Blind Drift Calibration of Sensor Networks Using Sparse Bayesian Learning

Yuzhi Wang, *Student Member, IEEE*, Anqi Yang, Zhan Li, Xiaoming Chen, *Member, IEEE*, Pengjun Wang, and Huazhong Yang, *Senior Member, IEEE*

***Abstract*— The lifetime of wireless sensor networks (WSNs) has been significantly extended, while in long-term large-scale WSN applications, the increasing sensor drift has become a key problem affecting the reliability of sensory data. In this paper, we propose a blind online drift calibration framework based on subspace projection and sparse recovery for sensor networks in general-purpose monitoring. Temporal sparse Bayesian learning is used in the proposed method to estimate the sensor drift from under-sampled observations. The proposed method needs neither dense deployment nor the presence of a prior data model. Both simulated and real-world data set are used to evaluate the proposed method. Experimental results demonstrate that the proposed method can detect and recover the sensor drift when the number of drifted sensors are less than 20%, and when 40% sensors are drifted, the recovery rate is 80%.**

***Index Terms*— Wireless sensor networks, blind calibration, compressed sensing, temporal sparse Bayesian learning.**

I. Introduction

Wireless sensor networks (WSNs) are important in many emerging applications such as smart cities [1], industrial controlling [2], home automation [3], etc. In these applications, a number of sensors with the ability of wireless communication are deployed to the area of interest to collect necessary data from the physical environment. Typically, sensors pass the measured data to a central node or a server. Depending on the application requirements, the collected data can be utilized for either decision making or environment monitoring.

In recent years, there has been a significant technology advance in WSNs and related research areas. For hardware, the widespread use of micro-electromechanical systems (MEMS) and System-on-Chip (SOC) technologies promoted many vendors to provide low-power and low-cost sensor nodes with sufficient computation and communication performance. For network, protocols like 6LoWPAN [4], CoAP [5] were standardized and implemented in open-source WSN softwares such as TinyOS [6] and Contiki [7]. With the help of a mature WSN ecosystem, large-scale and long-term WSNs are possible to be deployed at an acceptable cost. In our previous works [8], [9], we deployed hundreds of sensor nodes in different applications including environment monitoring and structural health monitoring.

Meanwhile, sensor drift has become a new problem as temporal drift inevitably increases with time. Although in many applications, sensors are calibrated before deployment [10], it cannot prevent sensor drift after deployment, especially when the lifetime of WSNs can be as long as years. In some cases, even costly high-accuracy sensors still produce faulty data [11]. Therefore, it is necessary to correct sensors' measurement after deployment to ensure the trustworthiness of large-scale and long-term WSNs. However, large-scale WSNs often consist of hundreds to thousands of sensor nodes deployed in nearly inaccesible locations under harsh environment, leading to the infeasibility of unmounting and re-calibrating these sensors individually.

One possible solution is to calibrate sensors using a prior model. In some applications, a prior data and error model can be employed. By using sensor fusion algorithms, the uncertainty of parameters in the data model can be reduced. Thus, the final sensory results are calibrated. This fusion-based calibration scheme is popular in target discovery applications [12], [13], but not common in other applications due to the lack of a prior model. Another calibration scheme is called *bind calibration*, which calibrates sensors by utilizing measurement redundancy. Many existing blind calibration approaches [14]–[16] require dense deployment, where neighbour sensors are assumed to share the same ground-truth measurement. Blind calibration is also available in mobile sensor networks, as sensors sometimes move to the same location, so that they collaboratively calibrate each other [17], [18]. Unfortunately, in many applications such as environment monitoring, neither dense deployment nor sensor mobility is easy to satisfy.

In this paper, we propose an online blind drift calibration algorithm based on subspace projection and sparse recovery which can be applied in the sensor networks for general-purpose monitoring. The calibration process is done in the data-center and transparent to sensors. We model the data redundancy as *signal subspace*, which was first proposed by Balzano and Nowak [19]. For a sensor network monitoring

Manuscript received April 28, 2016; accepted June 9, 2016. Date of publication June 20, 2016; date of current version July 18, 2016. This work was supported by the National Natural Science Foundation of China under Grant 61271269 and Grant 61321061. The associate editor coordinating the review of this paper and approving it for publication was Prof. Ravinder S. Dahiya.

