No-Sense: Sensing without Sensors

Chayan Sarkar, Vijay Rao, R Venkatesha Prasad
Embedded Software Group
Delft University of Technology
The Netherlands
{c.sarkar, v.rao, r.r.venkateshaprasad}@tudelft.nl

ABSTRACT

The lifetime of a wireless sensor network mainly depends on battery capacity and energy consumption at each node for operations such as, sensing, processing and data transmissions. Energy can be saved significantly if we can restrict the frequency of these operations. In this article, we propose *virtual sensing paradigm*, which reduces the frequency of the above operations at each node while not compromising on the sensing interval at the sink. Virtual Sensing creates virtual sensors at the sink by exploiting the spatio-temporal correlations between multiple physical sensors in the network. Using an adaptive filter based model, the virtual sensors can predict multiple consecutive sensing data of some sensors that are in sleep mode with the help of only a few active physical sensors. We show that even when the actual physical sensor and a virtual sensor of another sensor do not sense the same physical parameter, our proposed technique can work well. Applying our technique on the real-world data sets, we are able to show substantial reduction in energy consumption per node while maintaining high enough accuracy of the sensed data. To achieve higher energy reduction, virtual sensing paradigm has to be used in conjunction with various layers and protocols. Thus, virtual sensing paradigm has the potential to open up new research insights into the use of statistical properties of sensed data taken together in a network.

Keywords

Virtual Sensing, Sensor Data Estimation, Reduced Energy Consumption

1. INTRODUCTION

Wireless sensor networks have enabled continuous monitoring of an area of interest (body, room, region etc.) while eliminating expensive wired infrastructure. Over the years, the advancement in embedded technology has led to increased processing power and memory capacity for these battery powered devices. However, the batteries can supply limited energy, thus limiting the lifetime of the network. In order to prolong the lifetime of the deployment, various efforts have been made to improve the battery technologies and also to reduce the energy consumption of the sensor network at various layers in the network stack, in order to prolong the lifetime of the deployment. Since wireless data transmissions consume most of the energy, many works target reducing them through intelligent schemes like power control, reducing retransmissions etc. We propose a scheme that reduces number of transmissions itself.

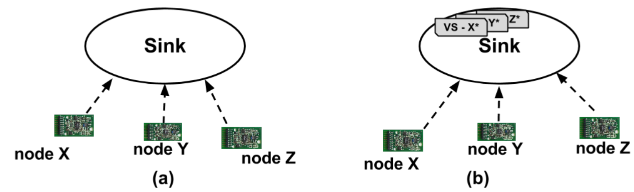

Figure 1: (a) Data collection scenario in a wireless sensor network (star topology); (b) Data collection with virtual sensing technique.

Typically in a wireless sensor network deployment, many parameters are application dependent. For example, placement of sensors, frequency of sensing and transmission of those sensed data. If the number of transmissions are reduced then significant amount of energy can be saved. In this article we propose a new paradigm, called *Virtual Sensing Paradigm* (VSP), which aims to satisfy application requirements while conserving energy at the sensor nodes.

In VSP, *Virtual Sensors* (VS) represent physical sensors to conserve energy. VS operate by predicting sensing data on behalf of the physical sensor by exploiting the temporal and spatial correlation among sensors' data. It is important to note that the VSP does not consider any *a priori* knowledge about the correlation structure among the sensors. This is known as non-model approach [15]. Rather, it creates adaptive estimation schemes for each sensor based on inherent correlation among the sensors' data. The virtual sensors follow collaborative and adaptive energy saving technique to maintain a balance between a high accuracy of the predicted values and energy consumption by each physical sensor. Since the VS "reside" in the sink/base station, the collaboration has almost zero overheads; the only overhead is notifying the physical sensors of their actions i.e., message notifying that the node may sleep for certain duration or reduce the data transmission frequency. The concept of VSP is shown in Fig. 1. Below we list the major contributions of this work:

- To the best of our knowledge, we are the first to propose a method that predicts multiple consecutive sensor values, while the sensor is in sleep mode, without having a-priori knowledge (or model) of the sensor value statistics. Even the physical nature of the sensing and deployment are not taken into account.

