Evaluating Efficient Data Collection Algorithms for Environmental Sensor Networks

William Evans, Alexander Bahr and Alcherio Martinoli

Abstract Although there exists a large body of work on efficient data collection in sensor networks, the vast majority of proposed techniques have not been implemented on real networks or thoroughly studied on real data. As algorithm performance is highly dependent on the characteristics of the data being reported, it is very difficult to make suggestions as to the relative performance of any particular method. In this work we seek to compare and evaluate existing approaches to efficient data gathering in the specific context of environmental monitoring. We examine a choice algorithm that has not, to the best of our knowledge, been thoroughly studied on real data. We detail a number of algorithmic modifications necessary to bring it from theory to reality, and study the algorithm's performance in simulation using extensive traces from real world sensor network deployments.

1 Introduction

Low-cost sensor networks are becoming ubiquitous, as they have a broad range of applications from target tracking [2, 7] to health monitoring [10, 11], and our specific focus in this paper, environmental monitoring. This paper is part of an ongoing effort to deploy distributed intelligent algorithms into sensor networks consisting of resource-constrained nodes. The overall idea is that data collected by a network should be not only be used by the end user but should also at the same time allow more intelligent control of the activities of its nodes, possibly achieving the same

William Evans
Ecole Polytechnique Fédérale de Lausanne, Switzerland, e-mail: william.evans@epfl.ch

Alexander Bahr
Ecole Polytechnique Fédérale de Lausanne, Switzerland, e-mail: alexander.bahr@epfl.ch

Alcherio Martinoli
Ecole Polytechnique Fédérale de Lausanne, Switzerland, e-mail: alcherio.martinoli@epfl.ch

level of data accuracy with less power consumption. Ultimately, our goal is to validate and subsequently deploy such algorithms in real world scenarios.

We base our study on typical configurations used in environmental campaigns in our local Alpine region, where 10-20 sensor nodes are placed such that they are able to communicate via short range wireless transceivers. These networks always include at least one sink that uses long range communication (e.g., GPRS) to forward the data to a central location.

Currently there is a significant gap between theory and real world usage. While many algorithms for efficient data collection from sensor networks have been proposed, many existing systems simply take the naive ”always broadcast“ approach [4]. Indeed, the mention of efficient monitoring techniques in data-oriented sensor network deployments is rare, in general authors do not discuss the possibility of an intelligent data collection approach. The naive approach is proven and reliable, as uniform sampling with high relative frequency (e.g., one sample per minute) provides environmental engineers with easily manipulated data that have built-in redundancy due to the fact that many environmental fields change slowly the majority of the time (i.e. on the order of tens of minutes to days). However, there is much to gain by reducing energy consumption; doing so allows developers to lengthen sensor network autonomy, increase sampling frequency, lower reliance on expensive power sources, and so on.

This gap is in part to due to a lack of comprehensive analysis of existing algorithms. Authors tend to test their algorithms on small datasets that may not be representative for more general applications, and hardware implementations are rare. We seek to perform an exhaustive comparison of a large number of such algorithms as they apply to environmental monitoring in terms of vulnerability to node failure and message loss, communication overhead, and data accuracy.

(a) (b)

Fig. 1: SensorScope has seen several deployments under diverse conditions, including (a) a two-month deployment on Le Génépi comprised of 16 stations, and (b) a three-month deployment in the Wannengrat using 18 stations.

Fig. 2 Three weeks of sensor data were collected on the rooftop of EPFL's GR building.

2 Related Work

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 (i.e., it does not need to be transmitted). 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 [21]. It uses autoregressive models maintained locally per sensor 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) [20], 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. Prorok et al. study hierarchical network topologies based on spatial clustering [13]. In this approach, cluster heads may choose to prune their children if the part of the monitored field they represent is highly isotropic as defined by some statistically computed threshold. 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 [22].

Chu et al. propose the use of replicated dynamic probabilistic models between the sink and *disjoint cliques* of data sources [6]. 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.

A third and wholly separate approach, compressed sensing, draws on recent advances in signal processing that have interesting implications in sensor networks. This technique allows the accurate reconstruction of a signal while sampling at a rate that does not satisfy the Nyquist sampling theorem. Note that unlike other approaches, compressed sensing reduces the number of measurements needed, an obvious advantage when using active sensors, i.e., sensors that require considerable power just to sample. However compressed sensing also imposes strict requirements on the properties of measured data and makes in-network processing very difficult.

