2014 IEEE International Conference on Internet of Things (iThings 2014), Green Computing and Communications (GreenCom 2014), and Cyber-Physical-Social Computing (CPSCom 2014)

Continuous Monitoring and Distributed Anomaly Detection for Ambient Factors

Yang-Chi Shen*, Alvin Chiang*, Yi-Ren Yeh† and Yuh-Jye Lee*
*Department of Computer Science and Information Engineering
National Taiwan University of Science and Technology, Taipei, Taiwan
Email: {M10115003, yuj-jye, M10115088}@mail.ntust.edu.tw
†Department of Applied Mathematics, Chinese Culture University, Taipei, Taiwan
Email: yyr2@ulive.pccu.edu.tw

***Abstract*—Considering the diverse application scenarios involving wireless sensor networks (WSNs), accurate continuous monitoring requires a solution to the essential task of estimating unmeasured locations in the monitored space. In this paper, we utilize ε-Smooth Support Vector Regression (ε-SSVR) to report monitoring information of environment, furthermore we combine spatial and temporal correlation to strengthen monitoring accuracy. However if our sensors are too sparsely deployed, the resulting coverage holes problem will adversely impact the monitoring result. Therefore, we utilize Uniform Design and different local interpolation methods to assist ε-SSVR to mitigate the coverage holes problem. In our experiment, we compare our method with different methods applied to different sensors deployments. ε-SSVR has better accuracy and computation speed than others. Besides continuous monitoring, we also propose a distributed anomaly detection mechanism to report anomaly information, in order to provide a reliable and real time anomaly monitoring system.**

***Keywords*-ε-SSVR; wireless sensor networks; anomaly detection; continuous monitoring;**

I. Introduction

In the past few years, wireless sensor networks (WSNs) have been widely applied in different domains (e.g., agriculture, traffic control, and the management of indoors environments [2, 7, 24, 29]). WSNs are used to monitor and track events of concern [37]. For example, through the use of sensors farmers are able to monitor environmental factors such as temperature and humidity giving them the chance to react quickly to changing crop conditions and ensure an optimal harvest. In many cases, the value of these WSNs lies in their ability to act as precise continuous monitoring systems. It is desirable to monitor the region of interest as uniformly and densely as possible, however, it is often impractical to do due to restrictions in sensor placement and availability. In order to reduce this problem, interpolation methods are often used to fill in the blanks where there are no sensors. Traditional methods include Ordinary Kriging (OK) and Inverse Distance Weighting (IDW) [36,37], which calculate a weighted sum of measurements from surrounding sensors to interpolate a surface over the region. However, if there is large variability in the data environment, it may cause great monitoring error, because the interpolated values will fall within the minimum and maximum reading range, which will underestimate the highs and overestimate the lows.

In order to solve this problem, we use ε-Smooth Support Vector Regression (ε-SSVR) [22] to interpolate a surface over the region. ε-SSVR constructs a regression model over all the sensors, and predicts the region values. Predicted results are not subject to maximum and minimum range limits, which can effectively reflect real environment. In addition to consider the spatial influence of sensors, we also refer to historical information to strengthen our calculation [31]. Taking into account existing spatial and temporal relationships allows more accurate interpolation. In spite of ε-SSVR having good ability for interpolation, it will perform poorly if sensors are too sparsely deployed due to the coverage holes problem [1]. Several different reasons will cause coverage holes. (1) Obstacles of different forms such as mountains, lakes and rocks [32]. (2) Technical failures, malicious activities or power exhaustion cause accidental death of the nodes [25]. (3) Scale or the hostility of monitored area such as ocean buoys. [30].

To combat the coverage holes problem, we generate synthetic sensors in the field using Uniform Design (UD) [19] so that projecting the location of synthetic sensors onto any dimension results in a uniform distribution. The UD seeks its design points to be uniformly scattered on the experimental domain. This guarantees that all coverage holes will have at least one synthetic sensor. Next, we find a localized estimate for the synthetic sensor readings using one of the aforementioned methods, Ordinary Kriging or Inverse Distance Weighting, over their K nearest neighboring real sensors. Since the synthetic sensor readings are predicted, the error is larger compared to the real sensors, thus we give a higher level of fault-tolerance to the synthetic sensor than we do to the real sensors in the model training stage. Low fault-tolerance means that we will adjust model hyperplane for tiny error in fitting a given data set linearly or nonlinearly, otherwise means we will ignore tiny errors. Our ε-SSVR allows the integration of synthetic and real sensors to improves the performance and monitoring accuracy when there are coverage holes. In our experiments, we show that our approach can get better

978-1-4799-5967-9/14 $31.00 © 2014 IEEE
DOI 10.1109/iThings.2014.14

performance than other interpolation methods not only in terms of monitoring accuracy but also computation speed.

In addition to continuous monitoring, we also detect anomalous events through a distributed anomaly detection mechanism [5,9,27,28]. Distributed anomaly detection can be divided into two phases. The first part is the front-end detection, where a simple anomaly detection model is constructed on each sensor, the purpose of which is to detect anomalous events in real-time. The second part is the back-end detection, which uses various of continuous monitoring result to detect anomalies, and provides a visual interface allowing users to understand the scope of the occurrence of the anomaly. These two steps are combined to achieve real-time and reliable anomaly detection.