Y. Wang, A. Yang, Z. Li, P. Wang, and H. Yang are with the Department of Electronic Engineering, Tsinghua University, Beijing 100084, China (e-mail: yz-wang12@mails.tsinghua.edu.cn; yang-aq14@mails.tsinghua.edu.cn; lizhan13@mails.tsinghua.edu.cn; wangpj@tsinghua.edu.cn; yanghz@tsinghua.edu.cn).

X. Chen is with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: xchen3@andrew.cmu.edu).

Digital Object Identifier 10.1109/JSEN.2016.2582539

the physical environment, if a sufficient number of sensors are deployed, the sensing field is oversampled, so the ground-truth measurement lies in a signal subspace determined by physical conditions. By projecting measurements to the orthogonal complement of the signal subspace, the signal and drift component of sensor measurements can be separated, and an undersampled observation of the drift can be obtained. We assume that a sparse set of sensors have drift, therefore the sensor drift can be recovered from the under-sampled observation using compressed sensing algorithms. In this way, sensors are calibrated.

The main contributions of this paper are listed as follows.

- We propose an online blind calibration framework based on subspace projection and sparse recovery (SPSR) to estimate sensor drift without the requirement of dense deployment or the knowledge of the data model, so it can be applied in general-purpose monitoring applications.
- To our knowledge, this is the first work that models the sensor calibration problem as sparse signal recovery and applies compressed sensing to estimate sensor drift.
- The proposed method calibrates sensors using undersampled observations of the actual sensor drift, and calibration error does not accumulate. While in existing prediction-based methods, once the predicted value become inaccurate, it will be taken as the true value to further predict other sensors' measurements, leading to the error accumulation. Experiments show that the proposed method is more robust than previous prediction-based blind calibration algorithms.
- We design and deploy a sensor network testbed reporting almost drift free temperature measurements, so the dataset can be used to benchmark different online calibration algorithms.
- We compare the performances of the proposed method under different simulated sensing fields and real-world datasets. We also analyze the cause of different performance, and give some guidelines for planning the deployment of a sensor network.

The rest of this paper is organized as follows. We first present some related works in Section II. In Section III, we formulate the drift calibration problem. In Section IV, we describe the proposed two-phase calibration algorithm. In Section V, the proposed algorithm is evaluated and compared with existing algorithms using both simulated and real-world datasets. Possible future work is discussed in Section VI. We conclude in Section VII.

II. Related Work

Sensor calibration is a fundamental problem of sensor networks. Many research works have studied this topic from different perspectives.

In some applications such as target detection, sensory data has a prior model or probability distribution, and the measurement error is a parameter in the model. By combining the measurement data from multiple sensors, the error parameter can be estimated using sensor fusion algorithms. In [12], Tan et al. proposed a system-level two-tier calibration method for fusion-based WSNs. It could balance computation overhead among sensor nodes with little transmission overhead. In [13], Xiang et al. modeled the pollution source detection as a maximum likehood estimation problem, and proposed an iterative algorithm inspired by expectation maximization (EM) to estimate parameters.

However, in many applications, such as general purpose monitoring systems, the data model is either unavailable or too expensive to build. Therefore, blind calibration is more applicable in these kinds of applications.

Many blind calibration schemes rely on dense deployment or node mobility. In [14], Bychkovskiy et al. proposed a collaborative calibration approach by utilizing data correlation among neighboured sensors. In [15], Lee et al. used a graph model to describe dense mobile sensor networks and calibrate the sensors by solving a Laplacian linear equation. Takruri et al. [16], [20] proposed the "drift-aware WSN" and a drift calibration algorithm using Kalman filter for dense sensor network. In mobile sensor networks, sensors can calibrate each other when they sometimes move to the same location [17], [18].

A more general case is that sensors are not densely deployed but still correlated. Balzano and Nowark [19] first proposed a calibration method without the requirement of dense deployment. The proposed error model includes gain and offset, and the ground-truth measurement is assumed to lie in a lower dimensional *signal subspace* of the measurement space. An orthorgonal projection matrix to the orthogonal complement of the signal subspace could be utilized to set up a group of equations of error parameters. The gain parameter can be recovered while estimating the offset needs extra assumptions.

