What Does Model-Driven Data Acquisition *Really* Achieve in Wireless Sensor Networks?

Usman Raza*[†], Alessandro Camerra[†], Amy L. Murphy*, Themis Palpanas[†], Gian Pietro Picco[†]
* *Center for Scientific and Technological Research, Bruno Kessler Foundation,Trento, Italy*
{raza,murphy}@fbk.eu
[†]*Department of Information Engineering and Computer Science (DISI), University of Trento, Italy*
{themis.palpanas,gianpietro.picco}@unitn.it

***Abstract*—Model-driven data acquisition techniques aim at reducing the amount of data reported, and therefore the energy consumed, in wireless sensor networks (WSNs). At each node, a model predicts the sampled data; when the latter deviate from the current model, a new model is generated and sent to the data sink. However, experiences in real-world deployments have not been reported in the literature. Evaluation typically focuses solely on the quantity of data reports suppressed at source nodes: the interplay between data modeling and the underlying network protocols is not analyzed.**

In contrast, this paper investigates *in practice* whether *i)* model-driven data acquisition works in a real application; *ii)* the energy savings it enables in theory are still worthwhile once the network stack is taken into account. We do so in the concrete setting of a WSN-based system for adaptive lighting in road tunnels. Our novel modeling technique, Derivative-Based Prediction (DBP), suppresses up to 99% of the data reports, while meeting the error tolerance of our application. DBP is considerably simpler than competing techniques, yet performs better in our real setting. Experiments in both an indoor testbed and an operational road tunnel show also that, once the network stack is taken into consideration, DBP *triples* the WSN lifetime—a remarkable result per se, but a far cry from the aforementioned 99% data suppression. This suggests that, to fully exploit the energy savings enabled by data modeling techniques, a coordinated operation of the data and network layers is necessary.

I. INTRODUCTION

Wireless sensor networks (WSNs) provide the flexibility of untethered sensing, but pose the challenge of achieving extended lifetime with a limited energy budget, often provided by batteries. In this respect, it is well-known that communication causes the biggest energy drain. This is unfortunate, given that the ability to report sensed data is the one motivating the use of WSNs in several pervasive computing applications.

An approach to reduce communication without compromising data quality is to *predict* the trend followed by the data being sensed. This technique is referred to as *model-driven data acquisition* and is applicable when data is reported periodically—the common case in many pervasive computing applications. In these cases, a model of the data trend can be computed locally to a node, and constitutes the information being reported to the data collection sink, in place of several raw samples. As long as the locally-sensed data are compatible with the model prediction, no further communication is needed: only when the sensed data deviates from the model, must the latter be updated and sent to the sink.

The aforementioned approach is well-known, and adopted by several works we concisely survey in Section V. Nevertheless, to the best of our knowledge none of these works has been applied in a real-world pervasive application. Therefore, their practical applicability remains unascertained. Moreover, these works typically evaluate the gains only in terms of messages suppressed w.r.t. a standard approach sending all samples. This data-centric view, however, is quite optimistic. WSN network protocols consume energy not only when transmitting and receiving data, but also in several *continuous* control operations, e.g., when maintaining a routing tree for data collection, or probing for ongoing communication at the MAC layer. Therefore, the true question, currently unanswered by the literature, is to what extent the theoretical savings enabled by model-driven data acquisition are actually observable *in practice* when the application and network stack are combined.

Hence, in contrast with the existing literature, our goal is:

- to investigate the benefits of model-driven data acquisition in an existing deployment [1] providing closed-loop adaptive lighting in an *operational* road tunnel. As described in Section II, the WSN is used to periodically report light samples, and is therefore representative of several pervasive computing applications, e.g., smart environments, building management, home automation [2];
- to assess the interplay of data modeling and the underlying network protocols, by evaluating qualitatively and quantitatively the relationship between the two.

We achieve these goals with the following contributions:

- we propose a novel method, called *Derivative-Based Prediction* (DBP) for locally predicting the trend of data sensed by a WSN node. DBP, described in Section III, is considerably simpler than existing methods—a plus on resource-scarce WSN platforms. Nevertheless, our evaluation based on real-world data

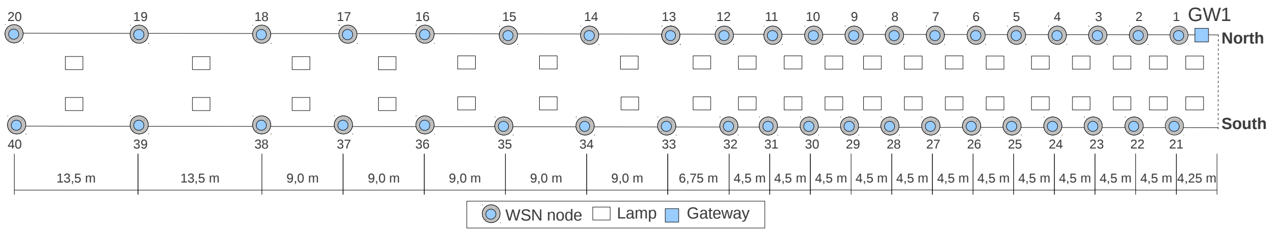


Figure 1. Physical placement of WSN nodes in the tunnel.

from the tunnel deployment shows that DBP performs comparably to existing techniques. As shown in Section IV, DBP suppresses up to 99% of the raw reports in our application, while maintaining its data quality within the required error tolerance.

- we analyze to what extent this staggering improvement is affected by the interaction with network protocols, by running our application on top of popular WSN protocols (i.e., CTP [3] and Box-MAC [4]). Moreover, we feed the application the same light data "replayed" from the tunnel deployment, to directly compare the theoretical gains against the practical ones. We do so in two settings: an operational tunnel, representative of our target application, and a 40-node indoor testbed, representative of alternate application scenarios. Our results in Section IV confirm the expectation that the gains attained in practice when considering the network stack are dramatically lower than those derived in theory by taking into account only the application messages. However, the improvements are still remarkable in absolute terms: DBP *triples* the WSN lifetime w.r.t. a standard solution with periodic reporting.

Our results confirm that model-driven data acquisition can yield substantial lifetime improvements in practical settings. However, as we point out in the final remarks of Section VI, the results also suggest that, to fully exploit the energy savings made possible by model-driven data acquisition, coordination between the data and network layers is necessary.

II. WSN-based Adaptive Lighting in Road Tunnels

Our application case study is a WSN deployed in a road tunnel to acquire light readings [1]. These are relayed in multi-hop to a gateway, and from there to a Programmable Logic Controller (PLC) that closes the control loop by setting the intensity of the lamps inside the tunnel. In contrast with the state of the art in tunnels, where light intensity is pre-set based on the current date and time, or at best determined by the external conditions, this closed-loop adaptive lighting system maintains optimal light levels by considering the *actual* conditions inside the tunnel. This increases safety, and enables considerable energy savings.