This paper is organized as follows. Sec. II, we address some related works of the continuous monitoring and anomaly detection. Sec. III describes our approach of continuous monitoring and how to combine spatial and temporal correlation. The strategy of coverage holes problem is introduced in Sec. IV. In Sec. V, we describe how to utilize distributed anomaly detection mechanism to implement real time anomaly detection. Our experiment result is showed in Sec. VI. The conclusion and future work will be described in Sec. VII.

II. Related Work

We discuss related work in this section, include continuous monitoring and distributed anomaly detection. In continuous monitoring, interpolation is the most common method for creating the surface over the region. Two famous interpolation methods are Kriging and inverse-distance weighted interpolation (IDW). Jafar et al. [36] Dale et al. [37] and Reiner et al. [18] compared these two methods in different environments. All results show that Kriging performs much better than IDW, because Kriging not only considers the distance from the sensors, but also takes into account all sensor's correlations. George et al. [23] and Mohammad et al. [14] modified the IDW formula so that it achieved higher monitoring accuracy than Kriging. Both methods are not only utilized in monitoring, but can also be used to reduce coverage holes and discover anomalies. Muhammad et al. [20] used Kriging to reconstruct regions with coverage holes. Yunfeng et al. [35] used different interpolation methods to discover soil heavy metal pollution. It can be seen that continuous monitoring is widely use for different goals. Furthermore, Uwe et al. [13] used radar readings to modify the kriging result in order to consider more physical factors.

Tomislav et al. [16] proposed Regression Kriging to monitor the region of interest. This method is most similar to ours, because it combines regression and interpolation to make estimates. Regression Kriging utilizes linear regression to predict all the unmeasured locations, next it modifies the regression residual through Ordinary Kriging; these steps can efficiently promote monitoring accuracy. However, Regression Kriging does not solve the coverage holes problem. Our ε-SSVR utilizes uniform design and different fault tolerance settings to reduce this problem.

In their survey of distributed anomaly detection, Miao et al [34] and Varun et al [8] cover different categories of anomalies and solutions to detecting these anomalies. Distributed anomaly detection has higher computation speed than centralized method, and can be implemented in real-time anomaly detection systems. However, sensors have a rather limited amount of computational resources available. In order to implement distribution in each node, previous researchers proposed several anomaly detection algorithms. Vassilis et al. [12] defined a threshold based on environment variance to determine anomalies. Sutharshan et al. [21] and Wenliang et al. [11] used local clustering to distinguish normal data and anomalies at each node. Joel et al. [6] not only detected anomalies but also reduced communication energy cost. In addition to detecting anomalies at a single node, people also want to know the region where the anomaly occurred. Franke et al. [10] used a simple threshold method to determine the range within which the anomaly may have occurred. Annalisa et al. [4] rely on spatial and temporal correlation to predict anomalous event locations. Our distributed anomaly detection uses front-end and back-end mechanisms, can detect anomalies in real-time at each sensor node, and provides a visual interface allowing users to understand the scope of the occurrence of the anomaly.

III. Continuous Monitoring via ε-SSVR

A. Continuous Monitoring

Figure 1 shows the process of our continuous monitoring. First, all the sensor readings are aggregated at the base station, where they are synchronized and then filtered and cleaned. Next we evaluate all sensors' coverage over the region. If the sensors did not adequately cover all locations, we use uniform design and an interpolation method to create synthetic sensors to reduce coverage holes. Otherwise, we only need to use ε-SSVR. Finally, we produce a continuous visualization map. This allows users to receive in real-time information about changes in the environment.

B. The ε-Smooth Support Vector Regression

In conventional regression, we are given a training dataset $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ to find function $f(x)$ with least squares error where $x_i \in R^n$ is independent variable and $y_i \in R$ is dependent variable. The least squares problem is formulated as following:

$$\min_{(\mathbf{w},b)\in R^{n+1}} \quad \sum_{i=1}^{m} \|(y_i - x_i^\top \mathbf{w} - b)\|_2^2, \tag{1}$$

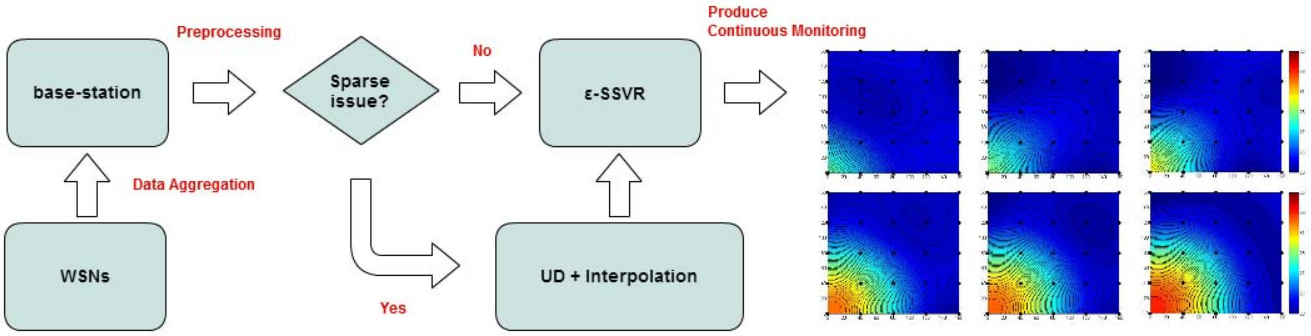

Figure 1. The process of continuous monitoring.

where $\mathbf{w} \in R^n$ is weight, $b \in R$ is bias.