- Our proposal exploits both the spatial and temporal correlations among sensors without assuming any *a priori* knowledge about the correlations between sensed values of various sensors. Our method can track the changes in the correlation and adapts to the situation (through the use of a blind adaptive filtering technique).

- The energy saving technique proposed in this article ensures that every node spends energy almost evenly. We show that the overall energy consumption of the whole deployment is reduced, while maintaining high accuracy of the estimated sensor value.
- Predicting consecutive sensing data helps in keeping the sensor nodes in sleep mode for a longer duration whereby reducing the overhead of switching on and off the sensors.
- We have also provided many plausible extensions of this work that can enhance the ideas to achieve higher energy savings.

2. RELATED WORK

While many energy saving techniques/protocols have been proposed for wireless sensor networks, we only review here the work that is most relevant, important and similar in nature to our proposed scheme.

One common approach to reduce energy consumption is to select a subset of nodes among all the sensors deployed in the network. As the sensors show correlation, all the sensor values can be reconstructed from a subset of values. Most of these works do not calculate the correlation among sensors. Rather, a proper correlation structure among the sensors is assumed to be known a priori [3, 4, 5, 7, 9, 14]. We differ from these works since we do not assume any predefined correlation structure and our adaptive scheme can track the correlation among the sensors in real-time. Further, note that we do not draw any inference based on the physical nature of sensing, physical parameters and deployment of sensors to support our prediction.

A LMS-based adaptive prediction technique has been exploited by Santini *et al.* [15], which does not consider a priori knowledge about the sensor data. We have followed a similar strategy, but we make our prediction scheme adaptive. Moreover, we have exploited spatial correlation to estimate the sensing values blindly, i.e. our scheme can predict multiple consecutive sensing values while the sensor is in complete sleep mode. A real-time, blind prediction scheme has also been proposed by Li *et al.* [10]. However, they assume predefined spatial correlations among the sensors and the spatial correlation between the sensors is required to be very high.

Guestrin *et al.* [6] provided a technique where sensor nodes save energy by restricting their data transmission. The nodes only transmit the estimation model parameters, instead of the data itself, if any significant changes are noticed. In contrast with this method, our approach can predict consecutive sensor values accurately while the node is in complete sleep mode. There are some works where estimation techniques have been used to predict sensor data in order to fill the missing data points and complete the sensor data set, for example [13]. Thus, they are not suitable for real-time sensor data estimation for consecutive values.

3. VIRTUAL SENSING PARADIGM

The virtual sensing paradigm (VSP) aims to reduce energy consumption of a sensor network deployment by reducing its activity, i.e. reducing frequency of sensing, processing and data transmission. The data collection is complemented by predicting sensor data at the sink or the gateway of the network. To predict the sensor data, a virtual sensor (VS) is created for each physical sensor in a deployment at the sink as shown in Fig. 1. A VS instructs its associated physical sensor to reduce its activity, whenever it can predict the sensor value(s) within a certain error limit. The virtual sensors also collaborate among themselves in order to save more energy and maintain high data accuracy. In the subsequent sections, we discuss how sensors collaborate among themselves, before discussing how the prediction is done.

3.1 Energy saving technique

In order to increase the energy savings, the virtual sensing technique predicts successive values while keeping the physical sensor in deep sleep mode. As we do not assume any *a priori* knowledge about the sensor data statistics, any abrupt change in the phenomena might not be predicted by the temporal correlation based scheme. Further, if two sensors have had very high correlation in their past data streams, we can assume that both the sensor values will change in a similar fashion. Hence for VSP, we choose to exploit temporal as well as spatial correlations among sensors to capture abrupt changes in the ambiance while predicting multiple consecutive sensor values. This facilitates that between two correlated sensors, one sensor is kept in sleep mode for the duration of prediction of its sensor values, while the other sensor plays the role of a helper. The latter sensor aka helper is called as *companion*. The virtual sensor that puts its associated physical sensor in sleep mode is referred to *active virtual sensor* (AVS).

