, robust sensor nodes are placed at strategically important positions. Nevertheless, in order to guarantee the scalability and robustness of the system, redundancy must be foreseen by implementing efficient role selection strategies. Finally, although our current algorithm does not explicitly take into account node mobility, its design easily accommodates extensions such as node redeployment or network reconfiguration. This capability may equally be deployed non-homogeneously throughout the sensor network.

II. Spatial Suppression using Hierarchical Network Topologies

In accordance with our above-mentioned motivation to port our algorithms onto mobile platforms, we base the following elaborations on robotic sensor networks. As suggested in the theoretical work of [26], we superpose a quadtree (Fig. 1) on the robotic sensor network. Especially when computing spatial problems typical in computer aided design and geo-data applications [12], the quadtree data structure has proven an efficient and powerful tool [11, 20]. An early work in [10] shows how an active quadtree network facilitates image representation and analysis. Also, a recent study in [9] shows how a quadtree can be utilized for in-network data querying in a fully distributed wireless network.



(a) Quadtree hierarchy

Fig. 1. A 16-node quadtree structure. The quadtree hierarchy is decomposed into 3 hierarchy levels. A node will participate in either of the 3 subsets: $\{L_0\}, \{L_0, L_1\}$ or $\{L_0, L_1, L_2\}$.

A. Distributed Network Organization

Here, although our controllers and models are general to any hierarchical topology, we showcase our study on a quadtree based network with each robotic node within our sensor field representing a leaf node in the tree structure. The robots are distributed on a regular grid in a square arena. In a network of a total n nodes, assuming that the robots are aware of their location, each one allocates itself to one of n sensing cells in the decomposed space. We thus obtain a robotic sensor network ordered by the intrinsic hierarchy of the quadtree. Adapted and implemented in a fully distributed sensor network, this hierarchy can be explored in terms of i) communication channels and ii) fine-tuning the spatial resolution of the sensor network. Whereas exploring i) is relatively straightforward as we can directly exploit the quadtree hierarchy, there are many approaches to ii)—our chosen approach will be discussed later in Section II-B.2.

On a global level, the quadtree structure depends only on the number of nodes (implicitly a power of 4), and can be constructed in a distributed manner, assuming that all nodes know their location. As is evident in Fig. 1, a single node may have multiple roles within the network, depending on the status of the network. Thus, we create the notion of layers L_i. In a network of 4^K nodes, we have $K+1$ layers $(L_0, ..., L_K)$, and a node's current role in the network is defined by its current processing layer $L_{current}$. Every node N_i has a maximum layer $L_{k_{max}}$ with $N_i \in L_{k_{max}}$ such that there is no $k > k_{max}$ with $N_i \in L_k$. Also, any node N_i in $L_k, k > 0$ is a clusterhead, with four descending nodes in L_{k-1} as its cluster children (including itself). Lastly, for the sake of clarity, we don't go into the details of an eventual clusterhead rotation or election strategy.

The group of robotic nodes uses wireless communication as a means of inter-node organization. There are two classes of messages being used within the network: control messages and data messages (measurements). The messages typically contain the following elements: *control* or *measurement data*, i and k, with i the id of the sender node N_i and L_k its current processing layer. Control messages are sent top-down through the network structure, and measurement messages bottom-up. Nodes throughout the network or within the communication range of the transmitting node may receive messages at all times and asynchronously from various senders.

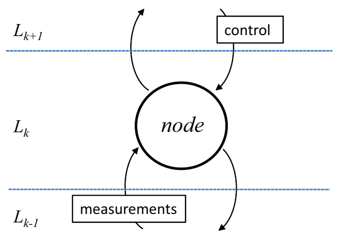


Fig. 2. The node is currently processing data in layer L_k. Measurement messages are sent bottom-up and control messages are sent top-down the quadtree structure.

A clusterhead will only accept measurement data from nodes belonging to its cluster, and following the top-down control principle, a node will only accept control messages from its clusterhead. Fig. 2 illustrates the communication protocol.