Haupt and Nowak develop the idea of compressed sensing in a multi-sensor scenario [8]. They envision using a uniform array of sensors to measure some phenomenon, each transmitting its processed results to a common destination, using inter-signal correlations to reduce power usage. Baron et al. generalize this idea, allowing the use of irregular spatial sampling and additionally taking advantage of sensors' recent history [3].

Outside of continuous monitoring, many approaches to efficient data collection seek to reduce overall message volume by eliminating uninteresting data in-network. TinyDB [9] provides such functionality, returning sensor data to the sink in response to simple aggregation queries such as SUM or MAX. Other techniques use in-network triggers to decide when data should be sent to the sink. Yang et al. present a Two-Phase Self-Join scheme that accepts complex monitoring queries from the user and informs the sink should an appropriate event be detected [23].

In [14], Sadagopan et al. compare their query resolution algorithm for sensor networks, ACQUIRE, with other more basic approaches. They conduct a theoretical analysis using mathematical models, using the results to tune algorithm parameters and draw performance estimates. While this approach allows theoretical insight into algorithmic performance, the authors make a number of assumptions that make their conclusions unlikely to extend to the uncontrolled conditions considered in this paper (e.g., uniform deployment and communication range). Our work seeks to compare algorithms experimentally, specifically evaluating algorithmic performance in a real-world context.

3 Materials and Methods

SensorScope stations [4] are a replacement for traditional large, centralized weather stations that may be time-consuming and expensive to deploy (see Figure 2). Instead they are a distributed array of smaller, cheaper stations that leverage greater coverage area and ease of deployment to provide more valuable data. SensorScopes guiding principle is to provide environmental scientists with real-time, total access to the data collected by their stations. This is best reflected by their online data browser, Climaps [17], which serves as an interface for both retrieving a particular

dataset (from a specific time period, collection of sensors and/or set of stations), as well as monitoring the battery levels and communication links of users' stations.

The stations themselves are built around 2-3m metal poles. Up to seven sensors may be attached along with the main controller, protected by a hermetically sealed container. Each station is attached to an energy source, in our case consisting of a large NiMH battery that is recharged by a small solar panel. Network sinks have an attached GPRS module for sending data from the network to off site SensorScope servers for viewing and archival. In this paper we use data collected by an attached SHT75 air humidity/temperature sensor and Zytemp TN901 infrared thermometer for sensing surface temperature.

The station itself is controlled by a ShockFish TinyNode 584 [18] running TinyOS 2.x. Local communication is performed on the 868MHz band using a Semtech XE1205 radio transceiver, with a range of one kilometer given strong line of sight.

SensorScope stations are an ideal platform for deploying efficient monitoring algorithms. The hardware platform has already proved itself successful in many previous deployments in various locales (see Figure 1), with many further deployments in planning by environmental scientists around Switzerland. Currently SensorScope stations take the naive approach to data collection, i.e., stations simply broadcast every sensor measurement made at regular time intervals. By implementing a data collection algorithm that takes advantage of the strong spatial and temporal correlations often present in environmental fields, we can reduce SensorScope stations' reliance on expensive power systems, driving down per-station cost.

SensorScope was specifically developed for the purpose of monitoring environmental phenomena. Environmental fields tend to lend themselves well to the suppression-based approaches discussed in the previous section. Strong temporal patterns are often present in environmental data, which may be described as having *trend* and *seasonal* components [21]. One clear example of a seasonal pattern is the typical day/night cycle, which has a clearly visible effect on ambient temperature as seen in Figure 3. Environmental data is also prone to high degrees of spatial correlation (see Figure 4); it is clear that spatio-temporal suppression is highly appropriate in this domain.

4 Constraint Chaining

In this section we explain *constraint chaining* (CONCH) [19], a technique that monitors constraints between adjacent nodes rather than the absolute values of nodes themselves. Like other recent work in efficient monitoring, CONCH uses a combination of spatial and temporal suppression to reduce network traffic. The in-network computational cost of this approach is minimal, making it a strong candidate for real world usage on the SensorScope platform. The algorithm provides real-time, accurate monitoring at a potentially greatly reduced communication cost while being flexible enough to tailor to specific data collection scenarios.