WSNs are an asset in this scenario, as the nodes can be placed at arbitrary points along the tunnel, not only where power and networking cables can reach. This drastically reduces installation and maintenance costs, and makes WSNs particularly appealing for already existing tunnels, where changes to the infrastructure should be minimized. The downside to such flexibility is the reliance on an autonomous energy source. Nevertheless, battery costs are minimal and the replacement process can be easily combined with regularly-planned tunnel maintenance.

Figure 1 shows the placement of WSN nodes inside our 260 m-long, two-way, two-lane tunnel. Overall, 40 nodes are split evenly between the tunnel walls and placed at a height of 1.70 m, compatible with legal regulations. Their data reports are collected by a gateway, installed 2 m from the entrance. Each node is functionally equivalent to a TelosB mote [5], augmented with a sensor board equipped with 4 ISL29004 digital light (illuminance) sensors. The light readings, collected at a sampling rate of 5 s, are locally aggregated and filtered. Every 30 s, the result of this aggregation is reported to the sink. The WSN nodes are not time synchronized: a node reports its light value whenever its 30 s timer expires.

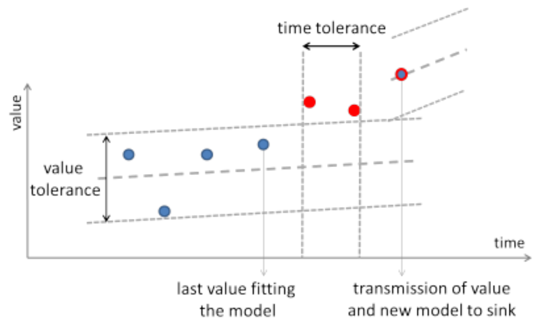
This setup is similar to the one reported in [1], where we detail and evaluate the operational WSN-based, closed-loop adaptive lighting system. In this paper we use a different application and network stack, and compare our model-driven data acquisition technique against the baseline constituted by the aforementioned periodic reporting of all raw light samples.

III. Data Modeling of Time Series with DBP

We define the data modeling problem we address in this paper, and illustrate our novel DBP technique.

A. Problem Formulation

The application we described in Section II is an instance of a general class of WSN applications where nodes periodically take sensor measurements and report the corresponding samples to a data sink. Moreover, we make the additional assumption that the application running at the sink allows for a small tolerance in the accuracy of the reported data. In contrast with the ideal requirements of the sink obtaining *exact* values in *all* data reports, the correctness of these applications is unaffected as long as *i)* the reported values match *closely* the exact ones; *ii)* inaccurate values occur only *occasionally*. In other words, deviations from the exact


time tolerance
value
value tolerance
time
last value fitting the model
transmission of value and new model to sink
Figure 2. Value and time tolerance.

reports are acceptable, as long as their extent in terms of difference in *value* and *time interval* during which the deviation occurs are small enough.

We capture these assumptions, common to many applications, with the following definitions:

- Let V_i be an exact measurement taken at time t_i. The *value tolerance* is defined by the maximum relative and absolute errors acceptable, $\varepsilon_V = (\epsilon^{rel}, \epsilon^{abs})$. From the application perspective, reading a value V_i becomes equivalent to reading any value $\hat{V}_i$ in the range R_V defined by the maximum error, $\hat{V}_i \in R_V = [V_i - \epsilon, V_i + \epsilon]$, where $\epsilon = \max\{\frac{V_i}{100}\epsilon^{rel}, \epsilon^{abs}\}$. In other words, the application considers a value $\hat{V}_i \in R_V$ as *correct*.
- Let $T = |t_j - t_k|$ be a time interval, and $\hat{V}_T = \{\hat{V}_j, \ldots, \hat{V}_k\}$ the set of values reported to the application during T. The *time tolerance* ε_T is the maximum acceptable value of T such that *all* the values reported in this interval are incorrect, i.e., $\hat{V}_i \notin R_V$, $\forall \hat{V}_i \in \hat{V}_T$.

The intuition behind these definitions is shown in Figure 2.

Similarly to other model-driven data acquisition techniques, DBP aims at suppressing as many data reports from the WSN nodes as possible, while ensuring that the data used by the application at the sink is within the value and time tolerances ε_V and ε_T specified as part of the requirements.

The combined use of absolute and relative errors in the value tolerance is worth commenting further, in the context of our application. When light levels are low, e.g., at night, even small absolute variations are large in terms of percentage. With only an absolute error, these minimally-perceivable changes would trigger model changes. Instead, by considering the maximum between the relative and absolute error, our control algorithm is able to both adjust to the meaningful changes and avoid unnecessary communication.

B. Derivative-based Prediction

DBP is based on the observation that, in our application, the trends of the sensed values in short and medium time intervals can be accurately approximated using a linear model. Even though this idea has appeared in previous works, there is a key difference to our approach: while previous studies compute models that aim to reduce the approximation error to the *data points* in the recent past, DBP aims at producing models that are consistent with the *trends* in the recently-observed data.

Figure 3 provides an illustration of DBP. Initialization consists of a *learning phase*, gathering enough data to produce the first model. The learning phase involves m data points; the first and the last l we call *edge points*. The model is linear and is computed as the slope δ of the segment that connects the average values over the l edge points at the beginning and end of the learning phase. This computation resembles the calculation of the derivative, hence the name *Derivative-Based Prediction*. It is interesting to note that the computation of this prediction is not only very simple, and therefore appealing for implementation on resource-scarce nodes, it also mitigates the problem of noise and outliers.

The first DBP model generated is then sent to the sink, along with its last data point. From that point on, each node buffers a sliding window of the last m data points sampled from its sensor. Upon sampling a point, the "true" value sensed is compared to the "predicted" one computed by DBP according to the current model, i.e., following the slope δ. If the sensor reading is within a value tolerance ε_V w.r.t. the model, no action is required: the sink will automatically generate a new value that is an acceptable approximation of the real one. Otherwise, if the readings continuously deviate from the model for more than ε_T time units, a new model must be recomputed. This is accomplished by using the last m data points in the buffer; the resulting model is transmitted to the sink along with the last data point.

IV. Experimental Evaluation

We focus on our tunnel lighting application, and therefore use raw data from the 40-node deployment described in Section II. Light readings were reported every 30 s from each node for 47 days, for a total of $5,414,400$ measurements. This tunnel offers a challenging scenario as its entrance is subject to direct sunlight, creating wide variations in the sensor readings. We also applied DBP to data collected by 40 nodes over 90 days in a second tunnel subject to less radical light variations, yielding similar results. Due to space constraints we present only results from the first tunnel.