Energy can also be saved in the companion sensor when its virtual sensor can predict its values. In other words, when there is relatively no change in the sensing values, the companion sensor need not transmit its sensed values to the sink; those values can be predicted by exploiting the temporal correlation in the data of the companion by its virtual sensor. Such a virtual sensor is called *passive virtual sensor* (PVS). Note that, in this case, even though the (active) physical sensor observes the phenomena continuously, it only sends the data if the sensed value shows sufficiently large deviation from the expected value. In other words, if the prediction error lies within a permissible error threshold, then the sensor does not transmit the sensed data. By reducing only the data transmission, a significant energy can also be saved for the companions. Further, note that sensor data for AVS is predicted using PVS whenever the error threshold is not breached.

It is clear that an AVS can conserve more energy than a PVS. In order to maintain fairness in energy consumption of the sensor nodes, the roles of AVS and PVS switch after a certain number of iterations. Further note that every AVS requires a companion, but one companion can help multiple AVS. To maximize the energy saving, the VSP tries to assign as many active virtual sensors as it can, and reduce the number of companions.

In our implementation of the VSP, we have divided the whole data collection period into three phases - training period, operational period and revalidation period (Fig 2). The

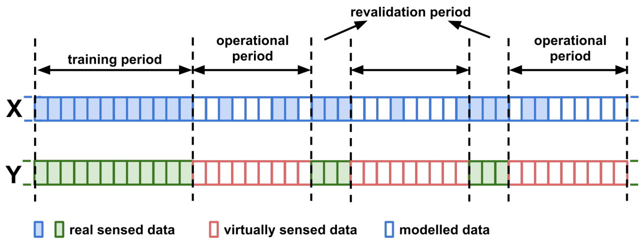

Figure 2: Data collection phases in virtual sensing technique.

following sections describe each phase in detail.

3.2 Training period

During the training period, N training data samples ($N > 0$) are collected from the physical sensor. As mentioned earlier, an AVS exploits temporal as well as spatial correlation, whereas a PVS uses only temporal correlation. Using the N training data samples, a temporal prediction mechanism is created for each virtual sensor at the end of the training period. To create a prediction mechanism based on temporal correlation, a transversal or tapped-delay line filter is created. Past p values of the same sensor are provided as filter input, where p is the order of the filter. To calculate the filter coefficients, first, the training data are stored in a $(N-p)\times p$ matrix (1) and $(N-p)\times 1$ column vector (see (2) as filter inputs and outputs respectively. Finally, the coefficients are calculated by minimizing the mean-square error as in (3) [8].

$$U = \begin{pmatrix} y[p] & y[p-1] & \cdots & y[1] \\ y[p+1] & y[p] & \cdots & y[2] \\ \vdots & \vdots & \ddots & \vdots \\ y[N-1] & y[N-2] & \cdots & y[N-p] \end{pmatrix} \tag{1}$$

$$\underline{y} = [y[p+1], y[p+2], ..., y[N]]^T \tag{2}$$

$$\underline{\alpha} = (U^T U)^{-1} U^T \underline{y} \tag{3}$$

These p filter coefficients, i.e. $\underline{\alpha} = [\alpha_1, \alpha_2, ..., \alpha_p]^T$ are used to predict the sensor data during the operational period (9). If the correlation is known a priori a *Wiener filter* can be developed, which is said to be the *optimum in the mean-square error sense*. As the autocorrelation function is unknown, the filter coefficients can become outdated and result in erroneous prediction. Using an adaptive filtering technique, the coefficients will be updated at later stages.

An AVS also need to develop a prediction mechanism based on spatial correlation. As it keeps the associated physical sensor in sleep mode for the entire operational period, the temporal correlation based estimation model might fail to capture abrupt changes in sensor data. A suitable companion node that remains active during the operational period captures those changes and can influence the prediction of the AVS. As a result, more accurate sensor values are predicted.

$$V = \begin{pmatrix} 1 & x[p+1] \\ 1 & x[p+2] \\ \vdots & \vdots \\ 1 & x[N] \end{pmatrix} \tag{4}$$

$$\underline{\beta} = (V^T V)^{-1} V^T \underline{y} \tag{5}$$

Linear regression is a statistical method, which models the relationship between a dependent variable and one/more independent variable(s) [12]. In our approach, we treat the companion as an independent variable that influences the AVS. The inputs for linear regression are collected from the companion node and are stored in a matrix form (4). The vector $\underline{y}$ (2) is used as the regressor output. Then, by minimizing the mean-square error, we calculate the coefficient of the linear regression (5). These coefficients, i.e. $\underline{\beta} = [\beta_0, \beta_1]^T$ are used for spatial prediction during the operational period (10), and they are also updated during the revalidation period.