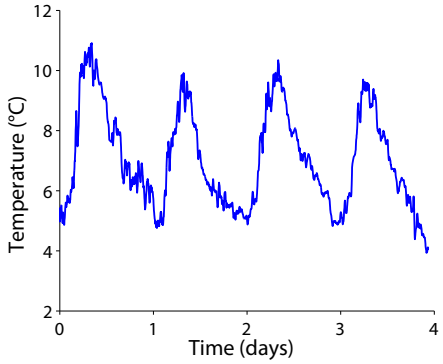

Fig. 3 Temperature data over a four-day period from a SensorScope deployment in the Génépi. This plot illustrates the seasonal component often present in environmental data.

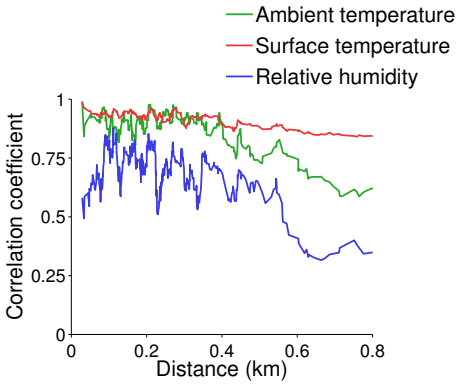

Fig. 4 Spatial correlation varies greatly depending on the type of environmental process in question. We calculate the correlation between each pair of sensor nodes within each deployment listed in Table 1 and plot these values directly according to distance.

4.1 Basic Algorithm

This approach revolves around a subset of the network's communication edges that we will refer to as the monitored edges. The nodes adjacent to each monitored edge are given different responsibilities for that edge. One node is assigned the role of *reporter*, responsible for sending a message to the sink every time the relative value of the two nodes along that edge changes. The opposite node is called the *updater*, and sends a message to the aforementioned reporter whenever its own value changes. Note that messages are not sent to the sink if both nodes change by the same amount simultaneously or if neither node changes at all. Nodes determined to be outliers, i.e. those that do not change in a pattern similar to any of their neighbors, are monitored directly and thus report their measured values to the sink whenever they change- we refer to these as monitored nodes.

A particular set of monitored edges, monitored nodes and updater/reporter assignments is called a *Conch plan* (see Figure 5 for an example). Ideally, we would like our plan to be configured such that the values along monitored edges change as infrequently as possible. Note that in order for a plan to be valid, each monitored edge must be connected to a tree that includes at least one directly monitored node

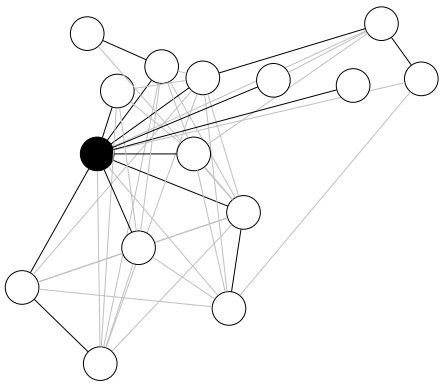

Fig. 5 Example CONCH plan from simulation over data from a real world SensorScope deployment in the Génépi. The sink is filled in black. Monitored edges are shown in black, and communication edges are shown in light gray.

(or the sink). In this way the sink is able to infer the values of all members of the network by chaining relative values along monitored edges, starting from this subset of nodes where absolute values are known.

Our search for a near-optimal CONCH plan is carried out in three steps. First, an arbitrary plan is chosen and used for a given period of time. In this phase network performance will be suboptimal, however note that as long as the plan is valid, no data will be lost. During this period we build up a data history that we will use to create a plan better suited to the observed environmental patterns. Once a hopefully representative dataset has been gathered (discussed further in Section 4.2), we assign a cost to all communication edges in the network. The cost of monitoring edge e is $freq(e) \times dist(e)$, where $freq(e)$ is the number of times the relative value along edge e changes throughout the dataset, and $dist(e)$ is the edge's distance from the sink. We also add an imaginary edge from every station s directly to the sink with cost $freq(s) \times dist(s)$. The minimum spanning tree over this graph represents the set of monitored edges and monitored nodes.

All that remains is our choice of updaters and reporters. The authors of this algorithm propose a mixed integer program that seeks to minimize network traffic given the data observed thus far. Their program accounts for the per-message and per-byte sending and receiving costs for the radios used as well as the number of messages required given a particular configuration. Again, the details of this formulation are left to [19].

4.2 Modifications

In this section we explain two algorithmic modifications necessary before CONCH can operate in the kind of network configuration described previously.