To establish the proper value and time tolerances, we consulted the lighting engineers who designed the control algorithm that establishes the lamp levels. Notably, lamp

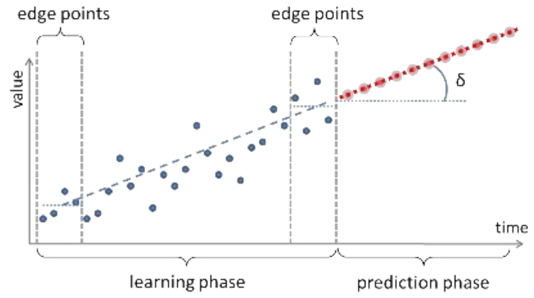


Figure 3. Derivative-based Prediction.

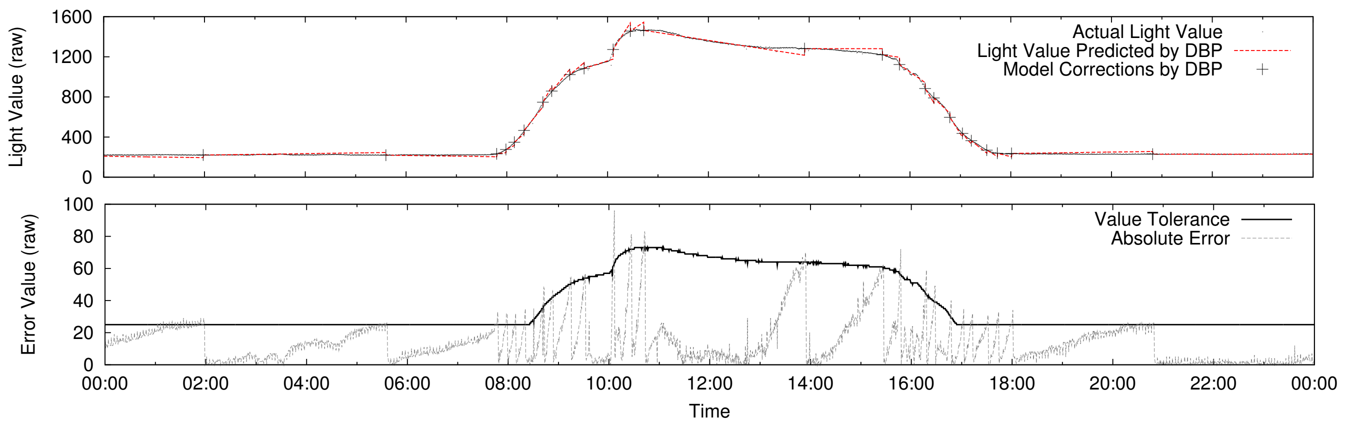


Figure 4. Absolute light values (top) and error (bottom), reported as raw sensor values.

levels are adjusted slowly to minimize the effects of changes on drivers, therefore tight real-time deadlines are not necessary. By taking into consideration also the inherent error of illuminance sensors, we determine a value tolerance $\varepsilon_V = (5, 25)$, i.e., values generated by the model can differ by at most 5% or 25 w.r.t. the raw sensor reading. Further, we identified a time tolerance of one minute. For convenience, we express ε_T in terms of the 30 s reporting intervals of the application; a one-minute time tolerance corresponds to $\varepsilon_T = 2$. We further fix the core parameters of DBP, namely the number of values in the learning phase $m = 20$, and the size of the edge point sets $l = 3$. We verified that these values yield the best performance, and that their impact is nonetheless rather limited.

We approach the experimental evaluation from two angles. First, we evaluate DBP's ability to reduce the amount of data generated at the nodes, and compare it against other existing methods. Then, we consider the combination of DBP and a mainstream WSN network stack, evaluating the benefits both in an indoor testbed and an operational tunnel.

A. *Performance of Model-driven Data Acquisition*

Our main performance metric is the *transmission ratio*, $TR = \frac{\text{\# messages generated with DBP}}{\text{\# messages generated without DBP}}$. Although the *suppression ratio* $SR = 1 - TR$ directly measures the number of messages whose reporting can be avoided thanks to DBP, we show our results in terms of TR as it is easier to relate the transmission of messages (instead of their absence) to the light patterns.

Our evaluation is divided in three parts. First, we gain a deeper understanding of DBP in the context of our application by considering the operation of a single node over a single day. Second, we report DBP performance across our entire 47-day data set, and analyze the parameter space, investigating the impact of various settings on the suppression rate. Finally, we compare to the state of the art.

1) DBP *in Action:* Our first goal is to understand whether DBP satisfies the error tolerance requirements of our application and, in this context, understand its operation. For this, we set $\varepsilon_V = (5, 25)$ and $\varepsilon_T = 2$, as discussed earlier.

We begin by analyzing DBP in the small, dissecting the operation of a single node over a single day of operation. We choose node 1 because, as shown in Figure 1, it is placed at the tunnel entrance, and subject to radical changes in its light readings. The top of Figure 4 shows the light values for this node both in the original case where data is reported every 30 s and when DBP is applied. In the latter case, the cross points indicate the generation of a new model, while the lines between the points show the light values calculated from those models. The two curves are very similar, and yet a significant reduction in messages is achieved. Notably, without DBP, 2, 880 messages are sent, instead, with DBP, only 25 messages are sent: a suppression ratio of 99.1%.

As expected, the majority of the DBP models are generated in the time intervals where light trends change, namely sunrise and sunset. The rest of the time, DBP generates very few models. Interestingly, even though the light at night is quite constant, a few models are generated in order to correct for models that have a non-zero derivative.

Node 1 is placed at the tunnel entrance, where the widest excursion in light values are expected. Nevertheless, improvements are achieved throughout the tunnel. To measure this, we divide our 24-hour experiment into 5-minute intervals and count the number of models generated by all nodes in each interval. We report the totals in Figure 5. As one expects, the majority of changes are concentrated during daylight hours, in particular around sunrise and sunset.

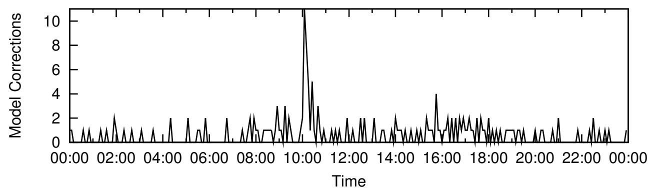


Figure 5. Total number of DBP model corrections over 5-minute intervals, during a 24-hour experiment. $\varepsilon_V = (5, 25)$ and $\varepsilon_T = 2$.

Nevertheless the total number of models in any 5-minute interval was below 10 and usually below 4. At night, there are even fewer updates, with many intervals in which no models are generated.