Different from conventional regression, ε-support vector regression adds a tolerance variable in its optimization process to discard small errors in fitting the given dataset linearly and nonlinearly. The loss function is written as $(|A_i\mathbf{w} + b - y_i|_\varepsilon) = max\{0, |A_i\mathbf{w} + b - y_i| - \varepsilon\}$. Using this loss function means if it is not over the critical range, the error will be zero. Figure 2 diagrams the fault-tolerant mechanism of ε-insensitive regression.

The ε-SVR can be formulated as a constrained minimization problem in Equation (2)

$$\begin{aligned} \min_{(\mathbf{w},b,\xi,\xi^*)\in R^{n+1+2m}} \quad & \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\mathbf{1}^\top(\xi + \xi^*) \\ \text{s.t.} \quad & y - A\mathbf{w} - \mathbf{1}b \leq \mathbf{1}\varepsilon + \xi \\ & A\mathbf{w} + \mathbf{1}b - y \leq \mathbf{1}\varepsilon + \xi^* \\ & \xi, \xi^* \geqslant 0, \end{aligned} \tag{2}$$

where $\mathbf{w}^\top\mathbf{w}$ is the regularization term, $\xi \in R^m$ is the slack variable, $A \in R^{m\times n}$ and m observations of real value associated with each point. $y \in R^m$ is the response vector, $\mathbf{w} \in R^n$ and $b \in R$. ξ is greater than 0 when estimated values fall outside the tube defined by ε, otherwise it is 0. As seen in Figure 2, ξ will discard tiny errors in the modelling procedure. Parameter C here weights the tradeoff between the fitting errors and the flatness of the linear regression function $f(x)$. Increasing parameter C will achieve a better accuracy on the training set, but it can lead to over-fitting.

This problem can be also formulated as an unconstrained minimization problem with a squared 2-norm ε-insensitive loss function given in Equation (3):

$$\min_{(\mathbf{w},b)\in R^{n+1}} \quad \frac{1}{2}\mathbf{w}^\top\mathbf{w} + \frac{C}{2}\sum_{i=1}^{m} \|(|A_i\mathbf{w} + b - y_i|_\varepsilon)\|_2^2 \tag{3}$$

(The weight is $\frac{C}{2}$ instead of C, to cancel out the extra constant after differentiation.)

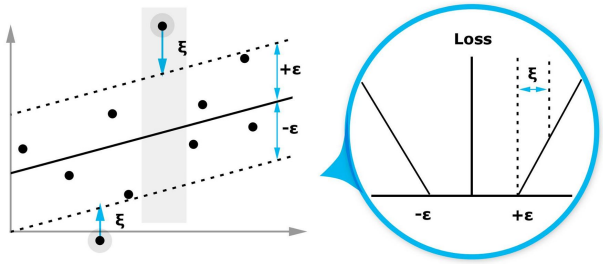

Figure 2. ε-insensitive regression. ε is a tolerance variable in its optimization process to discard small errors in fitting the given dataset linearly and nonlinearly. ξ will discard tiny errors in the modelling procedure.

A problem with the 2-norm ε-SVR is that it is not twice differentiable. To make the objective function twice differentiable, Lee et al [22] used the smooth p-function to replace the minimizing error term $\|(|A_i\mathbf{w} + b - y_i|_\varepsilon)\|_2^2$ of ε-SVR. After being reformulated as the ε-insensitive smooth support vector regression, the optimization problem can be quickly solved using Newton's method. The details of ε-SSVR can be seen in [22]

C. The ε-Smooth Support Vector Regression with Nonlinear Kernel

In order to generalize our experiments from the linear case to the nonlinear case, we employ the kernel technique to represent the inner product of two training points in a higher dimensional feature space, so that our linear formulation can be transformed into a nonlinear formulation. The kernel function, $K(x, z) =< \Phi(x)\cdot\Phi(z) >$, merges two steps. Step 1: mapping the input data from input space to feature space. Step 2: calculating the inner product in the feature space. There are many forms of kernels and the Gaussian (Radial Basis) kernel (shown in Equation (4)) is one of the most commonly used:

$$K(A, A^\top)_{ij} = e^{-\gamma\|A_i - A_j\|_2^2}, i, j = 1, 2, \ldots, m. \tag{4}$$

γ is a tuning parameter determining how far the influence of a single training example reaches. Low values tend to result in under-fitting, and higher values tend to result in over-fitting.

D. Spatial and Temporal Correlation

We will describe how to combine spatial and temporal correlation for model training stage in this section. Table I are each sensor information from wireless sensor networks, include ID, location, temperature and time.

Table I
SENSORS INFORMATION FROM WIRELESS SENSOR NETWORKS.

NodeID	LocationX	LocationY	Temperature	Time
1	0	120	27.4	00:38:27
2	40	80	23.7	00:38:27
3	80	40	23.5	00:38:27
4	120	0	23.7	00:38:27

The data all come from the same time. In our given training dataset $S = \{(x_1, y_1), \ldots, (x_m, y_m)\}$, x_i = locationX and locationY and y_i = Temperature. We use two dimension locationX and locationY training regression model to predict the temperature at unmeasured location. In addition to considering spatial correlation, we also use Auto-Regressive and Moving Average Model (ARMA) to strengthen our prediction. ARMA models be used to predict behavior of a time series from past values alone. Such a prediction can be used as a baseline to evaluate the possible importance of other variables to the system. We combine past T values and the original two dimensional location (locationX and locationY) to strengthen our prediction. Table II data from one sensor at different times with the time interval set to one minute. If we define T = 3, then the past three y_i temperatures will be added into the feature set. For example, node 1's feature set x_i = [0, 120] will become x_i = [0, 120, 27.4, 27.9, 30.1]. Combining location and past labels to train the model works better than only considering spatial correlation, because some nodes may not be affected by their neighbors. In spite of ε-SSVR having good ability for monitoring, it will perform poorly if sensors are too sparsely deployed due to the coverage holes problem. To address this problem, we proposed using ε-SSVR with synthetic sensors. The details will be described in the following section.