The first change deals with CONCH plan optimization. The linear program originally proposed contains in its objective function a number of variables that increases exponentially with the number of time steps and stations. As previously stated, it is

important to optimize over a length of time that is representative for the phenomenon being measured, e.g., one day/night cycle. In many deployments, SensorScope stations are configured to take a sensor reading once per minute. For a modestly sized network of 10 nodes over one day of measurements at the aforementioned rate, the objective function contains over 25,000 variables, with an even larger number of constraints. The result is an intractable linear program that is simply not solvable in a time frame appropriate for this problem.

Consider that only reporters are responsible for communicating data to the sink, and an optimal CONCH plan will be likely to place reporters closer to the sink than their corresponding updaters. Bearing this in mind, we simply iterate through all edges in the plan, marking the adjacent node closest to the sink as its reporter, and the other as its updater. This simplication may be cause excessive energy usage if the node chosen as updater changes value significantly more often than its reporter, however we do not observe such scenarios in our experiments.

Second, we introduce a strategy for repeatedly updating the CONCH plan. In evaluating their algorithm, the authors of CONCH build a CONCH plan using data collected over an initial training period. This plan is then used for the remainder of the test. In other words, they assume that relationships between nodes will never change. In a real environment it may be beneficial to explore more complicated techniques for building and maintaining an optimal CONCH plan; outdoor environments can be extremely dynamic, with unpredictable local and regional weather patterns playing a significant role on the spatial relationships present.

As creating and disseminating a new CONCH plan is a cheap operation compared to sensor reporting, we examine the performance of a scheme that builds an optimal plan using the previous N hours of network data, repeating this process after another N hours have passed.[1] One could imagine more dynamic schemes in which CONCH plan generation is triggered by some condition detected at the sink; however for the sake of brevity we leave such approaches to further study.

5 Simulation

5.1 Existing Datasets

We have collected a number of datasets from previous SensorScope deployments throughout Switzerland. We specifically selected deployments of about 10 or more nodes where the network was dense enough to be connected, yet sparse enough to require multi-hop communication. The results in this paper are found using datasets from three such deployments (see Table 1), downloaded with geographical metadata from Climaps [17], a data visualization and archiving system for environmental data.

[1] During the first N hours of operation we have no available data from which to guess stable relationships between nodes, so we simply use the routing tree as a temporary measure.

Table 1: Past SensorScope deployments provide over five years of individual station data appropriate for our algorithm evaluation framework.

Location	Nodes	Duration (days)
Génépi	15	65
Great St. Bernard Pass	16	43
GR Rooftop	8	17

It is the focus of this paper to examine algorithm performance via realistic simulation. Unfortunately, it is rare for data-oriented sensor network deployments to record the data necessary to accurately do so. In particular, we note a lack of useful topology and link quality information. We evaluated a number of approaches for generating simulated topology information, however we found such techniques to be inappropriate due to the nature of these deployments (i.e., in outdoor, uncontrolled environments, some times even including manual, undocumented redeployment over the experiment period). Indeed, as one may observe inFigure 5, wireless communication is highly unpredictable under our circumstances.

Incorporating datasets from new sources into our framework is a straightforward process. There are many publicly available environmental datasets that would be useful in our simulations, and we suggest where such datasets may be found in Section 6.

5.2 Results

We have developed a modeling framework sufficient for simulating a wide variety of algorithms over datasets from any external source. While we currently only make use of three datasets and three algorithms, we have established a common evaluation platform for performing deeper studies in the future. For further discussion, see Section 6.

All algorithms use error bounds according to individual sensor accuracy as listed on the relevant datasheet. Thus, for the Sensiron SHT75, we report ambient temperature to an accuracy of $\pm 0.3^{\circ}$C and relative humidity to $\pm 1.8\%$ [16]. We report surface temperature as measured by the Zytemp TN901 to an accuracy of $\pm 0.6^{\circ}$C [24]. We speak about algorithmic efficiency in terms of the overall reduction in transmitter power used, as calculated using the TinyNode datasheet [18]. Note that we do not talk about algorithmic overhead as it is negligible for all algorithms, especially when compared to the sampling frequency. However, it is accounted for in our results.