Finally, in the bottom of Figure 4 we focus again on the entrance node 1 as a representative example, to analyze the error in the values provided by DBP to the application during each 30 s reporting interval. The solid line indicates the value tolerance $\varepsilon_V = (5, 25)$ set by our application requirements, while the lighter line shows the error of DBP as the difference between the predicted value and the sensed raw value. In most cases, the error falls below the value tolerance. Excursions above the value tolerance are caused by data predicted at the sink that, albeit incorrect, are within the time tolerance. In each of these cases, either subsequent values fell back below value tolerance or a new model was generated after the maximum number of incorrect reports ($\varepsilon_T = 2$ in our case) was exceeded. Interestingly, at night, one can see the absolute error growing for a while, then dropping and growing again. The drop in error corresponds to the generation of a new model, visible also in the top of Figure 4. The growing error is because the DBP model is linear with a small, but non-zero slope, which is slightly off the measured light values that remain mostly constant.

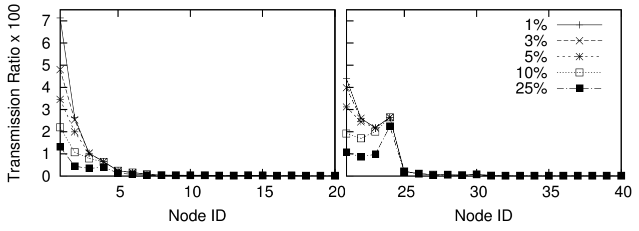
2) Impact of Error Tolerance: The previous evaluation shows that DBP performs well for the requirements of the tunnel application. However, we want to explore the parameter space for DBP, to understand the effect of changes in the value and time tolerances on the transmission ratio. Figures 6(a)–6(c) show how TR changes at individual nodes for various combinations of parameters. Recall from Figure 1 that nodes 1–20 are placed on the same North wall, while nodes 21–40 belong to the South wall. We plot a line connecting the TR at each node, because this best highlights the trends as one proceeds from the entrance to the interior of the tunnel (e.g., from node 1 to 20 on the North wall).

In Figure 6(a) we vary the relative error ϵ^{rel} from 1% to 25%, keeping the absolute error constant $\epsilon^{abs} = 25$. By setting the time tolerance to $\varepsilon_T = 0$, we force all deviations from the value tolerances to be reported. To put these values in context, recall that the value tolerance ε_V is defined as the maximum between the relative and absolute errors, ϵ^{rel} and ϵ^{abs}. In Figure 6(b) we fix $\epsilon^{rel} = 5\%$ and vary ϵ^{abs} between 0 and 50, keeping $\varepsilon_T = 0$. In Figure 6(c), we use the value tolerance $\varepsilon_V = (5, 25)$ of our target application and vary ε_T between 0 and 4, i.e., from 0 to 2 minutes.

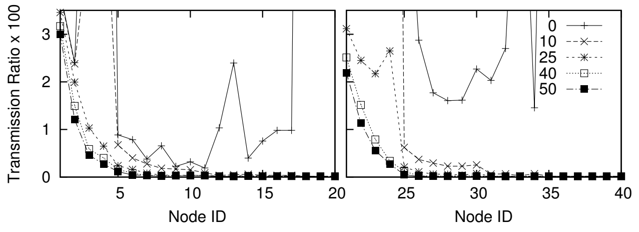
In all cases it is worth noting that, as expected, the biggest savings are harvested from the nodes inside the tunnel, where light variations are rarer, and absolute values of illuminance are smaller. Under these conditions, the linear nature of DBP accurately models the linear nature of the data.

Interestingly, the trends seen for nodes 21-24 in Figure 6(a) are due to the flickering of a light that introduced noise to the sensor readings. Nevertheless, even in this case DBP achieved suppression ratios greater than 95% for these nodes. Further, in Figure 6(b), we clearly see the need for both the absolute and relative value tolerances, as when the error tolerances are very low, e.g., $\epsilon^{abs} = 0$ or $\epsilon^{abs} = 10$, TR is off the charts. This is because the light sensors themselves have an error that often takes them outside the small, fixed relative error $\epsilon^{rel} = 5\%$, triggering unpredictable model changes. Further, the flickering light introduces additional noise that DBP cannot compensate for with low error thresholds.

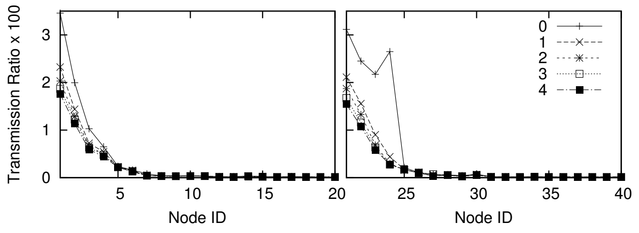
For each of these parameter combinations we also show, in Figure 6(d), the average TR over all nodes. An increase in the value of ϵ^{rel} brings a near linear reduction of TR. Instead, ϵ^{abs} and ε_T both achieve the greatest benefit at small values, with diminishing returns as the value increases. In the former case, the reduction in TR progresses rapidly as ϵ^{abs} varies from 0 to 10, going from a suppression ratio of 88% to 98%; a further (and larger) ϵ^{abs} increase from 10 to 25 yields only an additional 2% reduction of TR. Similarly, time tolerance reflects the fact that changes in light values are gradual, and thus introducing even a small delay $\varepsilon_T = 1$



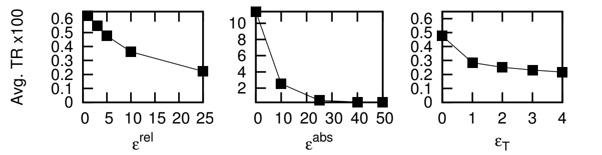
(a) Relative error ϵ^{rel} ($\epsilon^{abs} = 25$, $\varepsilon_T = 0$).



(b) Absolute error ϵ^{abs} ($\epsilon^{rel} = 5\%$, $\varepsilon_T = 0$).



(c) Time tolerance ε_T ($\epsilon^{rel} = 5\%$, $\epsilon^{abs} = 25$).



(d) Average transmission ratio for the combinations above.

Figure 6. Impact of error tolerance parameters on transmission ratio.

achieves most of the possible gain.

In addition to the combinations in Figure 6, we also computed the TR achieved with the *strictest* combination of the three parameters: $\epsilon^{rel} = 1\%$, $\epsilon^{abs} = 0$, and $\varepsilon_T = 0$. Even with these worst-case requirements DBP still suppresses, on average, 63% of the reports. More interesting is the *real* combination of parameters ($\epsilon^{rel} = 5\%$, $\epsilon^{abs} = 25$, and $\varepsilon_T = 2$) suggested by the tunnel engineers, and used in the rest of our experiments. In this case, the average suppression rate is a staggering 99.7%—TR is reduced by almost two orders of magnitude w.r.t. reporting all raw values. The individual TR achieved at each node is shown in Figure 7, where we compare DBP against state-of-the-art techniques, as discussed next.