Table II
ONE SENSOR INFORMATION IN DIFFERENT TIME.

NodeID	LocationX	LocationY	Temperature	Time
1	0	120	27.4	00:38:27
1	0	120	27.9	00:39:27
1	0	120	30.1	00:40:27
1	0	120	31.4	00:41:27

IV. REDUCING COVERAGE HOLES PROBLEM

In this section, we will introduce how to reduce the coverage holes problem. Several different reasons will cause coverage holes. (1) Obstacles of different forms such as mountains, lakes and rocks. (2) Technical failures, malicious activities or power exhaustion cause accidental death of the nodes. (3) Scale or the hostility of monitored area such as ocean buoys. Coverage holes happen when sensors are distributed too sparsely. This phenomenon will cause large prediction errors of ε-SSVR. Predicting the value of variables in the coverage hole is difficult due to there being no sensors values on which to rely. In order to reduce this problem, we proposed a sampling method uniform design to solve.

A. Synthesizing Sensors via Uniform Design Sampling

Uniform design (UD) was first proposed by Fang et al [19]. The uniform design is one kind of space-filling designs that can be used for computer and industrial experiments. The UD seeks its design points to be uniformly scattered on the experimental domain. This is a method to uniformly generate nodes in different amount. Figure 3 shows the different uniform design patterns, include nine points and thirteen points, as the case to decide which pattern to used. We use these patterns to generate various amounts of synthetic sensors. The advantage of using uniform design to do this is that projecting the location of synthetic sensors onto any dimension results in a uniform distribution. This guarantees that all coverage holes will have at least one synthetic sensor. Although synthetic sensors are not real, they can help with our prediction result. We combine real sensors and synthetic sensors, where the coverage holes have been filled by synthetic sensors.

B. Combining Local Interpolation Methods and ε-SSVR

After deploying our synthetic sensors, we need have them calculate readings to help in the training model stage. We utilize different localized interpolation methods to estimate synthetic sensor readings, because they are suitable to be used in environments with low variability. It is our assumption that each local field has low variability. Figure 4 shows the interpolation process. First, we generate synthetic sensors in the field and calculate distances between real and

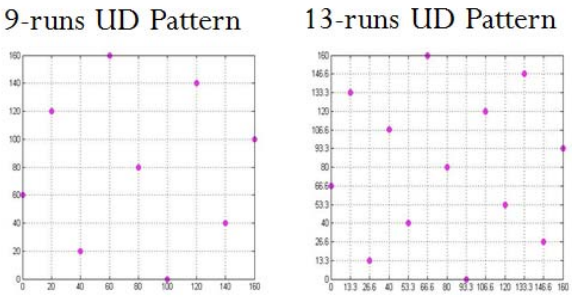

Figure 3. Uniform design sampling cases include nine points and thirteen points.

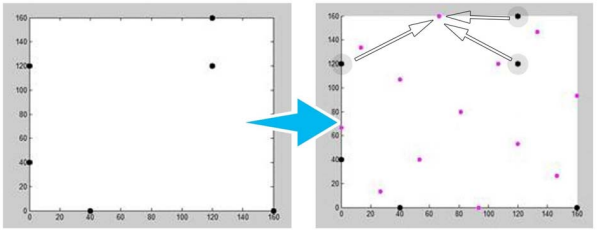

Figure 4. Generate 13 synthetic sensors in the field and selects its K nearest neighboring (K=3) real sensors to interpolate values.

synthetic sensors. Next, every synthetic sensor selects its K nearest neighboring real sensors to interpolate values from.

Finally, we combine values from the real and synthetic sensors to train our model. Due to the synthetic sensor readings being predicted, their errors are larger compared to the real sensors. Therefore, we use different values of ε for ε-SSVR to decide our fault-tolerance. Low ε means that we will adjust model hyperplane for tiny errors in fitting a given data set linearly or nonlinearly, otherwise means we will tolerate large errors. So we give a low ε for real sensors to indicate a low fault-tolerance rate, and give a high ε for synthetic sensors. Integration of synthetic and real sensors improves the performance and monitoring accuracy when there are coverage holes.

V. Distributed Anomaly Detection Mechanism

In this section, we will introduce how to detect anomalies via our distributed anomaly detection. Our distributed anomaly detection is composed of a front-end and back-end mechanism, and can detect anomalous events in real-time, while also providing a visual interface allowing users to understand the scope of the anomalous.