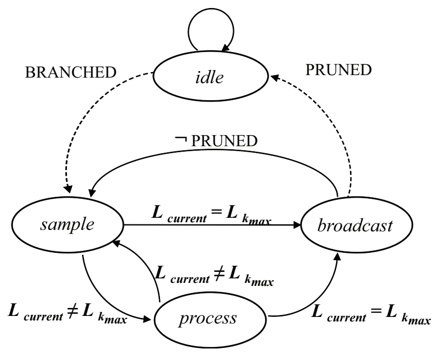
B. Control of the Robotic Node

We elaborate two control variants: first, a *naive sensing* strategy (NS), and second, an improved *threshold-based sensing* strategy (TBS). With NS, the nodes are in one of three possible states, whereas with TBS, the nodes are in one of four possible states. The controller is simple and distributed, homogeneous on all nodes.

1) State Machine: The controller can be represented by a simple state-machine, and is depicted in Fig. 3. Initially, a node is in the *sample* state. Each time a node takes a measurement, it will transition to the *process* state. If the node is a leaf node (its processing layer is $L_{current} = L_{k_{max}}$ at all times) it will transition directly to the *broadcast* state, send its measurement and then return to the *sample* state. If the node is a clusterhead, it will increment its processing layer $L_{current}$ once it has received (and aggregated) the data from all the nodes in its cluster, and will enter the *broadcast* state if it has reached its maximal layer $L_{k_{max}}$. Otherwise, it will re-enter the *sample* state. Finally, upon sending the (collected) measurement data in the *broadcast* state, the clusterhead will return to the *sample* state.

In a further step, we develop the controller for TBS, with the goal of optimizing the use of resources by reducing the number of messages sent and measurements taken. The aim is to *prune* certain node-clusters off the quadtree by putting the nodes in those clusters to sleep. A clusterhead will then replace measurement values of all its descendant nodes with its own. A fourth state is added to the NS controller, and is illustrated by dashed line on the right-hand side in Fig. 3. If a node has received a relevant pruning control message, it will be absorbed by the *idle* state.

2) Threshold-Based Pruning Algorithm: In TBS, a clusterhead makes the decision to prune or not prune its child nodes. Thus, we implemented a threshold-based pruning algorithm, which builds on the theoretical formula proposed in [18]. Assuming that the field is anisotropic, the chosen approach is to prune sensor-node clusters which are sampling values in isotropic subparts of the field. The resulting field estimator will display a higher sensing resolution along the boundaries of the anisotropic field and lower resolution in the isotropic subparts. This principle is illustrated by the



idle
BRANCHED
PRUNED
¬ PRUNED
sample
broadcast
process
$L_{current} = L_{k_{max}}$
$L_{current} \neq L_{k_{max}}$
$L_{current} \neq L_{k_{max}}$
$L_{current} = L_{k_{max}}$

Fig. 3. Schematic illustration of two variant state-machines implemented for the quadtree structure. (a) NS (without dashed line): A node samples environmental events. Measurement data from cluster nodes is received and processed. When the cluster data is complete, a node will broadcast the collected data. (b) TBS (with dashed lines): A node which is shut down is absorbed by the idle state. If change is perceived an idle node may re-enter the sampling state.