Prediction-based drift tracking was first proposed by Takruri et al. [16], [21]. For each sensor, the neighbours' measurement can be used to predict its ground-truth so that the drift can be estimated, and then Kalman filter is used to track sensor drift. The first work in [16] used the mean measurement of neighbour sensors as the prediction, so it could only be used in dense networks. In [21], the authors used support vector regression (SVR) to predict the ground truth measurements so that the proposed method could be applied in non-dense sensor networks. Kumar et al. [22], [23] employed Kriging interpolation as prediction method so the training phase of SVR-KF could be avoided. However, once the predicted ground-truth become inaccurate, the erroneous prediction will be taken as the true value to further predict other sensors' measurement. As as result, the accuracy of prediction-based methods are limited by the accumulative prediction error.

In our previous work [24], we modeled the sensor error as a time variable drift, and employed the concept of signal subspace introduced by Balzano and Nowak to obtain an observation of the sensor drift. To calibrate the drift, we took the sensor drift as the system state and used Kalman filter to track the drift. Unlike prediction-based approaches, the system state is a lower-dimensional observation, instead of a prediction of the sensor drift, so it has better accuracy and robustness. The shortcoming of this approach is that only one drifted sensor can be detected due to the brute-force-based detection method.

In this paper, we use the temporal sparse Bayesian learning (T-SBL) algorithm proposed by Zhang and Rao [25] to solve the drift observation equation, so that multiple drifting sensors can be detected and calibrated.

III. Problem Formulation

Consider a sensor network with n sensors. At each time instant t, let $\boldsymbol{x}_t = [x_{1,t}, x_{2,t}, \ldots, x_{n,t}]^T$ be the ground-truth signal supposed to be measured by the sensors, where $x_{i,t}$ represents the signal value of sensor i at time instant t with no drift and noise. Obviously, $\boldsymbol{x}$ lies in an n-dimension *measurement space*, denoted as $\mathcal{M}$.

Let y denote the actual sensor measurement. Since each sensor has unknown drift and noise, we assume $y_{i,t} = x_{i,t} + d_{i,t} + v_{i,t}$, where $y_{i,t}$ is the sensor reading, $x_{i,t}$ is the unknown ground-truth, $d_{i,t}$ is the unknown sensor drift and $v_{i,t}$ is measurement noise. This can be represented using vector notation

$$\boldsymbol{y}_t = \boldsymbol{x}_t + \boldsymbol{d}_t + \boldsymbol{v}_t. \tag{1}$$

The blind drift calibration problem is to recover the unknown ground-truth value $\boldsymbol{x}$ from drifted and noisy measurement $\boldsymbol{y}$.

Among the terms of Eq. (1), only $\boldsymbol{y}_t$ can be measured, so extra constraints are needed for blind recovering. Since sensor networks often consist of a number of sensors monitoring limited environmental conditions, it is reasonable to assume that the sensors oversample the physical environment. We assume that the ground-truth signal lies in a lower dimensional *signal subspace* of $\mathcal{M}$, denoted as $\mathcal{S}$. Let's say the dimension of $\mathcal{S}$ is r, and $r < n$.

One typical example of the signal subspace is that the sensing field is spatially bandwidth-limited [19], where $\boldsymbol{x}_t$ can be represented by a linear combination of a limited number of Discrete Fourier Transform (DFT) vectors, so the signal subspace is spanned by this group of DFT vectors.

Another example is the linear source-sensor model. Given a sensing field which is driven by a group of signal sources: $\boldsymbol{s} = [s_1, s_2, \ldots, s_r]^T$, and the signal at each sensor's position is a linear combination of signal sources, then $x_i = \sum_{j=1}^{r} a_{ij} s_j$, where a_{ij} is the contribution of source j to sensor i. We rewrite this model in the vector notation

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{s}$$

where the element of i-th row and j-th column of $n \times r$ matrix $\boldsymbol{A}$ is a_{ij}. Obviously, the ground-truth signal vector $\boldsymbol{x}$ lies in the r-dimensional column space of matrix $\boldsymbol{A}$. In this example, the signal space $\mathcal{S} = \text{col}(\boldsymbol{A})$. This model is also adopted by [26] and [27] to describe temperature sensors and light sensors, respectively.

For non-linear systems, kernel methods [28] can be employed to map the non-linear signal subspace to a higher-dimensional linear subspace.