Our first significant result is that while CONCH may at first appear to yield significant power savings (see Table 2), in fact its built-in temporal sampling behavior is doing almost all of the work. It is likely that SensorScope stations simply sample so frequently that the probability of two nodes simultaneously perceiving a change in

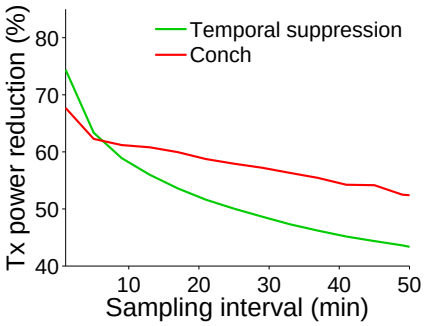

Fig. 6 As we decrease the sampling frequency, CONCH gains a greater advantage over temporal sampling. However, as subsequent measurements become less temporally correlated, the performance of both approaches drops significantly.

the environment is small. Indeed, as we increase the time between samples, CONCH gains a greater advantage over temporal sampling (see Figure 6).

The duration and frequency of CONCH plan optimization have a clear effect on the algorithm's ability to suppress messages. As previously described, we generate a new plan using the previous N hours of network data every N hours. In general, we observe that long training intervals are best suited to low sampling frequencies (see Figure 7). When sampling with high frequency, the unstable edge constraints make CONCH less performant than temporal suppression. However, as before, CONCH shows an increasing advantage over temporal suppression as we lower the sampling frequency.

6 Conclusion and Future Work

In this paper we presented our approach to evaluating data collection algorithms specific to environmental monitoring. We selected an algorithm likely suited to common patterns observed in environmental fields, CONCH, adapted it for deployment on real hardware, and studied its performance using simulations over a number of datasets collected from real world sensor network deployments. While temporal

Table 2: Algorithm performance by average power savings over all previously listed datasets. Here we set the training duration to $N = 4$ hours, and the sampling interval to 6 minutes. Temporal suppression yields tremendous savings (i.e., power consumed in transmitting sensor values is 43% of the original), while CONCH yields a small additional savings.

Algorithm	Tx power reduction	Standard deviation
Naive (SensorScope)	0.0%	0.0%
Temporal suppression	57.3%	6.8%
CONCH	62.2%	11.5%

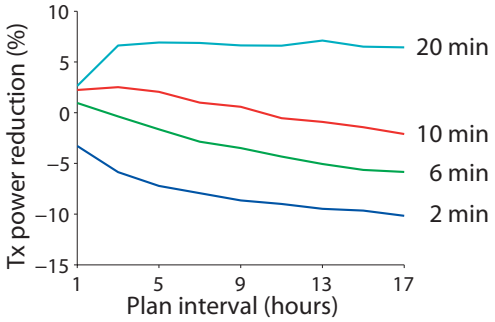

Fig. 7 CONCH's decrease in energy usage relative to temporal suppression as a function of plan duration (x-axis) and sampling interval (right). We show that a short training interval is more suited to high sampling frequencies, while longer plans only benefit very low sampling frequencies. Additionally, high sampling frequencies (2 and 6 minutes) fail to out perform temporal suppression, while lower sampling frequencies (10 and 20 minutes) see increased benefit from both CONCH and longer plan durations.

suppression yields large power savings (see Table 2), CONCH fails to bring a significant amount of additional performance at high sampling frequencies. However, CONCH was originally presented as simply a modeling framework. The models used in this paper could be replaced by something more complex that may even be specifically suited to different types of sensors. For example, in the case of CONCH, it may be beneficial for nodes to maintain edge constraints based on the parameters of the autoregressive models proposed in [21], rather than directly comparing sensor measurements. We plan to explore integrating such orthogonal approaches to efficient monitoring in the future.

Many research groups release data from internal sensor network deployments to the community. While we currently use datasets from long term SensorScope deployments, UCLA makes sensor network data available via SensorBase [5], PermaSense releases data to the public on their online repository [12], and often an individual researchers' datasets are made available upon request. We also plan to incorporate classic datasets such as indoor monitoring data from the Intel/Berkeley laboratory [15]. Such data can be used to obtain a more thorough understanding of algorithm performance.

Our laboratory has a number of SensorScope stations ready for use in implementing and testing algorithms under real-world conditions (one such deployment took place on the EPFL campus in Spring 2010, see Figure 2). We are currently implementing CONCH on real stations in order carry out further campaigns that investigate energy savings as a function of the spatial distribution of the stations and anisotropy of the monitored field.

7 Acknowledgements

This work was partially funded by "The Swiss Experiment" of the Competence Center Environment and Sustainability of the ETH Domain (CCES), and by the Swiss National Science Foundation's Stabilization Measures Program, associated with the project "Tamperproof Monitoring Solution for Weather Risk Management" managed by the National Center of Competence in Research in Mobile Information and Communication Systems (NCCR-MICS).