3) Comparison to Other Approaches: We compared DBP against the following techniques, which in Section V are also put in the wider context of related work:

- *Piecewise Linear Approximation* (PLA) is a popular technique that uses least square error linear segments to approximate a set of values [6]. In our case, each node uses a single segment to model for sensed values.
- *Similarity-based Adaptable Framework* (SAF) [7] relies on an autoregressive moving-average model of order 3 with moving-average parameter of order 0. In SAF a value V_i is predicted by a linear combination of the last three: $V_i = l_i + \alpha_1(V_{i-1} - l_{i-1}) + \alpha_2(V_{i-2} - l_{i-2}) + \alpha_3(V_{i-3} - l_{i-3})$, where α_1, α_2, α_3 are constants the model must estimate, and l_i models the linear trend of data over time.
- As an additional point of comparison, we implemented a *Polynomial Regression* (POR) method. In contrast to DBP, POR allows the use of non-linear models for prediction. Intuitively, this may yield better performance through a better fit to the data. Like PLA, POR uses the least squares measure for selecting the most appropriate coefficients for the polynomials, which have the form $y = \sum_{k=0}^{p} \alpha_i x^i$. In this study, we evaluated polynomials of order $p = 2$, 3, 4. We used $p = 2$ as it provides the best results for POR.

We used $\varepsilon_V = (5, 25)$ and $\varepsilon_T = 2$, the requirements of our target tunnel application. Table 7(a) shows the results w.r.t. the error in predicting the actual sensor readings. The values shown are the average error per point, over the entire 47-day dataset and over all nodes, computed as the Euclidean distance between the real sensed value and the value predicted by the corresponding model. We note that PLA achieves a lower error than DBP. This is because DBP inherently permits some amount of error in the model, while PLA employs an objective function that explicitly chooses the model that minimizes the error. However, as we present next, DBP achieves a higher reduction in TR, because it better models the data trends.

In terms of communication performance, all approaches perform quite well, however, DBP achieves the best results, as shown in Table 7(b) and Figure 7(c). As already mentioned, DBP suppresses 99.7% of the message reports with our tunnel application requirements.

Because all approaches achieve very good results, it is worth noting that finding the derivative of the sensed data, at the core of DBP, is significantly less complex than solving linear equations with 2 or 3 unknowns, as required by PLA, POR and SAF. Notably, in our DBP implementation, used in the in-network evaluation explained in the following section, the core module to calculate the derivative contains only 25 lines of TinyOS code, notably with no floating point arithmetic. As node memory is limited, eliminating the floating point arithmetic module is greatly desirable.

Finally, in the course of our investigation, we stressed DBP by artificially modifying the data set, specifically introducing a significant amount of noise while maintaining the trends. Notably, DBP was still able to achieve the best suppression ratios, even though the error of DBP was the largest among the alternate approaches. This is due to the fact that the other approaches are designed to operate on relatively smooth data, while DBP focuses on accurately predicting trends. The ability of DBP to achieve significant gains in the presence of noisy data can lead to a significant advantage with other data sets. For example in the same tunnel project, we collected carbon monoxide (CO) data, and although the sampling period was lower (5 minutes instead of 30 seconds), the noise in this data is significantly greater. Nevertheless, linear trends are present, as CO increases during the day when traffic is higher and decreases at night. Therefore we expect DBP to perform quite well, significantly reducing the generated data.

B. Impact of the Network Stack

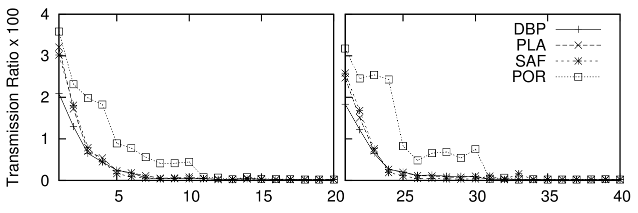
We study the performance of DBP in conjunction with the commonly-used network stack composed of CTP [3], BoX-MAC [4], and TinyOS v2.1.1. We experiment in two settings: an operational road tunnel to evaluate DBP in the real conditions of our target application, and an indoor testbed, representative of scenarios with different connectivity.

DBP	0.00830
PLA	0.00817
SAF	0.00907
POR	0.01900

(a) Average error.

DBP	0.00259
PLA	0.00328
SAF	0.00312
POR	0.00712

(b) Average TR.



(c) Transmission ratio (TR) at individual nodes.

Figure 7. Comparing DBP to alternative approaches.

Tunnels are complex environments where factors such as road traffic affect network behavior. For example, we previously observed [8] that in the presence of high traffic, nodes consistently select parents on their same side of the tunnel, while at low traffic nodes across the tunnel are often selected. This is due to the interference caused by vehicles, nevertheless, it profoundly affects the shape and maintenance cost of the routing tree. For these experiments, we relied on the 40-node WSN in Figure 1. The testbed is composed of 40 TelosB nodes in a 60x40 m^2 office area shown in Figure IV-B. The node placement, along with the power setting of -1 dBm, creates a network topology that approximately forms three segments, loosely reminiscent of the linear tunnel topology, but with larger diameter.

To assess directly the impact of the network stack on the improvements theoretically attainable by DBP, we "re-played" the same data we used in Section IV-A both in the tunnel and testbed. As we could not re-execute the entire 47-day dataset with multiple combinations of parameters, for the tunnel we selected a single 23-hour period, ensuring variability in the vehicular traffic. Moreover, restrictions on the usage of the testbed forced us to run only 2-hour experiments. Therefore, in this latter case we chose to focus on the sunrise period, the most challenging because values change dramatically and, unlike sunset, are not followed by the night constant light levels. Figure 9 shows the number of models generated by each node in both cases. We begin the evaluation after DBP has been initialized, specifically after generation and transmission of the first model.

We now study how data delivery to the application, network lifetime, and routing costs are affected by DBP. All of these aspects, and particularly the first two, are deeply affected by the operation of the MAC layer, specifically the rate at which the radio duty cycles, which therefore becomes a key parameter in our experiments. At low sleep intervals, nodes frequently check the channel but find no activity, increasing idle listening costs. At large sleep intervals, the cost to transmit a packet increases. In BoX-MAC, transmission to a non-sink node takes on average half the sleep interval, due to the fact that the sender must transmit until the receiver wakes up, receives the packet, then acknowledges its reception [4]. This long transmission interval also increases the probability of packet collisions among hidden terminals, further decreasing the delivery ratio and increasing energy consumption. The ideal sleep interval balances idle listening and active transmission costs. To identify the best interval for our application, we ran experiments with a range of values from 500 to 3000 ms.