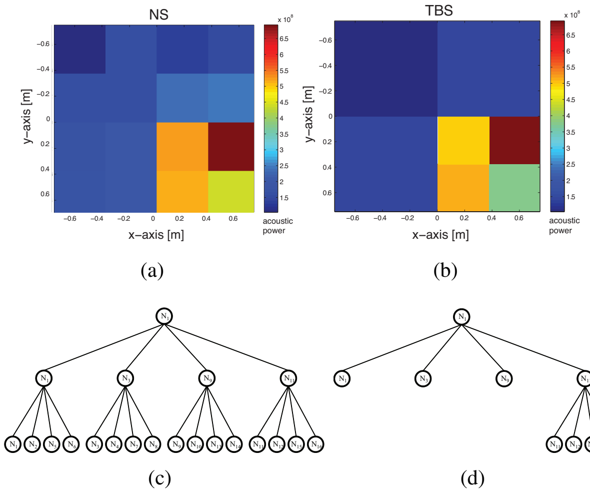


Fig. 4. The graphs show the calculated power of an acoustic event at a given moment. Each of the 16 cells is occupied by one robotic sensor node. An acoustic source is located in the bottom left corner of the arena. (a) A snapshot of the true field values (b) The data sent out of the network by the top-level node after completion of the pruning algorithm

example in Fig. 4. Fig. 4 (a) and (c) show a fully active (un-pruned) quadtree and the values transmitted by the full network, whereas Fig. 4 (b) and (d) show a pruned quadtree and the values transmitted by the remaining active nodes.

The following formal details are as previously elaborated in [19]. From [18] we have

$$\hat{f}_n = \underset{f(\theta),\theta\in\Theta_n}{\operatorname{argmin}} \; R(f(\theta),x) + 2s^2 p(n)|\theta| \qquad (1)$$

where s^2 is the signal noise variance and $p(n)$ a monotonically increasing function of the total number of nodes. The finite set Θ_n includes all possible pruning variations (partitions) of a quadtree with n nodes, and θ is one particular partition. Then, for the set of partitions Θ_n, the algorithm will seek the optimal partition θ which minimizes the cost of the resulting field estimator, $\hat{f}_n$. This cost is comprised of two terms. The first term $R(f(\theta),x)$ is the approximation error resulting from the pruned clusters in the partitions. The error is calculated as in

$$R(f(\theta), x) \quad = \quad \sum_{i=1}^{n} (f_i(\theta) - x_i)^2$$

where $f_i(\theta)$ is the estimated value for a node N_i in a particular partition θ and x_i is the true field value. The aim of the second term in (1), $2s^2 p(n)|\theta|$, is to penalize increasing complexity, where the factor $|\theta|$ is the number of not pruned nodes in the partition. In [17], $p(n) = 2/3 \log n$ and s^2 is homogeneous on all sensor nodes.

We can solve equation (1) in a distributed manner by using the bottom-up messaging protocol mentioned in Section II-A. The work in [17] confirms that both terms of the estimator are additive functions, thus the error and the penalty cost of a subsquare can be calculated by each corresponding clusterhead independently. Then, following our messaging protocol, a clusterhead in the quadtree hierarchy will receive from its 4 child nodes (three child nodes and itself) the field estimate which minimizes the estimation cost as given by the formula.

In order to implement the field estimation technique in our distributed network, we propose a threshold-based pruning algorithm. We are interested in studying the performance of a fixed-size sensor network in function of a threshold T_k. At layer L_0, there is no propagated error from lower levels, the cost $\hat{f}_i(\theta_{L_1})$ at a clusterhead N_i is thus equal to

$$\hat{f}_i(\theta_{L_1}) = \begin{cases} 8s^2 p & \text{if not pruning} \\ R(f_i(\theta_{L_1}), x) + 2s^2 p & \text{if pruning} \end{cases}$$

The algorithm will seek the minimal cost $min\{\hat{f}_i(\theta_{L_1})\}$, therefore the threshold on the approximation error $R(f_i(\theta_{L_1}), x)$ for layer L_1 is

$$T_{1,i}(s, p) \quad = \quad 6s^2 p$$

In other words, if the approximation error $R(f_i(\theta_{L_1}), x) < T_{1,i}(s, p)$, the cluster will be pruned. For layers L_k with $k > 1$, the estimator takes into account the propagated errors and complexity penalizers from lower level layers, with

$$\hat{f}_i(\theta_{L_k}) = \begin{cases} \sum_{j \in C_{k,i}} \hat{f}_j(\theta_{L_{k-1}}) & \text{if not pruning} \\ R(f_i(\theta_{L_k}), x) + 2s^2 p & \text{if pruning} \end{cases}$$

where $C_{k,i}$ is the set of all children nodes of clusterhead N_i at layer k. Since the network size is fixed, p is constant and the threshold $T_k(s)$ for level L_k, $k > 1$ is then

$$T_{k,i}(s) \quad = \quad 6s^2 p + \sum_{j \in C_{k,i}} R(f_j(\theta_{L_{k-1}}), x) \qquad (2)$$