3.2.1 Finding a suitable companion

Assuming the vector $\underline{x}$ represents the sensor values from the companion, the spatial prediction coefficients are calculated using (5). Again, since we do not assume any a priori spatial correlation, a companion cannot be chosen beforehand. A common assumption is that two geographically co-located sensors show high spatial correlation. Nevertheless, in reality, they may show poor correlation sometimes, whereas two sensors located relatively far can show high correlation too. By finding a suitable companion, we find a spatially correlated sensor that can predict the sensor values in the best possible way. Since we do not consider any a priori knowledge about the physical parameters of the sensed data we can indeed make a any sensor a companion for any other sensor during the training period. Training samples are collected from multiple sensor nodes that can become potential companions. Then, the linear regression coefficients are created separately for each sensors using (5). The sensor, which scores the highest in the *goodness of fit* test, is selected as the companion.

$$\underline{\hat{y}}_{spa} = V\underline{\beta} \tag{6}$$

$$\chi^2_{spa} = \sum_{i=p+1}^{N} \frac{(y[i] - \hat{y}_{spa}[i])^2}{\sigma^2} \tag{7}$$

$$\delta = 1 - \frac{\chi^2_{spa}}{\nu} \tag{8}$$

Chi-squared statistics is a well-known method to test goodness of fit [17]. To this end, the error values of the estimated signal need to be known. Using the model parameters and the training data set, first, the sensor values are estimated as in (6). Then, the Chi-squared statistics can be obtained by taking normalized sum of the squared-errors. In (7), the Chi-squared statistics is calculated, where σ^2 is the variance of the observed signal. To get an inference from the statistics, a reduced Chi-squared statistics can be calculated by simply dividing it by the number of degrees of freedom. The degrees of freedom, ν, in (8) is equivalent to the number of samples $(N-1)$.

The score of the goodness of fit test, represented as δ in (8) lies between $(0, 1)$, where 0 implies complete failure of

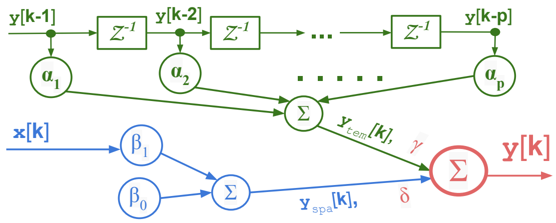

Figure 3: The filter used for an active virtual sensor is a hybrid of a transversal filter (for temporal prediction) and linear regression (spatial prediction).

capturing the system behavior using the model parameters and 1 implies complete resemblance of the system behavior by the model parameters.

3.3 Operational period

Once the prediction coefficients are created, the operational period starts (see Fig. 2). In this period, an AVS informs its corresponding physical sensor to follow its energy saving technique, i.e. to go to sleep mode for M sample duration. On the other hand, the virtual sensor predicts consecutive M data samples using a hybrid prediction model of temporal and spatial prediction (see Fig. 3).

$$y_{tem}[k] = \sum_{i=1}^{p} \alpha_i \cdot y[k-i] \quad (9)$$

$$y_{spa}[k] = \beta_0 + \beta_1 \cdot \hat{x}[k] \quad (10)$$

$$\hat{y}[k] = \frac{(\gamma \cdot y_{tem}[k] + \delta \cdot y_{spa}[k])}{(\gamma + \delta)} \quad (11)$$

At k^{th} time instance, first, the temporal prediction is done using the filter coefficients (9) during the training period. Then, the AVS collects the sensed value from its companion (actually, from the corresponding PVS). Using the companion's data and the linear regression coefficient, the spatial prediction is done (10). The final prediction is calculated by taking a weighted average of these two predicted values (11). The weights, i.e. γ and δ are calculated using the *Chi-squared statistics*. They are the *goodness of fit* score of the temporal and spatial predictions respectively.