A. Front-End Detection

Front-end anomaly detection proceeds in every individual sensor, whereby it immediately checks for error readings and avoids severe factor changes. However, sensors have small computation ability and storage so they can only use simple models for detection. We propose a front-end anomaly detection method which we call dynamic range checking. This method detect the anomalous portions the new incoming data by comparison with previous records. Dynamic range checking aggregates past data by storing the mean and standard deviation and updating these statistics as new readings arrive. For example, if we assume T = 10, where T is the time series window size, dynamic range checking calculates the mean and standard deviation using the past ten time series data values in each time interval, and updates new readings to replace the oldest reading in the time window. Due to sensors having such a small computation ability the mean and standard deviation (std) formula should have as reduced a computational complexity as possible. In general, the formula for the mean is $\frac{\sum_{i=1}^{T} x_i}{T}$, for the std it is $\frac{\sum_{i=1}^{T}(x_i - mean)}{T}$, where x_i is the reading at time i, $mean$ is the previously calculated mean. The time complexity of the above two calculations is O(T), because all the time windows after the new reading are recalculated. Therefore, we revise the formula for the mean to $\frac{mean*T - x_1 + x_{T+1}}{T}$, and for the standard deviation to $\frac{((mean^2 + std)*T - x_1^2 + x_{T+1}^2}{T} - mean^2$, where std is the previously calculated standard deviation, and x_{T+1} is the new reading. The time complexity is now O(1) after revising the formula, since we only need to store the previous mean and standard deviation. If we assume T is very large, the savings in computational cost will be improve greatly.

Figure 5 shows the anomaly detection procedure by dynamic range checking. The blue line is the sensor reading and the red dashed is 3 times the standard deviation (several times as the case may be decided). Data readings from close intervals are expected to have similar distributions to those from inside the 3 times standard deviation range, so some readings outside the boundary are considered anomalies.

B. Back-End Detection

Our back-end detection is based on the result from continuous monitoring, and runs at the base station therefore computational and storage cost is less of a concern. The goal of the back-end detection is finding dangerous areas and places with changes in environmental factors. During continuous monitoring, we keep the past T maps, where T is the time series window size, and we take the average of those maps. Next, we use the average map to determine the range of the anomaly. After the new map is created, we take the difference between these two maps. If the result has any region higher than the threshold, then we will report an anomaly message.

C. Architecture

Our anomaly detection architecture is summarized in Figure 6. Wireless sensors collect environment readings which are aggregated at the base station. Simultaneously, each sensor stores the past T readings to detect front-end anomalies. Users can get information about anomalies in

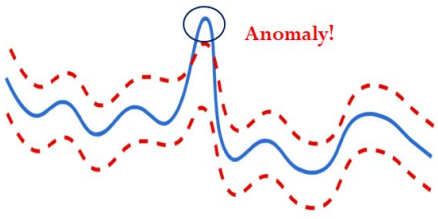

Figure 5. Dynamic range checking. The blue line is the sensor reading and the red dashed is 3 times the standard deviation. Some readings outside the boundary are considered anomalies.

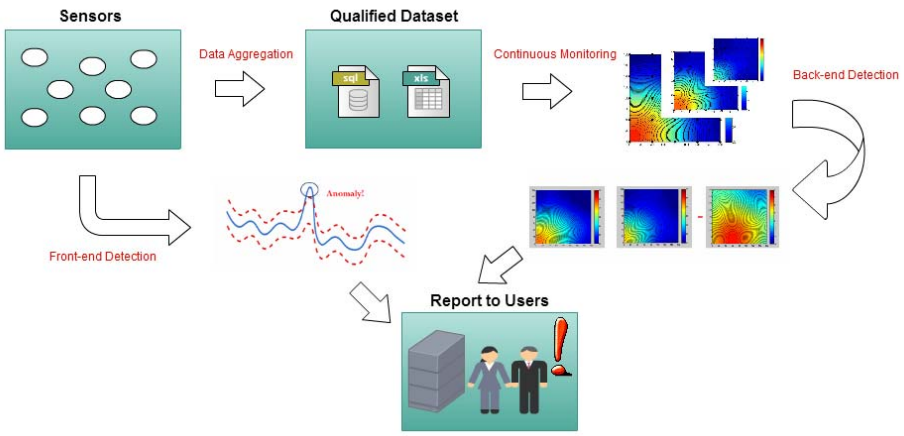

Figure 6. Anomaly detection architecture.

real-time. After data aggregation, we utilizes our continuous monitoring method ε-SSVR to produce a continuous visualization map. Visualization can help the user to understand all fields. Back-end detection utilizes a length T visualization map difference to detect anomalies. Messages will be sent to the user who can decide how to deal with the anomaly problem.

VI. Experiments

A. Dataset Description and Experimental Setting

Our experiments were conducted as part of Intel-NTU's smart agriculture demo. Intel-NTU is a research center whose mission is to explore new M2M (Machine-to-Machine) technologies for the future. A smart agriculture system was set up in a green house in Chiayi which could monitor the growth conditions of orchids. Of the sensor nodes used, a portion were attached to fixed metal beam. These are the fixed nodes. Other sensor nodes were then set up on a mobile plant bed. We call them the mobile nodes because the plant bed can move around the green house.

The smart agriculture demo simulated the operation of an actual real green. 25 sensors were uniformly deployed in a 160x160cm region to collect temperature and humidity data. Uniform distribution is convenient for validation, because removing a portion of the sensor nodes allows us to simulate coverage holes where we can calculate our error rate by comparing the original value and the predicted value. To simulate moving nodes, we can remove different sensors at different times. The 25 sensors are 40cm apart and the data collection period is one minute. An electric heater fan heats the lower left corner of the region to simulate the heat generated by increased sunshine. A heater fan heats diffusively, so each sensor will slowly warm up over time. Simulating an environment with high variability is important due to anomalous conditions being of great concern, and we want to show that our method accurately monitors anomalous temperature information.