3) Branching Algorithm: Sensor networks often deal with non-static environments. In order to take into account these changes in the environment, we extend the pruning algorithm elaborated above in order to enable an adaptive pruning behavior. We develop a branching mechanism, which enables initially pruned nodes to resume their full activities (sampling, data processing and message sending). This behavior is illustrated by the left dashed arrow in the state-machine depicted in Fig. 3. In contrast to the controller described in Section II-B.2, where pruned clusters remain pruned, nodes can now potentially receive reactivation signals enabling entire clusters to *branch*.

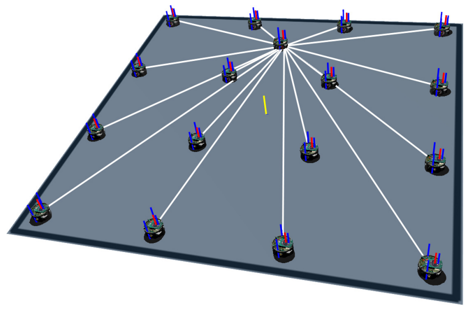


Fig. 5. The figure shows a screenshot from the Webots simulation environment. 16 robotic nodes (e-pucks) are evenly spaces out in a $1.5 \times 1.5\, m^2$ large space. The links show the detection of the acoustic source, a 17^{th} robot, placed in the top half of the arena.

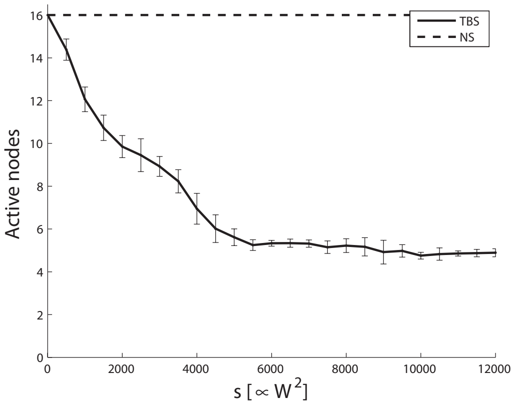
Intuitively, we might implement a simple branching algorithm by defining a constant time interval, at which a branching control message is sent to all nodes within the network. Yet, defining an optimal constant branching interval a-priori may be difficult or even impossible, due to the unknown and unpredictable characteristics of environmental phenomena. Thus, we developed a simple distributed strategy which will branch pruned clusters as a function of change perceived in the environment by the active nodes. This strategy exploits the fact that in a dynamic environment, the boundaries of an anisotropic field are moving. Thus, according to our threshold-based pruning algorithm, in a dynamic environment, active nodes may eventually be pruned as at they no longer cover anisotropic parts of the field. Each time an active node is pruned, it signals the need for a reevaluation of the current quadtree partition. Hence, the quadtree will branch if for a node i

$$R(f_i(\theta_{L_k}), x) \leq T_{k,i}.$$

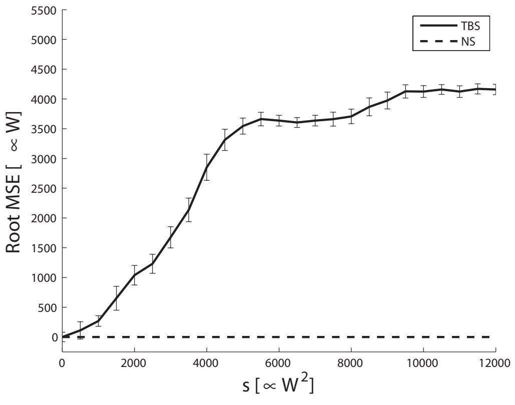
Following this additional threshold-based rule, active nodes in isotropic parts of the field will send branching control messages to pruned nodes in the quadtree.