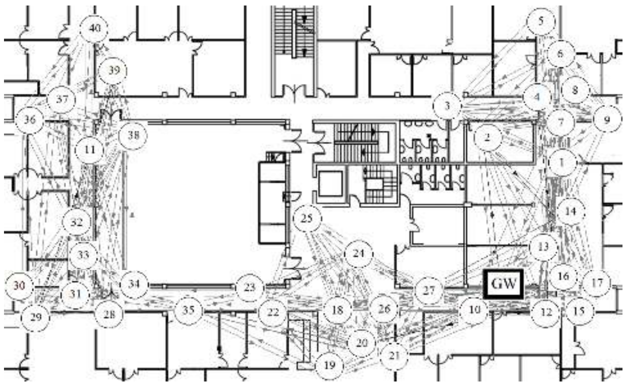
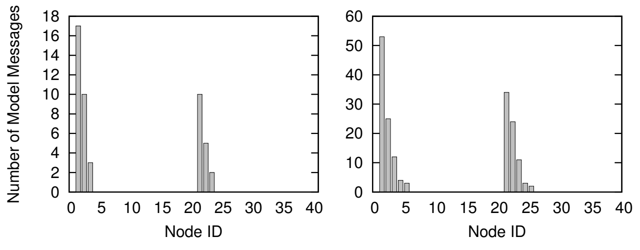


Figure 8. Testbed map and connectivity.

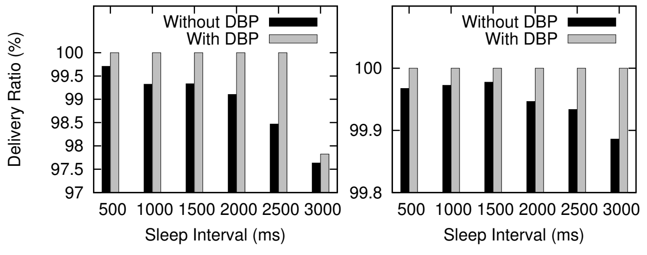


(a) Testbed: $TR = 0.0048$, 2 hours. (b) Tunnel: $TR = 0.0014$, 23 hours.

Figure 9. Number of model update messages.

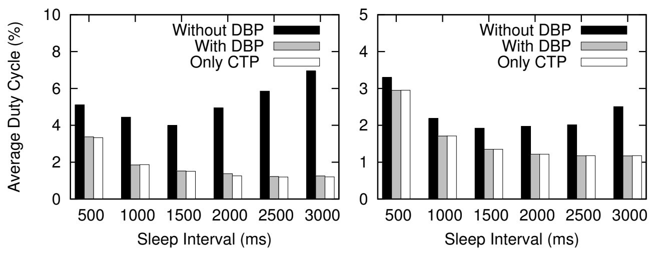
1) Data Delivery: DBP greatly reduces the amount of data in the network w.r.t. the baseline where all nodes send data every 30 s. The reduction in data transmitted reduces the probability of collisions, therefore increasing the delivery ratio. This is evident in Figure 10, where the system with DBP loses fewer messages than without DBP. In all cases the delivery is very good, above 97%, but DBP actually achieves 100% in all cases and in both scenarios, except for the case with the maximum sleep interval of 3000 ms in the testbed. In this case, a single model message was lost; however, as the absolute number of model changes is small, the total delivery ratio drops by almost 3%. Although this loss rate may be acceptable without DBP, losing a single DBP model has the potential to introduce large errors at the sink, as the latter will continue to predict sensor values with an out-of-date model until the next one is received. This suggests that, based on the target environment or parameter settings, dedicated mechanisms may be required to ensure reliability of model transmissions.

2) Lifetime: To study the impact on lifetime, we measure the duty cycle of the radio. Indeed, as this is the most power-hungry component, the time spent in communication activities is the most significant factor contributing to the system lifetime. Figure 11 clearly shows that DBP enables



(a) Testbed (2 hours). (b) Tunnel (23 hours).

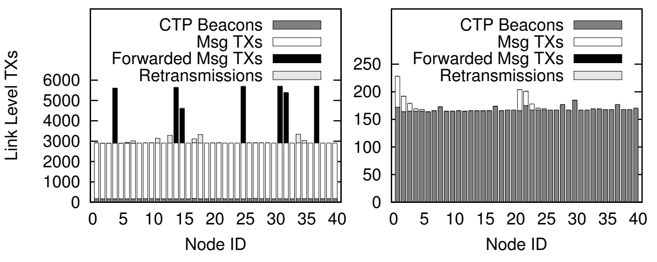
Figure 10. Delivery ratio.



(a) Testbed (2 hours).

(b) Tunnel (23 hours).

Figure 11. Average duty cycle. The y-axis scale is different.



(a) Without DBP.

(b) With DBP.

Figure 13. Tunnel: total link-level transmissions for a sleep interval of 1500 ms. The y-axis scale is different.

significant savings at any sleep interval. Indeed, the best sleep interval, corresponding to the lowest duty cycle, is 1500 ms without DBP. Further increasing the sleep interval decreases the idle listening cost, but it increases the transmission cost as the average transmission duration is half the sleep interval. This phenomenon instead bears a negligible effect in DBP where transmissions are greatly reduced. In this case, longer sleep intervals can be used to increase lifetime without affecting data delivery.

Figure 11(a) shows that in the testbed, with a sleep interval of 1500 ms (i.e., the best without DBP), DBP yields more than twice the lifetime of the no-DBP baseline—i.e., the WSN running DBP lasts twice as long, with the same MAC settings. Using the best sleep interval in both cases (i.e., 1500 and 3000 ms, respectively) yields a three-fold lifetime improvement. The energy savings in the tunnel, in Figure 11(b), are less remarkable although still significant. The network diameter in the tunnel is much smaller w.r.t. the testbed, due to the waveguide effect described in [8]; many direct, 1-hop links to the sink exist, leaving less room for improvement.

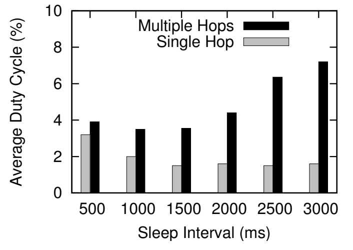


Figure 12. Tunnel: duty cycle by distance to the gateway, no DBP.