We compare our method with Ordinary Kriging(OK) and Inverse Distance Weighting(IDW) in MATLAB. For OK we used the freely available kriging toolbox from Thomas [15] , and we reimplemented IDW. OK has two parameters, range and sill, which need to be set. Range is set to 225 and sill is set to 10 to interpolate all global surfaces. Range is set 120 and sill set 8 to interpolate local surfaces. We wrote two versions of IDW for global and local interpolation. The following values were decided empirically through manual tuning. ε-SSVR also has some default parameters to set. Parameter C is set to 316.22 and parameter γ is set to 0.0000757. C weights the tradeoff between the fitting errors and the flatness of the linear regression function $f(x)$, and γ is a tuning parameter determining how far the influence of a single training example reaches. $\varepsilon 1$ is fault-tolerance for real sensors, and $\varepsilon 2$ is for synthetic sensors. $\varepsilon 1$ is set to 0.01 and $\varepsilon 2$ is set 0.6. Smaller values mean a lower fault-tolerance rate. We used a uniform design pattern with nine points. Parameter k is how many nearest neighbours to use for local interpolation, the default is set to 3. Parameter T is the time window size, used determine how much historical information to use when training our model and is the front-end and back-end anomaly detection window size, set to 5.

B. Experimental Results

As mentioned above, we randomly removed a group of sensors to validate our monitoring result, and calculate MAE and RMSE. MAE formula is $\frac{\sum_{i=1}^{n}|f_i-y_i|}{n}$ and RMSE formula is $\sqrt{\frac{\sum_{i=1}^{n}(f_i-y_i)^2}{n}}$, n is time stamp size 20, f_i is the prediction value and y_i is the true value. We take 20 rounds average for MAE and RMSE. However only using

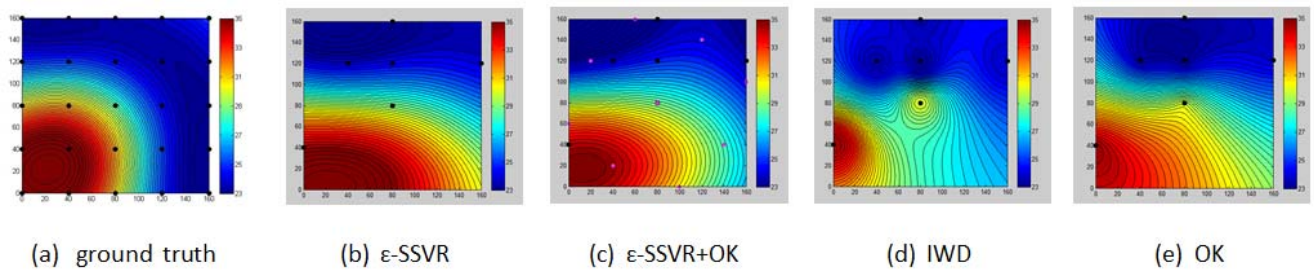

Figure 7. Result of experiment 2. Random deployment of 6 points. (a) is ground truth of 25 points. (b), (c), (d), (e) respectively are ε-SSVR, ε-SSVR+OK, IDW and OK. (c) has a result closest to (a), proving that our method can effectively deal with the problem of coverage holes.

one random deployment to compare the accuracy is not fair, so we repeated this procedure of randomly remove different sensors 100 times. Total calculation amount is 2000 rounds for MAE and RMSE. We also calculate each round's standard deviation to guarantee decreased variation. In setting 1, we want to know the performance of relatively uniform distribution condition, therefore we just randomly removed 10 of the sensors from the 25 total sensors. Table 3 shows the result of our methods, IDW and OK. ε-SSVR(S+T) is considering spatial and temporal correlation, ε-SSVR+OK and ε-SSVR+IDW are interpolating synthetic sensors by Ordinary Kriging and Inverse Distance Weighting. OK gets best monitoring result as measured by MAE and RMSE, but the running time is higher than one minute, which is unacceptable. IDW although has high speed for monitoring but bad accuracy. All our methods are as accurate as OK, and the processing time of each figure is never larger than 3 seconds.

Table III
RESULTS UNDER THE SETTING OF RELATIVELY UNIFORM DISTRIBUTION CONDITION.

Method	MAE	RMSE	STD	CPU sec.
IDW	1.845	2.185	0.317	0.2269
OK	1.103	1.396	0.282	64.1239
ε-SSVR	1.168	1.44	0.276	0.1126
ε-SSVR(S+T)	1.156	1.424	0.279	1.2374
ε-SSVR+OK	1.134	1.396	0.266	0.2233
ε-SSVR+OK(S+T)	1.127	1.385	0.271	2.5673
ε-SSVR+IDW	1.136	1.395	0.271	0.1744
ε-SSVR+IDW(S+T)	1.128	1.385	0.271	1.6626

Next we show the coverage holes case in setting 2. Randomly removing 19 of the 25 sensors, leaves only 6 sensors in the field. Table 4 is the result of experiment 2. ε-SSVR and ε-SSVR(S+T) have the worst accuracy in monitoring, even worse than IDW. ε-SSVR+OK and ε-SSVR+IDW purpose is to improve the coverage holes problem. In the result, their MAE and RMSE is better than others, and can plot a diagram in 0.19 seconds. Figure 7 is the visualization map of experiment 2 in one of the random deployment. (a) is the ground truth of 25 points. (b), (c), (d), (e) respectively are ε-SSVR, ε-SSVR+OK, IDW and OK. ε-SSVR+OK is most similar to the ground truth, proving that our method can effectively deal with the coverage holes problem. Looking at these two experiments, our method not only has better accuracy and computation speed, but is also suitable for any situation.