Let $\mathcal{S}^\perp \overset{\text{def}}{=} \{\boldsymbol{v} \in \mathcal{M} | \forall \boldsymbol{u} \in \mathcal{S}, \langle \boldsymbol{u}, \boldsymbol{v} \rangle = 0\}$ denote the orthogonal complement of the signal subspace $\mathcal{S}$, and $\mathcal{S}^\perp$ is a p-dimensional subspace of $\mathcal{M}$, where $p = n - r$. Let $\{\boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_p\}$ denote a set of orthonormal basis of $\mathcal{S}^\perp$, so for any ground-truth sensor measurement $\boldsymbol{x}_t \in \mathcal{S}$,

$$\boldsymbol{\phi}_i^T \boldsymbol{x}_t = 0 \quad i = 1, 2, \ldots, p. \tag{2}$$

Let $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_p]^T$ denote a $p \times n$ matrix, each row of which is a basis vector of $\mathcal{S}^\perp$, so

$$\boldsymbol{\Phi}\boldsymbol{x}_t = \boldsymbol{\Phi}(\boldsymbol{y}_t - \boldsymbol{d}_t - \boldsymbol{v}_t) = \boldsymbol{0}. \tag{3}$$

From Eq. (3) we can get,

$$\boldsymbol{\Phi}\boldsymbol{y}_t = \boldsymbol{\Phi}\boldsymbol{d}_t + \boldsymbol{v}_t \tag{4}$$

where $\boldsymbol{v}_t = \boldsymbol{\Phi}\boldsymbol{v}_t$.

Eq. (4) means that for each snapshot of drifted sensor measurement $\boldsymbol{y}_t$, $\boldsymbol{\Phi}\boldsymbol{y}_t$ is mostly driven by sensor drift. We name $\boldsymbol{\Phi}$ as the *observation matrix*, and assume that $\boldsymbol{\Phi}$ is known, then we can get a p-dimensional observation of sensor drifts.

Let $\boldsymbol{z}_t = \boldsymbol{\Phi}\boldsymbol{y}_t$, by substituting it into Eq. (4), we obtain

$$\boldsymbol{z}_t = \boldsymbol{\Phi}\boldsymbol{d}_t + \boldsymbol{v}_t \tag{5}$$

where $\boldsymbol{\Phi} \in \mathbb{R}^{p \times n} (p < n)$ is the observation matrix, $\boldsymbol{d}_t$ is the unknown sensor drift, $\boldsymbol{v}_t$ is the random noise and $\boldsymbol{z}_t$ is a transformation of sensor measurement.

Hence, we convert the problem of sensor drift calibration into two subproblems: a) how to construct the observation matrix $\boldsymbol{\Phi}$; and b) how to estimate drift vector $\boldsymbol{d}_t$ from the under determined linear system represented by Eq. (5).

IV. Drift Calibration Method

A. Assumptions and Overview

As was discussed in section III, by assuming that the ground-truth of sensor measurements are in a lower dimensional signal subspace to the measurement space, we convert the drift calibration problem into constructing and solving the linear system represented by Eq. (5). The proposed drift calibration method has two phases, *learning phase* and *calibration phase*.

We assume that sensors are calibrated before deployment, so we can infer that: a) within a short period after sensors are deployed, the drift should be zero or insignificant; b) within a reasonably long period, a few (less than p) sensors are drifted. We further assume that the signal subspace is time invariant. This assumption is appropriate in many applications, although the signal value changes a lot, the signal subspace is decided by the environmental structure and sensors' geographic locations, which change very slowly over time.

In the learning phase, using the sensor data with a near-zero drift collected during a short period after deployment, we use principal component analysis (PCA) to extract the orthonormal basis of the signal subspace and its orthogonal complement, and then the drift observation matrix can be generated using these basis vectors.

In the calibration phase, since the signal space is assumed to be time invariant, Eq. (5) holds, while a small number of sensors are drifted. This means that the drift vector $\boldsymbol{d}_t$ should be sparse. Therefore, we can estimate sensor drift accurately with a high probability using Compressed Sensing (CS) [29]. By subtracting sensor reading and the estimated drift, sensors can be calibrated online.

B. Learning Observation Matrix

The observation matrix is composed of the basis vectors of the orthogonal complement to the signal subspace. There are several ways to obtain these vectors.