III. Results

We designed an experimental setup using the robotic simulation software Webots [16]. Our robotic nodes are modeled by simulated e-puck robots [5] (which run on a microcontroller of the dsPIC30 family). The robots have a trinaural microphone array, enabling them to detect acoustic events, and are equiped with radio modules enabling short range communication [5]. An additional robot plays the role of a sound source, which will, depending on the experiment, remain stationary, or move randomly about the arena, avoiding the other robots and boundaries (Braitenberg vehicle with a speed of one robot-size per second). As elaborated in previous work by Cianci et. al. [6], the dynamics of the sound source are accurately modeled, taking into account reflection,

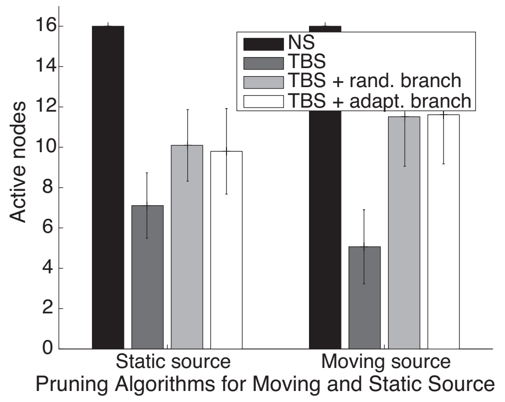


(a)

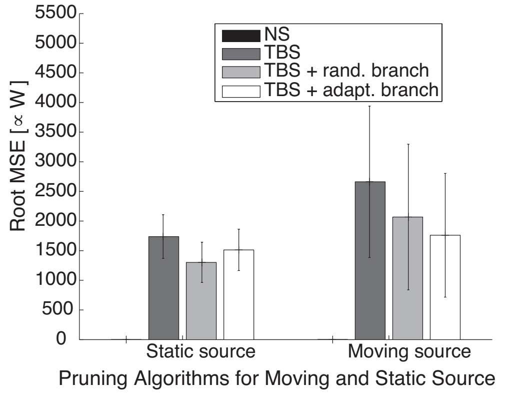


(b)

Fig. 6. Performance with i) NS and ii) TBS. 500 runs were performed per threshold, for 24 different thresholds with s in [0..12000]. (a) Total active nodes (b) MSE. The errorbars show a 95% confidence interval.



(a)



(b)

Fig. 7. The graphs show the MSE and average number of active nodes, for the quadtree structure implemented with three different controlling algorithms. The error-bars show the standard deviation.

fading and mixing. Also in [6], the radio communication is realistically modeled within the simulation software using a plugin based on *OMNeT++* [25], which accurately simulates the physical layer (i.e., with channel fading) and data link layer (i.e. modulation properties, channel coding, MAC protocol).

Fig. 5 shows the experimental setup with 16 robotic nodes spaced out evenly in a $1.5 \times 1.5\,m^2$ arena. The sound source in this setup generates a continuous, local acoustic field. The robotic nodes in the network sample at a frequency of approximately 288 kHz, take measurements at regular intervals of 256 ms, and calculate the power of this acoustic event. Figured 6 (a) and (b) summarize the behavior of the two control variants NS and TBS as elaborated above, with respect to (a) the number of active nodes and (b) the MSE. We performed 500 runs per threshold, for 24 different thresholds with s in [0..12000]. For NS, the total number of active nodes as well as the resulting MSE will remain constant. As expected for TBS, we observe a decreasing number of active nodes and an increasing MSE as the threshold increases.