Table IV
RESULTS UNDER THE SETTING OF COVERAGE HOLES.

Method	MAE	RMSE	STD	CPU sec.
IDW	1.865	2.313	0.405	0.0126
OK	1.665	2.099	0.41	57.8343
ε-SSVR	1.935	2.392	0.912	0.092
ε-SSVR(S+T)	1.933	2.389	0.919	0.75
ε-SSVR+OK	1.563	1.983	0.443	0.19
ε-SSVR+OK(S+T)	1.557	1.979	0.445	2.1479
ε-SSVR+IDW	1.561	1.99	0.472	0.1424
ε-SSVR+IDW(S+T)	1.557	1.986	0.476	1.0125

VII. CONCLUSION AND FUTURE WORK

In this paper, we have introduced a high quality continuous monitoring system based on spatial and temporal dependencies present in sensor data, which are used to address the issue of sparse sensor deployment and coverage holes. Furthermore, we developed front-end and back-end anomaly detection methods based on distributed anomaly detection mechanisms. In our experiments, our approach can get better performance than other interpolation methods not only in terms of monitoring accuracy but also in terms of computation speed.

Our experiments target a small-scale environment. Real world applications often require deploying sensors in a large-scale environment as needed. Therefore, how to monitor a large area via interpolation or prediction is of great importance. Gustavo et al. [17] and Wei et al. [33] proposed distributed kriging method, which make the field is divided into sub-blocks according to trends. Annalisa et al. [3] use distributed IDW. As the scale of the sensor deployment scales up, another issue is the sheer amount of data gathered. Christine et al. [26] use map-reduce structure deal with this sort of big data issue in the domain of wireless sensor

networks. Therefore, how to extend our method in large-scale and deal with big data are an important future work.

ACKNOWLEDGEMENTS

This work was jointly supported by National Science Council, National Taiwan University and Intel Corporation under Grants NSC102- 2911-I-002-001, NTU103R7501-1 and NSC103-2218-E-034 -001.

REFERENCES

[1] Nadeem Ahmed, Salil S. Kanhere, and Sanjay Jha. The holes problem in wireless sensor networks. *SIGMOBILE Mobile Computing and Communications Review*, 9(2):4–18, 2005.

[2] Ian F. Akyildiz, Weilian Su, Yogesh Sankarasubramaniam, and Erdal Cayirci. A survey on sensor networks. *Communications magazine*, 40(8):102–114, 2008.

[3] Annalisa Appice, Anna Ciampi, Donato Malerba, and Pietro Guccione. Using trend clusters for spatiotemporal interpolation of missing data in a sensor network. *Journal of Spatial Information Science*, pages 119–153, 2013.

[4] Annalisa Appice, Pietro Guccione, Donato Malerba, and Anna Ciampi. Dealing with temporal and spatial correlations to classify outliers in geophysical data streams. *Information Sciences*, 2013.

[5] Zoran S. Bojkovic, Bojan M. Bakmaz, and Miodrag R. Bakmaz. Security issues in wireless sensor networks. *International Journal of Communications*, 2(1), 2008.

[6] Joel W. Branch, Chris Giannella, Boleslaw Szymanski, Ran Wolff, and Hillol Kargupta. In-network outlier detection in wireless sensor networks. *Knowledge and Information Systems*, 34(1):23–54, 2013.

[7] Alberto Camilli, Carlos E. Cugnasca, Antonio M. Saraiva, Andre R. Hirakawa Alberto Camilli, Carlos E. CugnascaAndrie R. Hirakawa, and Pedro L.P. Correa. From wireless sensors to field mapping: Anatomy of an application for precision agriculture. *Computers and Electronics in Agriculture*, 58(1):25–36, 2007.

[8] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection : A survey. *ACM Computing Surveys*, 41(3), 2009.

[9] Xiangqian Chen, Kia Makki, Kang Yen, and N. Pissinou. Sensor network security: a survey. *IEEE. Communications Surveys & Tutorials*, 11:52–73, 2009.

[10] Franke Conny and Gertz Michael. Orden: Outlier region detection and exploration in sensor networks. *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data*, 4:1075–1078, 2009.

[11] Wenliang Du and Lei Fang. Lad: Localization anomaly detection for wireless sensor networks. In *Proceedings of the 19th IEEE International Parallel & Distributed Processing Symposium*, volume 66, pages 874–886, 2006.

[12] V. Chatzigiannakis; S. Papavassiliou; M. Grammatikou; and B. Maglaris. Hierarchical anomaly detection in distributed large-scale sensor networks. In *Proceedings. 11th IEEE Symposium on*, pages 761–767, 2006.

[13] Uwe Haberlandt. Geostatistical interpolation of hourly precipitation from rain gauges and radar for a large-scale extreme rainfall event. *Journal of Hydrology*, 332(1-2):144–157, 2007.

[14] Mohammad Hammoudeh, Robert Newman, Christopher Dennett, and Sarah Mount. Interpolation techniques for building a continuous map from discrete wireless sensor network data. *Wireless Communication and Mobile Computing*, 13(9):809–827, 2013.