For example, Balzano and Nowak [19] proposed the "bandwidth-limited subspace", in which the sensor data over space $\boldsymbol{x}_{\cdot,t}$ is bandwidth-limited in frequency domain. Considering that the spacial sample rate (deployment density) is limited, the measurements can be represented by a limited number of Discrete Fourier Transform (DFT) vectors. Therefore, the bandwidth-limited ground-truth signal can be represented by a subset of the DFT vectors, while the reset of the DFT vectors compose the basis of the orthogonal complement subspace.

In this paper, we utilize the sensor data collected within a short period after deployment, to learn the features of the signal subspace, and further construct the observation matrix. As the ground-truth signal $\boldsymbol{x}$ lies in the signal subspace $\mathcal{S}$, a number of drift-free samples $\{\boldsymbol{x}\}_l$ can be considered to span the signal subspace, and we can extract a set of orthonormal basis from $\{\boldsymbol{x}\}_l$ using PCA.

There are several reasons for us to use PCA:

- PCA is an unsupervised method and can extract enough features only using a set of signal samples without any prior knowledge. This is important, since in many WSN applications, prior knowledge about the signal subspace is difficult to obtain.
- Let $\boldsymbol{x}_p$ be the orthogonal projection of sensor measurement onto the estimated signal subspace. PCA can minimize the mean squared error $\|\boldsymbol{x}-\boldsymbol{x}_p\|^2$ [30]. This means that at an acceptable signal-noise-ratio (SNR), PCA can construct a best matching subspace of drift-free samples.

We use singular value decomposition (SVD) to implement PCA. Let $\boldsymbol{X} = [\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_\tau]$, where $\boldsymbol{x}_i$ is the drift-free sensor measurement at time instant i and $\boldsymbol{X} \in \mathbb{R}^{n\times\tau}$. We calculate SVD of $\boldsymbol{X}$ by

$$\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^* = \boldsymbol{X} \tag{6}$$

where $\boldsymbol{U} \in \mathbb{R}^{n\times n}$ is the left-singular matrix, whose columns are a group of orthonormal basis vectors of the measurement space; $\boldsymbol{\Sigma} \in \mathbb{R}^{n\times\tau}$ is a diagonal matrix composed of singular values, and $\boldsymbol{V}$ is the right-singular matrix.

To estimate the dimension of the signal subspace, we can find the $\hat{r}$ that satisfies

$$\sum_{e\in\Sigma,i=0}^{\hat{r}} e_{i,i} \geq T_r \cdot \sum_{e\in\Sigma} e \tag{7}$$

where T_r is the threshold value, and in practice, 0.99 is a proper number.

The first $\hat{r}$ columns of $\boldsymbol{U}$ are the orthonormal basis vectors of the estimated signal subspace, and the last $n-\hat{r}$ columns are the orthonomal basis vectors of its orthogonal complement. Therefore, the observation matrix can be constructed with the last $n-\hat{r}$ columns of $\boldsymbol{U}$:

$$\boldsymbol{\Phi} = [\boldsymbol{u}_{\cdot n-\hat{r}+1}, \ldots, \boldsymbol{u}_{\cdot n-1}, \boldsymbol{u}_{\cdot n}]^T \tag{8}$$

where $\boldsymbol{u}_{\cdot i}$ denotes the i-th column of $\boldsymbol{U}$.

C. Drift Model

Before we introduce the drift estimation algorithm, we describe the prior probabilistic model of the sensor drift.

Assuming that different sensors' drifts are independent, and the increment of each sensor's drift at each time instant is independent and Gaussian:

$$d_{i,t} = d_{i,t-1} + \delta_{i,t} \quad \delta_{i,t} \sim \mathcal{N}(0, \sigma_{i,t}^2) \tag{9}$$

where $d_{i,t}$ is the drift value, and $\delta_{i,t}$ is the increment of sensor-i's drift at time instant t.

Since the proposed algorithm is a general-purpose calibration algorithm, zero-mean Gaussian distribution is a suitable assumption of the increment of sensors' drifts.