The impact of 1-hop links to the sink is worth commenting further. Indeed, because the sink is always on, it quickly receives and acknowledges a packet, making transmissions from its direct children very short and therefore low-energy. This can be seen clearly in Figure 12 where, for the tunnel experiments, we measure separately the duty cycle of the nodes that spent their entire lifetime directly connected to the sink and those that, at any time, were more than one hop away. Directly-connected nodes enjoy much lower energy costs. The plot considers only the case without DBP. Interestingly, with DBP *all* the nodes reporting model changes (Figure 9(b)) where in direct range of the sink. Indeed, as shown in Figure 1, the latter is attached to the gateway placed at the entrance, where light variations, and hence model changes, occur. This placement was not our deliberate choice, as it was originally determined by the available power panels in the tunnel. Nevertheless, it hints at the fact that, if *a priori* application knowledge is available about the sensors that are likely to generate the most variations, this can be exploited by a consequent placement of the gateway. A similar optimization is not possible without DBP, as all nodes must send data.

3) Routing Costs: A natural question arises at this point: if DBP suppresses over 99% of the messages, why does the network lifetime increase "only" three-fold? This is due to the costs of the network stack, in particular the idle listening and average transmission times of the MAC protocol, and to the overhead of the routing protocol to build and maintain the data collection tree. As we already evaluated the impact of the MAC layer, here we turn to the routing layer.

To isolate the inherent costs (e.g., tree maintenance) of CTP, we ran experiments with no application traffic. The corresponding duty cycle is shown as *Only CTP* in Figure 11; interestingly, the DBP cost is very close to the cost of CTP tree maintenance, regardless of the sleep interval. A finer-grained view is provided by Figure 13, where we analyze the different components of traffic in the network. Without DBP, the dominate component is message transmission and forwarding; significant retransmissions are present for some nodes, while the component ascribed to CTP (i.e., the beacons probing for link quality) is negligible. When DBP is active, the number of CTP beacons remains basically unchanged. However, because application-level traffic is dramatically reduced, CTP beacons become the dominant component of network traffic.

In conclusion, these last observations highlight that further reductions in data traffic would have little practical impact on the system lifetime, as routing costs are dominated by topology maintenance rather than data forwarding. Further, applying alternate data modeling techniques, e.g., PLA, SAF and POR, will not have a significant effect on system lifetime, as they cannot reduce these fixed, routing costs. Therefore, improvements are more likely to come from radical changes at the routing and MAC layers, taking into account the traffic patterns of model-driven data acquisition.

V. Related Work

The limited resources, variable connectivity, and spatio-temporal correlation among sensed values make efficiently collecting, processing and analyzing WSN data challenging. Early approaches use in-network aggregation to reduce the transmitted data, with later approaches addressing missing values, outliers, and intermittent connections [9]–[11].

Model-driven data acquisition has also been extensively studied. Probabilistic models [12], [13] approximate the data with a user-specified confidence, but special characteristics of the data, such as periodic drifts, must be explicitly encoded by domain experts. In a similar parametric approximation technique [14], nodes collaborate to fit a global function to local measurements, but this requires an assumption about the number of estimators required to fit the data. In contrast, DBP requires neither expert domain knowledge nor lengthy training, but provides hard accuracy guarantees on the collected data. PAQ [15], SAF [7], and DKF [16], employ linear regression, autoregressive models, and Kalman filters respectively for modeling sensor measurements, with SAF outperforming the others. All are applicable in our target application but, as shown in Section IV-A3, SAF is more sensitive than DBP to the noise in our dataset.

As an alternative to data modeling, some solutions seek to suppress reporting at the source by using spatio-temporal knowledge of data [17] or by identifying a set of representative nodes and restricting data collection to it [18]–[22]. Others take the remaining energy of individual nodes [23] into account. These approaches further reduce communication costs and can be applied in combination with DBP. Work on continuous queries for data streams studies the tradeoff between precision and performance when querying replicated, cached data [24]. Finally, several studies focus on summarizing streaming time series, showing that the choice of the summarization method does not greatly affect the accuracy of the summary [6]. In our experiments, we compared against PLA [6], as it can be efficiently computed.

The above data driven approaches have been evaluated theoretically, but no prior work explores the real effect of the network stack on the overall energy savings. Network-level energy savings approaches can be classified into MAC level, cross-layer, or traffic-aware.

At the MAC layer [25], low-power listening protocols such as BoX-MAC [4] dominate real deployments due to their availability, simplicity and effectiveness in reducing duty cycle. Nevertheless, as our analysis shows, parameters such as the listening interval must be carefully tuned.

Vertical solutions crossing network layers achieve extremely low duty cycles. Dozer [26] achieves permille (0.1%) duty cycle by taking a TDMA-like approach in which a tree parent autonomously schedules its transmissions to and from its children. Unfortunately, Dozer does not scale well and is prone to choose poor quality parents. Koala [27] achieves similar low duty cycles, but by explicitly accepting delays between data generation and delivery. Koala is characterized by long periods of very low-power local data sampling followed by brief, high-consumption data collection intervals. While the energy savings are significant, the significant delays are not acceptable in our target application.

Other techniques [28], [29] adapt sleep schedules according to traffic statistics. Unfortunately, the data modeling approaches outlined above, of which DBP is another example, are difficult to predict due to the variability of the application data itself and the interaction with the modeling technique.

VI. Conclusions

Model-driven data acquisition relies on the fact that many applications can operate with approximated data, as long as the difference w.r.t. the real one remains within certain limits. In these cases, WSN nodes can avoid reporting *all* sensed data, communicating only deviations from the trend.

In this paper, we proposed our technique, DBP, motivated by a real-world WSN-based application deployment in an operational road tunnel. Based on a 47-day, 40-node dataset gathered in this deployment we showed that DBP suppresses 99% of the message reports. This is in line with other approaches, although the DBP implementation is significantly less complex. Our results confirm that model-driven data acquisition can have a significant practical impact. However, we did not stop at counting the messages suppressed as an indirect indication of lifetime improvement. Instead, we took the whole network stack into account, discovering that the improvements remain important—lifetime is *tripled*—but significantly reduced w.r.t. the above.

Our results suggest a few conclusions. First, a large fraction of energy costs arise from the continuous maintenance of the data collection tree. These costs are negligible for frequent reporting, but become dominant with model-driven data acquisition as it greatly reduces data generation. To improve lifetime further, we must revisit network design choices and address the extremely low data rates resulting from data modeling techniques. Second, although a certain amount of loss is usually tolerable, the loss of a single data model may significantly increase the error of data used by the application. Therefore, reliable mechanisms, beyond those of most routing protocols, should be considered.

Based on our experiments, new network solutions expressly targeting model-driven data acquisition are needed to achieve significant lifetime improvements.