Figures 7 (a) and (b) show the performance of the sensor network with four variant control algorithms: NS, TBS, TBS with random branching and TBS with adaptive branching. We see that in comparison with the pruning control TBS, adaptive branching reduces the resulting MSE for a moving sound source. Also, the number of active nodes is reduced by over 20% with respect to a fully active network as in NS. Post-evaluation of the data gathered by the adaptive pruning algorithm shows that in 42% of the time, the quadtree was branched. Thus, in order to better evaluate the adaptive pruning controller, we implemented a random branching mechanism with an equivalent branching probability instead of the threshold-based branching rule. We see that for both branching mechanisms the MSE is nearly identical, but that in the case of a dynamic environment, the adaptive algorithm outperforms the random one.

IV. Conclusion & Outlook

In this work we first developed a layer-based fully asynchronous distributed node controller, specific to hierarchical network topologies, and we implemented a self-configuration method based on an estimation technique. Whereas the theory for the estimation technique optimizes communication costs, we decoupled our performance metric by considering a sensor-node as either fully active or shut-down. In our previous work [19], we additionally verified the system's performance on hardware, and developed a probabilistic model that accurately captured the behavior of a real sensor network. Also, we developed a framework which ultimately allows for a specific, user-defined trade-off between the cost and accuracy of a sensor network. Beyond our previous work, this paper explores the feasibility of an augmented node control that envisions the reactivation of nodes absorbed by the idle state through the branching of pruned quadtree nodes. With our simulation results, we showed how the proposed quadtree branching algorithm may lead to significantly reduced resource usage without compromising the quality of the data obtained.

There are a number of possible extensions to this work, but most importantly, the introduction of clusterhead rotation cycles and distributed node responsibilities lead to increased robustness, which is a key factor for large-scale networks. Building upon the current baseline method, we will explore how the controlled movement of robotic sensor nodes affects the spatial resolution of the sensor network as a whole, thus also affecting its performance. Simultaneously, we will explore how to optimally allocate nodes in heterogeneous networks. To this purpose, we will employ SensorScope stations [21] as well as flying robotic vehicles [22] for outdoor operation, offering a promising set of tools for validating our future approaches on mobile systems as well as in outdoor scenarios.

References

[1] Tarik Arici and Yucel Altunbasak. Adaptive sensing for environment monitoring using wireless sensor networks. In *IEEE Wireless Communications and Neworking Conference (WCNC)*, pages 2347–2352, Atlanta, GA, USA, March 2004.

[2] Benjie Chen, Kyle Jamieson, Hari Balakrishnan, and Robert Morris. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 85–96, 2001.

[3] Chee Yee Chong and S. P. Kumar. Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, August 2003.

[4] D. Chu, A. Deshpande, JM Hellerstein, and W. Hong. Approximate data collection in sensor networks using probabilistic models. In *IEEE ICDE*, pages 48–48, 2006.

[5] Christopher M. Cianci, Xavier Raemy, Jim Pugh, and Alcherio Martinoli. Communication in a swarm of miniature robots: The e-puck as an educational tool for swarm robotics. In *Simulation of Adaptive Behavior (SAB-2006), Swarm Robotics Workshop*, pages 103–115, Rome, Italy, October 2006, Lecture Notes in Computer Science (2007), vol. 4433.

[6] Christopher M. Cianci, Jim Pugh, and Alcherio Martinoli. Exploration of an incremental suite of microscopic models for acoustic event monitoring using a robotic sensor network. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3290–3295, Pasadena, CA, USA, May 2008. IEEE Press.

[7] Christopher Michael Cianci. *Distributed intelligent algorithms for robotic sensor networks monitoring discontinuous anisotropic environmental fields*. PhD thesis, Lausanne, 2009. URL `http://library.epfl.ch/theses/?nr=4247`.