According to our assumption that the sensors are calibrated before deployment, at beginning the drift value should be zero:

$$d_{i,0} = 0. \tag{10}$$

At time instant t, the drifts are the sum of a series of Gaussian increments. Therefore, the accumulated drifts are still Gaussian [31]. The expectation of sensor drift is:

$$E(d_{i,t}) = E(\sum_{\tau=1}^{t}\delta_{i,\tau}) = \sum_{\tau=1}^{t} E(\delta_{i,\tau}) = 0 \tag{11}$$

and the variance is:

$$\begin{aligned}\mathrm{var}(d_{i,t}) &= \mathrm{cov}(\sum_{\tau=1}^{t}\delta_{i,\tau}, \sum_{\tau=1}^{t}\delta_{i,\tau}) \\ &= \sum_{j=1}^{t}\sum_{k=1}^{t}\mathrm{cov}(\delta_{i,j}, \delta_{i,k}) = \sum_{\tau=1}^{t}\sigma_{i,\tau}^2 = \Sigma_{i,t}^2\end{aligned} \tag{12}$$

Eq. (12) shows that the variance of the sensor drift is the accumulation of the variance of each incremental drift, so it increases with time.

In summary, the probabilistic distribution of sensor drift can be assumed as a Gaussian distribution.

$$\boldsymbol{d}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}_t) \tag{13}$$

where $\boldsymbol{d}_t$ is the sensors' drift vector at time instant t, and $\boldsymbol{\Sigma}_t$ is a diagonal covariance matrix, which is given by

$$\boldsymbol{\Sigma}_t = \begin{bmatrix} \Sigma_{1,t}^2 & & \\ & \ddots & \\ & & \Sigma_{n,t}^2 \end{bmatrix} \tag{14}$$

where $\Sigma_{i,t}^2$ is the variance of sensor-i's drift at time instant t.

D. Drift Estimation

Recalling Eq. (5) discussed in Section III, at each time instant t, by applying the observation matrix $\boldsymbol{\Phi}$ to sensor measurement $\boldsymbol{y}_t$, we get a lower dimensional observation $\boldsymbol{z}_t$ of sensor drift $\boldsymbol{d}_t$. As we assume that only a few sensors have unknown drift, sparse signal recovery, or compressed sensing methods, can be employed to estimate the sensor drift.

In most applications, sensors drift slowly, which means that multiple adjacent measurements could share the same set of sensors who has non-zero drift. This property is called *common sparsity*. Hence, our model can be extended to a multiple measurement vector (MMV) form:

$$Z_t = \Phi D_t + V_t \tag{15}$$

where $Z_t = [z_{t-L+1}, \ldots, z_t] \in \mathbb{R}^{p\times L}$ is consisted of L observation vectors, $D_t = [d_{t-L+1}, \ldots, d_t] \in \mathbb{R}^{n\times L}$ is drift matrix and V_t is noise matrix. In D_t, only the rows corresponding to the drifted sensors has non-zero values.

Previous work has shown that the recovery rate of sparse signal recoverying can be significantly improved using multiple measurements [25], [32], [33]. Among various compressed sensing algorithms, a sparse Bayesian learning (SBL) [34] based method proposed by Zhang and Rao named T-SBL outpeforms other algorithms with a higher recovery rate and lower error [25], [35] by taking advantage of temporal correlation of the unknown source vectors.

In the drift estimation problem, the drift of each sensor is a random process, and the drift within a short period of time is temporally correlated. Therefore, T-SBL is a suitable algorithm to solve sensor drift from low-rank observations.

First, we rewrite Eq. (15) to a blocked vector form. Let $z_t^L = \mathrm{vec}(Z_t^T) \in \mathbb{R}^{pL\times 1}$, $d_t^L = \mathrm{vec}(D_t^T) \in \mathbb{R}^{nL\times 1}$, $v_t^L = \mathrm{vec}(V_t^T) \in \mathbb{R}^{pL\times 1}$, and $\Phi_L = \Phi \otimes I_L \in \mathbb{R}^{pL\times nL}$, where I_L is a $L \times L$ unit matrix, $\otimes$ represents the Kronecker product operator, and vec($\cdot$) is the vectorization operator which is defined by

$$\begin{aligned}\mathrm{vec}(D_t^T) &= \mathrm{vec}\left(\begin{bmatrix} d_{1,t-L+1} & \cdots & d_{n,t-L+1} \\ d_{1,t-L+2} & \cdots & d_{n,t-L+2} \\ \vdots & \ddots & \vdots \\ d_{1,t} & \cdots & d_{n,t}\end{bmatrix}\right) \\ &= [\underbrace{d_{1,t-L+1}, \ldots d_{1,t}}_{D_{1\cdot,t}}, \ldots, \underbrace{d_{1,t-L+1}, \ldots d_{1,t}}_{D_{n\cdot,t}}]^T\end{aligned}$$