References

1. Arici, T., Altunbasak, Y.: Adaptive sensing for environment monitoring using wireless sensor networks. In: IEEE Wireless Communications & Networking Conf., pp. 2347–2352. Atlanta, GA, USA (2004)
2. Arora, A., Dutta, P., Bapat, S., Kulathumani, V., Zhang, H., Naik, V., Mittal, V., Cao, H., Demirbas, M., Gouda, M.: A line in the sand: A wireless sensor network for target detection, classification, and tracking. Computer Networks **46**(5), 605–634 (2004)
3. Baron, D., Wakin, M., Duarte, M., Sarvotham, S., Baraniuk, R.: Distributed compressed sensing. Preprint (2005)
4. Barrenetxea, G., Ingelrest, F., Schaefer, G., Vetterli, M.: The hitchhiker's guide to successful wireless sensor network deployments. In: ACM Conf. on Embedded Networked Sensor Systems, pp. 43–56. Raleigh, NC, USA (2008)
5. Center for Embedded Networked Sensing: SensorBase data repository. URL http://www.sensorbase.org/
6. Chu, D., Deshpande, A., Hellerstein, J., Hong, W.: Approximate data collection in sensor networks using probabilistic models. In: Int. Conf. on Data Engineering, pp. 48–48. Atlanta, GA, USA (2006)
7. Gui, C., Mohapatra, P.: Power conservation and quality of surveillance in target tracking sensor networks. In: ACM Int. Conf. on Mobile Computing and Networking, pp. 129–143. Philadelphia, PA, USA (2004)
8. Haupt, J., Nowak, R.: Signal reconstruction from noisy random projections. IEEE Transactions on Information Theory **52**(9), 4036–4048 (2006)
9. Madden, S., Franklin, M., Hellerstein, J.: TinyDB: an acquisitional query processing system for sensor networks. ACM Transactions on Database Systems **30**(1), 122–173 (2005)
10. Milenkovic, A., Otto, C., Jovanov, E.: Wireless sensor networks for personal health monitoring: Issues and an implementation. Computer Communications **29**(1314), 2521–2533 (2006)
11. Otto, C., Milenkovic, A., Sanders, C., Jovanov, E.: System architecture of a wireless body area sensor network for ubiquitous health monitoring. Journal of Mobile Multimedia **1**(4), 307–326 (2006)
12. PermaSense: PermaSense data frontend. URL http://data.permasense.ch/
13. Prorok, A., Cianci, C., Martinoli, A.: Towards optimally efficient field estimation with threshold-based pruning in real robotic sensor networks. In: IEEE Int. Conf. on Robotics and Automation, pp. 5453–5459. Anchorage, AK, USA (2010)
14. Sadagopan, N., Krishnamachari, B., Helmy, A.: Active query forwarding in sensor networks. Ad Hoc Networks **3**(1), 91–113 (2005)
15. Samuel Madden: Intel Berkeley Lab data. URL http://db.csail.mit.edu/labdata/labdata.html
16. Sensirion: Sht75 digital humidity sensor data sheet (2010) URL http://www.sensirion.com/
17. Sensorscope Sàrl: Climaps weather monitoring system. URL http://sensorscope.epfl.ch/climaps/
18. ShockFish: TinyNode 584 user's manual (2010) URL http://tinynode.com/

19. Silberstein, A., Braynard, R., Yang, J.: Constraint chaining: On energy-efficient continuous monitoring in sensor networks. In: ACM SIGMOD Int. Conf. on Management of Data, pp. 157–168. Chicago, IL, USA (2006)
20. Tulone, D., Madden, S.: An energy-efficient querying framework in sensor networks for detecting node similarities. In: ACM Int. Symp. on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pp. 191–300. Torremolinos, Malaga, Spain (2006)
21. Tulone, D., Madden, S.: PAQ: Time series forecasting for approximate query answering in sensor networks. In: European Conf. on Wireless Sensor Networks, pp. 21–37. Zurich, Switzerland (2006)
22. Willett, R., Martin, A., Nowak, R.: Backcasting: adaptive sampling for sensor networks. In: ACM Information Processing in Sensor Networks, pp. 124–133. Berkeley, CA, USA (2004)
23. Yang, X., Lim, H., Özsu, T., Tan, K.: In-network execution of monitoring queries in sensor networks. In: ACM SIGMOD Int. Conf. on Management of Data, pp. 95–105. Beijing, China (2007)
24. Zytemp: Tn9 infrared thermometer user manual (2010) URL http://www.zytemp.com/