During the operational period, a PVS informs its corresponding physical sensor to continue its sensing measurement at the predefined interval, but to avoid data transmission whenever possible. How does the physical sensor decide whether to transmit the data or not? As mentioned earlier, the PVS sends the filter coefficients to the physical sensor at the beginning of the operational period. At the k^{th} instance in operational period, the physical sensor predicts the sensor value using the filter coefficients (9) along with taking real measurement. Then, it calculates the prediction error (12). If the absolute error of the prediction is within the permissible (predefined) error threshold, then sensor discards the value. Otherwise, the sensor transmits the value to the sink and the model coefficients are updated. In the first case, the corresponding virtual sensor predicts the sensed value using the same filter available at the physical sensor. On the other hand, if the actual sensed value is received from the sensor, the model parameters are updated the same way as in the physical sensor. In this way, the model parameters remain in sync at both the places. At the same time, we can ensure that the prediction error remains within the permissible error threshold. To update the model parameters (at both the places), we have used a similar approach of the *least-mean-square (LMS) algorithm*. LMS is an adaptive filtering technique which is widely used in time-series prediction. It requires very less memory and computational capabilities, and performs very well [15].

$$e_{temp}[k] = y[k] - y_{temp}[k] \quad (12)$$

$$\underline{\alpha}[k+1] = \underline{\alpha}[k] + \mu \cdot \underline{y}[k] \cdot e_{temp}[k] \quad (13)$$

Let's assume, at the k^{th} instance, the actual and predicted sensor values are $y[k]$ and $y_{temp}[k]$ respectively. To update the temporal filter coefficients, i.e. the $\underline{\alpha}$ vector, first, the prediction error is calculated given by (12). Then, the filter coefficients are updated using (13) [8], where $\underline{y}[k] = [y[k-1], y[k-2], ..., y[k-p]]^T$ is the input vector of the filter, and μ is the learning rate of the adaptive algorithm. The value of mu is set according to [15].

3.4 Revalidation period

As an AVS asks the corresponding physical sensor to go to sleep mode, there is a chance that the predicted value might diverge from the ground reality. To tackle this, a revalidation of the model parameters are done after M sensing intervals. Revalidation period (i.e., R sensing instances), the physical sensor becomes active. That is it senses and transmits the data to the sink (see Fig. 2). Then, the AVS updates its temporal prediction coefficients as the same way as described for a PVS. The spatial prediction coefficient, i.e. $\underline{\beta} = [\beta_0, \beta_1]^T$, are updated based on the spatial prediction error using (14) and (15).

$$e_{spa}[k] = y[k] - y_{spa}[k] \quad (14)$$

$$\underline{\beta}[k+1] = \underline{\beta}[k] + \mu \cdot \underline{x}[k] \cdot e_{spa}[k] \quad (15)$$

$$\chi^2_{tem} = \chi^2_{tem} + \frac{e^2_{tem}[k]}{\sigma^2} \quad (16)$$

$$\chi^2_{spa} = \chi^2_{spa} + \frac{e^2_{spa}[k]}{\sigma^2} \quad (17)$$

As the final prediction of a AVS is dependent on γ and δ (11), an updated *goodness of fit* score, i.e. the *Chi-squared statistics* are also calculated (16)-(17). In this period the role of a PVS and its corresponding physical sensor remains the same as it was during the operational period.

4. EVALUATION

In this section we present some results when we applied our virtual sensing technique on real data sets. A set of real world measurement data available at [2] is used. The Lausanne Urban Canopy Experiment took place in the EPFL campus between July 2006 and May 2007. During this period, 97 sensor nodes monitored various environmental parameters, e.g. ambient temperature, solar radiation, relative Humidity, etc. These sensor nodes collected data every 30 s. We have applied VSP on the ambient temperature data collected from multiple sensors. To report our results, we chose node 3 and node 44, and assigned them the roles of active and passive virtual sensors respectively. We implemented our prediction technique in Matlab.