[8] David Culler, Deborah Estrin, and Mani Srivastava. Overview of sensor networks. *IEEE Computer, Special Issue in Sensor Networks*, 37(8):41–49, August 2004.

[9] M. Demirbas and Xuming Lu. Distributed quad-tree for spatial querying in wireless sensor networks. In *Communications, 2007. ICC '07. IEEE International Conference on*, pages 3325–3332, June 2007. doi: 10.1109/ICC.2007.551.

[10] T. Dubitzki, A. Y. Wu, and A. Rosenfeld. Parallel region property computation by active quadtree networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 3(6):626–633, 1981.

[11] R. A. Finkel and J. L. Bentley. Quad trees: A data structure for retrieval on composite keys. *JournalActa Informatica*, (1):1–9, 1974. ISSN 0001-5903.

[12] Antonin Guttman. R-trees: a dynamic index structure for spatial searching. *SIGMOD Rec.*, 14(2):47–57, 1984. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/971697.602266.

[13] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, pages 10 pp. vol.2–, Jan. 2000.

[14] S. Sitharama Iyengar and Richard R. Brooks. *Distributed Sensor Networks*. Chapman & Hall/CRC, USA, 2005.

[15] Juan Liu, James Reich, and Feng Zhao. Collaborative in-network processing for target tracking. *EURASIP Journal on Applied Signal Processing*, (4):378–391, 2003. doi: doi:10.1155/S111086570321204X.

[16] Olivier Michel. Webots: Professional mobile robot simulation. *Journal of Advanced Robotic Systems*, 1(1):39–42, 2004.

[17] Robert Nowak and Urbashi Mitra. Boundary estimation in sensor networks: Theory and methods. In *Information Processing in Sensor Networks (IPSN)*, pages 85–90, Palo Alto, CA, USA, April 2003. Springer-Verlag.

[18] Robert Nowak, Urbashi Mitra, and Rebecca Willett. Estimating inhomogeneous fields using wireless sensor networks. *IEEE Journal on Selected Areas in Communications*, 22(6):999–1006, August 2004.

[19] Amanda Prorok, C.M. Cianci, and Alcherio Martinoli. Towards optimally efficient field estimation with threshold-based pruning in real robotic sensor networks. *IEEE ICRA*, pages 5453–5459, 2010.

[20] Hanan Samet. The quadtree and related hierarchical data structures. *ACM Comput. Surv.*, 16(2):187–260, 1984. ISSN 0360-0300. doi: http://doi.acm.org/10.1145/356924.356930.

[21] Sensorscope Sàrl. `http://www.sensorscope.ch`.

[22] Ascending Technologies. `http://www.asctec.de`.

[23] Daniela Tulone and Samuel Madden. PAQ: Time series forecasting for approximate query answering in sensor networks. In *EWSN*, pages 21–37, 2006.

[24] Daniela Tulone and Samuel Madden. An energy-efficient querying framework in sensor networks for detecting node similarities. In *ACM MSWiM*, pages 191–300, 2006.

[25] Andras Varga. Software tools for networking: "OMNeT++". *IEEE Network Interactive (2002)*, 16(4), July 2002.

[26] Rebecca Willett, Aline Martin, and Robert Nowak. Backcasting: Adaptive sampling for sensor networks. In *Information Processing in Sensor Networks (IPSN)*, pages 124–133, Berkeley, CA, USA, April 2004. ACM Press.

[27] Ya Xu, John Heidemann, and Deborah Estrin. Geography-informed energy conservation for ad hoc routing. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 70–84, New York, NY, USA, 2001. ACM. ISBN 1-58113-422-3. doi: http://doi.acm.org/10.1145/381677.381685.

[28] Feng Zhao, Jie Liu, Juan Liu, L. Guibas, and J. Reich. Collaborative signal and information processing: an information-directed approach. *Proceedings of the IEEE*, 91(8):1199–1209, Aug. 2003. ISSN 0018-9219. doi: 10.1109/JPROC.2003.814921.