Acknowledgments

The authors thank the Autonomous Province of Trentino that partially funded this work within the TRITon project (http://triton.disi.unitn.it/). These activities have also been partially funded through the EU Cooperating Objects Network of Excellence (CONET—FP7-2007-2-224053).

References

[1] M. Ceriotti, M. Corrà, L. D'Orazio, R. Doriguzzi, D. Facchin, S. Guna, G. P. Jesi, R. L. Cigno, L. Mottola, A. L. Murphy, M. Pescalli, G. P. Picco, D. Pregnolato, and C. Torghele, "Is there light at the ends of the tunnel? Wireless sensor networks

for adaptive lighting in road tunnels," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2011.

[2] J. A. Stankovic, I. Lee, A. Mok, and R. Rajkumar, "Opportunities and obligations for physical computing systems," *Computer*, vol. 38, 2005.

[3] O. Gnawali, R. Fonseca, K. Jamieson, D. Moss, and P. Levis, "The collection tree protocol," in *Proc. of the Int. Conf. on Embedded Networked Sensor Systems (SenSys)*, 2009.

[4] D. Moss and P. Levis, "BoX-MACs: Exploiting Physical and Link Layer Boundaries in Low-Power Networking," Tech. Rep. SING-08-00, 2008.

[5] J. Polastre, R. Szewczyk, and D. Culler, "Telos: Enabling ultra-low power wireless research," in *Proc. of the Int. Conf. on Information Processing in Sensor Networks (IPSN)*, 2005.

[6] T. Palpanas, M. Vlachos, E. J. Keogh, and D. Gunopulos, "Streaming time series summarization using user-defined amnesic functions," *IEEE Trans. on Knowledge Data Engineering*, vol. 20, no. 7, 2008.

[7] D. Tulone and S. Madden, "An energy-efficient querying framework in sensor networks for detecting node similarities," in *Proc. of the Int. Conf. on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2006.

[8] L. Mottola, G. Picco, M. Ceriotti, S. Guna, and A. Murphy, "Not All Wireless Sensor Networks Are Created Equal: A Comparative Study On Tunnels." *ACM Trans. on Sensor Networks (TOSN)*, vol. 7, no. 2, 2010.

[9] L. Gruenwald, M. S. Sadik, R. Shukla, and H. Yang, "DEMS: a data mining based technique to handle missing data in mobile sensor network applications," in *Proc. of the Int. Conf. on Data Mgmt. for Sensor Networks (DMSN)*, 2010.

[10] A. Deligiannakis, Y. Kotidis, V. Vassalos, V. Stoumpos, and A. Delis, "Another outlier bites the dust: Computing meaningful aggregates in sensor networks," in *Proc. of the Int. Conf. on Data Eng. (ICDE)*, 2009.

[11] W. Wu, H.-B. Lim, and K.-L. Tan, "Query-driven data collection and data forwarding in intermittently connected mobile sensor networks," in *Proc. of Int. Conf. on Data Mgmt. for Sensor Networks (DMSN)*, 2010.

[12] A. Deshpande, C. Guestrin, S. Madden, J. M. Hellerstein, and W. Hong, "Model-driven data acquisition in sensor networks," in *Proc. of the Int. Conf. on Very Large Data Bases (VLDB)*, 2004.

[13] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong, "Approximate data collection in sensor networks using probabilistic models," in *Proc. of the Int. Conf. on Data Eng. (ICDE)*, 2006.

[14] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden, "Distributed regression: an efficient framework for modeling sensor network data," in *Proc. of the Int. Symp. on Information Processing in Sensor Networks (IPSN)*, 2004.

[15] D. Tulone and S. Madden, "PAQ: Time series forecasting for approximate query answering in sensor networks," in *Proceedings of the European Wkshp. on Wireless Sensor Networks (EWSN)*, 2006.

[16] A. Jain, E. Y. Chang, and Y.-F. Wang, "Adaptive stream resource management using Kalman filters," in *Proc. of the Int. Conf. on Management of Data (SIGMOD)*, 2004.

[17] A. Silberstein, G. Filpus, K. Munagala, and J. Y. 0001, "Data-driven processing in sensor networks," in *Proc. of the Conf. on Innovative Data Systems Research (CIDR)*, 2007.

[18] Y. Kotidis, "Snapshot queries: Towards data-centric sensor networks," in *Proc. of the Int. Conf. on Data Eng. (ICDE)*, 2005.

[19] Z. Zhou, S. Das, and H. Gupta, "Connected k-coverage problem in sensor networks," in *Proc. of the Int. Conf. on Computer Communications and Networks (IC3N)*, 2004.

[20] M. C. Vuran, O. B. Akan, and I. F. Akyildiz, "Spatio-temporal correlation: theory and applications for wireless sensor networks," *Computer Networks*, vol. 45, no. 3, 2004.

[21] H. Jiang, S. Jin, and C. Wang, "Prediction or not? An energy-efficient framework for clustering-based data collection in wireless sensor networks," *IEEE Trans. on Parallel Distributed Systems*, vol. 22, June 2011.

[22] M. Hassani, E. Mller, P. Spaus, A. Faqolli, T. Palpanas, and T. Seidl, "Self-organizing energy aware clustering of nodes in sensor networks using relevant attributes," in *Proc. of the Int. Wkshp. on Kowledge Discovery from Sensor Data (SensorKDD)*, 2010.

[23] R. A. F. Mini, M. D. V. Machado, A. A. F. Loureiro, and B. Nath, "Prediction-based energy map for wireless sensor networks," *Ad Hoc Networks*, vol. 3, 2005.

[24] C. Olston, J. Jiang, and J. Widom, "Adaptive filters for continuous queries over distributed data streams," in *Proc. of the Int. Conf. on Management of Data (SIGMOD)*, 2003.

[25] J. Rousselot, A. El-Hoiydi, and J.-D. Decotignie, "Low power medium access control protocols for wireless sensor networks," in *Proc. of the European Wireless Conf. (EW)*, June 2008.

[26] N. Burri, P. von Rickenbach, and R. Wattenhofer, "Dozer: ultra-low power data gathering in sensor networks," in *Proc. of the Int. Conf. on Information processing in sensor networks (IPSN)*, 2007.

[27] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis, "Koala: Ultra-low power data retrieval in wireless sensor networks," in *Proc. of the Int. Conf. on Information processing in sensor networks (IPSN)*, 2008.

[28] X. Ning and C. G. Cassandras, "Dynamic sleep time control in wireless sensor networks," *ACM Trans. on Sensor Networks*, vol. 6, June 2010.

[29] C. J. Merlin and W. B. Heinzelman, "Duty cycle control for low power listening mac protocols," in *Proc. of the Int. Conf. on Mobile Ad-hoc and Sensor Systems (MASS)*, 2008.