1000 data samples from sensor node 3 and 44 are used for prediction. A snapshot of the predicted sensor data by the virtual sensors are shown in Fig. 4. For this simulation,

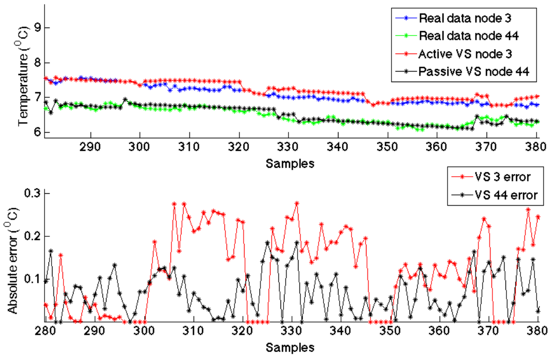

Figure 4: Sensor data predicted using an active and a passive virtual sensor.

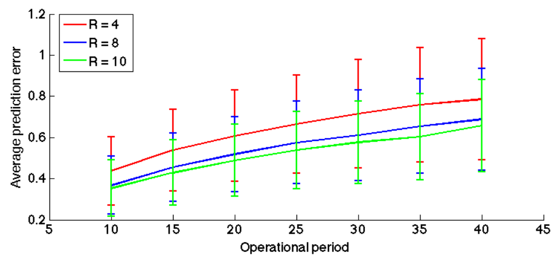

Figure 5: Average prediction error for various operational period and revalidation period (R).

the operational window and the revalidation window were fixed at 20 samples and 5 samples respectively. The snapshot shows multiple operational and revalidation periods. The training was done at the beginning of the data streams, which is not shown in the figure. The result shows that the virtual sensors can achieve significant prediction accuracy.

To further investigate the accuracy of the VSP based prediction, we have conducted a couple of simulations by varying the operational window size (M) and revalidation window size (R). The average error with the standard deviation of error as a confidence interval for various values of the <M, R> pair is shown in Fig. 5. From this study, it is evident that if we increase the revalidation window size, the average prediction error, as well as the variance of prediction error is reduced. Nonetheless, it is at the cost of energy consumption due to more data transmission. To tackle this we can also increase the operational window size. However, with increased operational window, the average prediction error also increases. The choice of these window sizes depends on the applications. Accuracy and energy cost goes together. From our simulation results, we can conclude that VSP provides the tool to restrict data transmissions from sensor nodes with a very less reduced quality of data.

Next, we show that the virtual sensing technique can save more energy using blind estimation. Three simulations are conducted for three different lengths of operational period (M) and revalidation period (R). For each pair of M and R, the error threshold is also varied from 0°C to 2°C. Error threshold 0 represents the base case, where all 1000 data from the sensors are sensed and transmitted.

Now, let us take the energy saving through VSP. The sys-

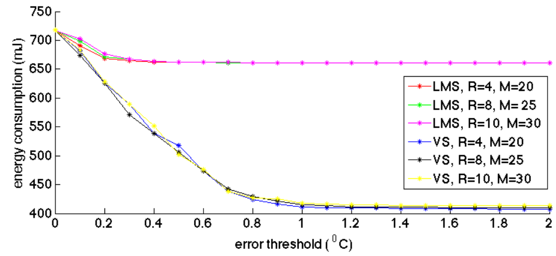

Figure 6: Combined energy consumption for two nodes (one active and one passive) using virtual sensors v/s both the sensor values are predicted using the LMS-based scheme [15].