[15] Thomas Mejer Hansen. mgstat: A geostatistical matlab toolbox [online], http://mgstat.sourceforge.net/. 2004.

[16] Tomislav Hengl, Gerard B.M. Heuvelink, and Alfred Stein. A generic framework for spatial prediction of soil variables based on regression-kriging. *Geoderma*, 120(1-2):75–93, 2004.

[17] Gustavo Hernandez-Penaloza and Baltasar Beferull-Lozano. Field estimation in wireless sensor networks using distributed kriging. In *Communications, 2012 IEEE International Conference on*, pages 724–729, 2012.

[18] R. Jedermann, J. Palafox-Albarran, J. Robla, P. Barreiro, L. Ruiz-Garcia, and W. Lang. Interpolation of spatial temperature profiles by sensor networks. *in Sensors, 2011 IEEE*, pages 778–781, 2011.

[19] Dietmar Maringer Kai-Tai Fang, Chang-Xing Ma, Yu Tang, and Peter Winker. Uniform design table [online], http://sites.stat.psu.edu/ rli/dmce/uniformdesign/.

[20] Muhammad Umer; Lars Kulik; and Egemen Tanin. Kriging for localized spatial interpolation in sensor networks. In *Proc. 20th Scientific and Statistical Database Management*, volume 5069, pages 525–532, 2008.

[21] Sutharshan Rajasegarar; Christopher Leckie; and Marimuthu Palaniswami. Distributed anomaly detection in wireless sensor networks. In *10th IEEE Singapore International Conference on communication systems*, pages 1–5, 2006.

[22] Yuh-Jye Lee, Wen-Feng Hsieh, and Chien-Ming Huang. Epsilon-ssvr: A smooth support vector machine for epsilon-insensitive regression. *Knowledge and Data Engineering, IEEE Transactions on*, 17(5):678–685, 2005.

[23] George Y. Lu and David W. Wong. An adaptive inverse-distance weighting spatial interpolation technique. *Computers & Geosciences*, 34(9):1044–1055, 2008.

[24] Th. Arampatzis; J. Lygeros; and S. Manesis. A survey of applications of wireless sensors and wireless sensor networks. In *Proc. 13th IEEE Mediterrean Conf. Control and Automation.*, pages 719–724, 2005.

[25] H. C. Ma, P. K. Sahoo, and Y. W. Chen. Computational geometry based distributed coverage hole detection protocol for the wireless sensor networks. *Journal of Network and Computer Applications*, 34(5):1743–1756, 2011.

[26] Christine Jardak; Janne Riihijarvi; Frank Oldewurtel; and Petri Mahonen. Parallel processing of data from very large-scale wireless sensor networks. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, volume 8, pages 787–794, 2010.

[27] Adrian Perrig, John Stankovic, and David Wagner. Security in wireless sensor networks. *Communications of the ACM*, 47(6):53–57, 2004.

[28] H. Redwan and Ki-Hyung Kim. Survey of security requirements, attacks and network integration in wireless mesh networks. In *Frontier of Computer Science and Technology*, Japan-China Joint Workshop on, 2008.

[29] Alan Mainwaring; David Culler; Joseph Polastre; Robert Szewczyk; and John Anderson. Wireless sensor networks for habitat monitoring. In *Proc. 1st ACM international workshop on Wireless sensor networks and applications.*, pages 88–97, 2002.

[30] Muhammad Umer, Lars Kulik, and Egemen Tanin. Spatial interpolation in wireless sensor networks: localized algorithms for variogram modeling and kriging. *Geoinformatica*, 14(1):101–134, 2010.

[31] Mehmet C. Vuran, Ozgur B. Akan, and Ian F. Akyildiz. Spatio-temporal correlation: theory and applications for wireless sensor networks. *Computer Networks*, 45(3):245–259, 2004.

[32] W. T. Wang and K. F. Ssu. Obstacle detection and estimation in wireless sensor networks. *Computer Networks*, 57(4):858–868, 2012.

[33] C.; Wei Zhuo; Prabhat; Paciorek and C Kaufman. Parallel kriging analysis for large spatial datasets. In *2011 11th IEEE International Conference on Data Mining Workshops*, pages 38–44, 2011.

[34] Miao Xie, Song Han, Biming Tian, and Sazia Parvin. Anomaly detection in wireless sensor networks: A survey. *Journal of Network and Computer Applications*, 34(4):1302–1325, 2011.

[35] Yunfeng Xie, Tong bin Chen, Mei Lei, Jun Yang, Qing jun Guo, Bo Song, and Xiao yong Zhou. Spatial distribution of soil heavy metal pollution estimated by different interpolation methods: Accuracy and uncertainty analysis. *Chemosphere*, 82(3):468–476, 2011.

[36] Jafar Yasrebi, Mahboub Saffari, Hamed Fathi, Najafali Karimian, Masome Moazallahi, and Reza Gazni. Evaluation and comparison of ordinary kriging and inverse distance weighting methods for prediction of spatial variability of some soil chemical parameters. *Research Journal of Biological Sciences*, 4(1):93–102, 2009.

[37] Dale Zimmerman, Claire Pavlik, Amy Ruggles, and Marc P. Armstrong. An experimental comparison of ordinary and universal kriging and inverse distance weighting. *Mathematical Geology*, 31(4):375–390, 1999.