Table 1: System Parameters and Settings

Parameter	**Value**
Message size	12 B
Transmission distance	50 m
Energy cost for sending a message	28.8 μJ
Energy cost for sensing temperature	330 μJ
Energy cost in idle mode	33 mW
Energy cost in low power mode	10 mW
Energy cost in sleeping mode	0.016 mW

tem parameters and their settings are listed in Table 1 to calculate the energy consumption. The values for each parameters are calculated using methods described in [11, 17]. Using these parameters and the number of times a sensor senses and transmits data, energy consumption per node is being calculated. Though an AVS conserves more energy than a PVS, an AVS is always accompanied by a passive one. Thus, the combined energy consumption is calculated and compared with the LMS-based method described in [15]. As the error threshold increases, virtual sensors consume lesser energy at the physical sensors. Note that energy consumptions calculated here are based on only sensing and data transmissions. Using VSP, an AVS keep the corresponding physical sensor in sleep mode during the operational period, where as a sensor only remains idle between two successive measurements using LMS-based technique. As the energy consumed in sleep mode is very less as compared to that of idle mode or low power mode, further energy consumption can be achieved.

As an extension of the virtual sensing technique, we have also tried *heterogeneous virtual sensing*, where the companion sensor is monitoring a different physical parameter. We have tried to predict temperature using a light sensor, see Fig. 7. The light and temperature sensor data are collected from [1]. This shows the effectiveness of heterogeneous virtual sensing. However, there are some issues to be addressed such as, light sensor data can change quickly over a short period of time, while temperature changes gradually. This affects accuracy of predictions. Fine tuning our technique to adapt to the situation based on estimated prediction errors may help in increasing the usability of VSP in large scale deployment. Further, VSP could also be used in case some type of sensors is not available at a location. To achieve better error bounds, more investigations are required with respect to heterogeneous virtual sensing.

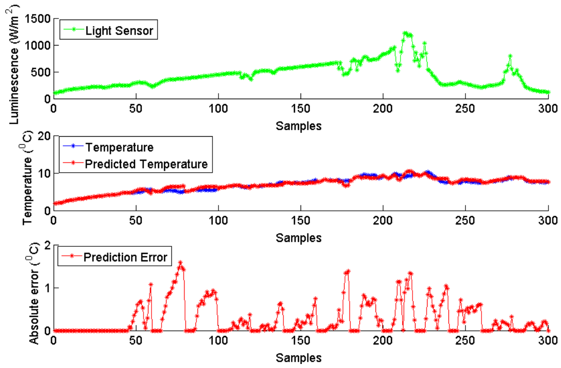

Figure 7: Temperature sensor values are predicted using a light sensor as its companion.

5. FUTURE WORK

The work hitherto has opened up many avenues to further build on it for achieving higher energy efficiencies. We have identified a number of issues that should be addressed to improve the current technique. We enlist them here.

1. We have used fixed length for the three phases of sensor data collection. Moreover, the order of the temporal estimator model, p in (9) and the enhancement factor of the adaptive model, μ in (13) have fixed values. All these fixed parameters need to be adaptive according to the nature of the sensor data, because spatio-temporal correlation may vary with time. It is a challenge to find a model that tracks the changes in these parameters.
2. The role of active and passive virtual sensors are predefined. Currently, we are working to automate the process of role assignment and role switching. Additionally, we want to maximize the number of active sensors so as to increase energy saving.
3. In the current scheme, the prediction within a predefined error bound can only be achieved by a passive virtual sensor. Since an active virtual sensor predicts the sensor value blindly, it is difficult to achieve strict error bounds. We need to adjust the prediction scheme in a way so that a confidence bound for the error can be defined.
4. The physical sensor corresponding to an AVS will be in sleep mode, thus we will never be able to find the real error as in the case of PVS. We are investigating methods to some how determine this error. Coupled with a method to estimate this error, a wake up radio could be brought in to make immediate corrections.
5. For each active sensor only one companion is used in our study. We want to use multiple companions so that the prediction accuracy can be improved.
6. In a large sensor network, multi-hop data transmission is a necessity. While using virtual sensing, – an important requirement and currently an unanswered question – is to ensure the connectivity of the network even if some nodes are in sleep mode. This requires an unusual type of clustering of sensors wherein sensed values of a sensor in a cluster can be predicted by other sensors in the same cluster. In this type of clustering the proximity of sensors are not used. The challenge is to pick the least number of nodes from each cluster to maintain the connectivity while increasing energy savings.

The above list is not complete and we believe that there are more vistas for enhancing current techniques and introducing newer ideas.

6. CONCLUSIONS

In this article, the virtual sensing technique is introduced, which predicts multiple consecutive sensor data while keeping the physical sensor in complete sleep mode. We have utilized the inherent spatio-temporal correlation among the sensor data without having any a priori knowledge about the statistics of the data and also the physical parameters the sensors are observing. A case in point is predicting temperature with a light sensor to a large extent. The proposed prediction scheme adapts to the changes in sensor data. Using our propose technique, we have achieved a significant improvement on energy saving as compared to other methods while maintaining high accuracy of sensor data. We have also listed many vistas for further investigations building on the proposed technique. We believe that our technique will be useful when large number of sensors are deployed and with the advent of Internet of Things (IoT) paradigm.

7. REFERENCES

[1] Nrel: Lowry range solar station, colarado state land board [online]. http://www.nrel.gov/midc/.

[2] Sensorscope: Lausanne urban canopy experiment deployment [online]. http://lcav.epfl.ch/cms/lang/en/pid/86035.

[3] D. Chu, A. Deshpande, J. M. Hellerstein, and W. Hong. Approximate data collection in sensor networks using probabilistic models. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 48–48. IEEE, 2006.

[4] R. Cristescu and M. Vetterli. On the optimal density for real-time data gathering of spatio-temporal processes in sensor networks. In *Proceedings of the 4th international symposium on Information processing in sensor networks*, page 21. IEEE Press, 2005.

[5] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 588–599. VLDB Endowment, 2004.

[6] C. Guestrin, P. Bodik, R. Thibaux, M. Paskin, and S. Madden. Distributed regression: an efficient framework for modeling sensor network data. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, IPSN '04, pages 1–10, New York, NY, USA, 2004. ACM.

[7] H. Gupta, V. Navda, S. Das, and V. Chowdhary. Efficient gathering of correlated data in sensor networks. *ACM Trans. Sen. Netw.*, 4(1):4:1–4:31, Feb. 2008.

[8] S. S. Haykin. *Adaptive Filter Theory, 4/e*. Pearson Education India, 2005.

[9] A. Jain, E. Y. Chang, and Y.-F. Wang. Adaptive stream resource management using kalman filters. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 11–22. ACM, 2004.

[10] Y. Li, C. Ai, W. Deshmukh, and Y. Wu. Data estimation in sensor networks using physical and statistical methodologies. In *Distributed Computing Systems, 2008. ICDCS '08. The 28th International Conference on*, pages 538 –545, june 2008.

[11] C. B. Margi, B. T. de Oliveira, G. T. de Sousa, M. Simplicio, P. S. Barreto, T. C. M. Carvalho,

M. NaÌĹslund, and R. Gold. Impact of operating systems on wireless sensor networks (security) applications and testbeds. In *Computer Communications and Networks (ICCCN), 2010 Proceedings of 19th International Conference on*, pages 1–6. IEEE, 2010.

[12] D. C. Montgomery, E. A. Peck, and G. G. Vining. *Introduction to linear regression analysis*, volume 821. Wiley, 2012.

[13] L. Pan. K-Nearest Neighbor Based Missing Data Estimation Algorithm in Wireless Sensor Networks. *Wireless Sensor Network*, 02(02):115–122, 2010.

[14] Z. Quan, W. J. Kaiser, and A. H. Sayed. A spatial sampling scheme based on innovations diffusion in sensor networks. In *Proceedings of the 6th international conference on Information processing in sensor networks*, IPSN '07, pages 323–330, New York, NY, USA, 2007. ACM.

[15] S. Santini and K. Romer. An adaptive strategy for quality-based data reduction in wireless sensor networks. In *Proceedings of the 3rd International Conference on Networked Sensing Systems (INSS 2006)*, pages 29–36, 2006.

[16] X. Tang and J. Xu. Extending network lifetime for precision-constrained data aggregation in wireless sensor networks. In *INFOCOM*, 2006.

[17] J. R. Taylor. *An introduction to error analysis: the study of uncertainties in physical measurements*. University science books, 1997.