where $d_{i,t}$ represents drift value of sensor i at time instant t, and $D_{i\cdot,t}$ is the i-th row of D_t, or the drift vector of sensor i from time instant $t-L+1$ to t. Therefore, the vectorized form of the MMV model is

$$z_t^L = \Phi_L d_t^L + v_t^L. \tag{16}$$

As we assume in section IV-C that each sensor's drift is Gaussian, so the the prior for d_t^L can be written as:

$$d_t^L \sim \mathcal{N}(\mathbf{0}, \Sigma_0) \tag{17}$$

where Σ_0 is a block-diagonal covariance matrix given by

$$\Sigma_0 = \begin{bmatrix} \gamma_1 B_{1,t} & & \\ & \ddots & \\ & & \gamma_n B_{n,t}\end{bmatrix} \tag{18}$$

where γ_i is a hyperparameter controlling whether sensor i has non-zero drift, and $B_{i,t}$ represents the correlation matrix of drift of sensor i over time. Compared with Eq. (14), d_t^L is a blocked vector where each block represents temporally correlated drift values of one sensor at multiple time instants, while different blocks reprensent different sensors' independent drifts, so the covariance matrix of d_t^L should be block-diagonal.

We assume that each element of the noise vector v_t^L is i.i.d and follows a Gaussian distribution:

$$v_t^L \sim \mathcal{N}(0, \lambda I) \tag{19}$$

where λ is the variance of noise distribution.

Combining Eq. (16) and (19), the likehood of observation z_t^L given drift d_t^L, i.e. the probabilistic form of Eq. (16) can be written as:

$$p(z_t^L | d_t^L) = \mathcal{N}(\Phi_L D_t, \lambda I). \tag{20}$$

The posterior distribution of d_t^L can be obtained using the Bayes rule. According to Tipping [34], the posterior distribution of the sensor drift is still Gaussian, which is

$$\begin{aligned} p(d_t^L | z_t^L) &= \frac{p(z^L | d^L) p(d^L)}{p(z^L)} \\ &= \mathcal{N}(\mu_t, \Sigma_{d,t}) \end{aligned} \tag{21}$$

where the mean and covariance matrix are [25]:

$$\mu_t = \Sigma_0 \Phi_L^T (\lambda I + \Phi_L \Sigma_0 \Phi_L^T)^{-1} z_t^L \tag{22}$$

$$\Sigma_{d,t} = \Sigma_0 - \Sigma_0 \Phi_L^T (\lambda I + \Phi_L \Sigma_0 \Phi_L^T)^{-1} \Phi_L \Sigma_0. \tag{23}$$

The maximum-a-posteriori (MAP) estimation of the sensor drift is the value that the posterior pdf of the sensor drift, i.e. $p(d_t^L | z_t^L)$, get maximized. Since d^L given z^L is Gaussian, the MAP estimation of d^L should be μ_t, i.e.

$$\hat{d}_t^L = \mathrm{argmax}\, p(d_t^L | z_t^L) = \mu_t. \tag{24}$$

Next, the hyperparameters $\Theta = \{\lambda, \gamma_i, B_{i,t}\}$ need to be estimated using evidence maximization [34]. Here, the "evidence" refers to the observation z_t^L, and the hyperparameters should be

$$\Theta = \mathrm{argmax} \log p(z_t^L; \Theta). \tag{25}$$

This can be solved by using expectation maximization (EM) [36], which treats d_t^L as hidden variables and maximizes

$$\begin{aligned} Q(\Theta) &= E[\log p(z_t^L, d_t^L; \Theta)] \\ &= E[\log p(z_t^L | d_t^L; \Theta)] + E[\log p(d_t^L; \Theta)]. \end{aligned} \tag{26}$$

The detailed derivation process can be found in [25]. By maximizing $Q(\Theta)$, the hyperparameters Θ, including λ, γ_i and $B_{i,t}$, can be estimated iteratively as follows:

$$\lambda = \frac{\|z_t^L - \Phi_L \mu_t\|_2^2 + \lambda[pL - \mathrm{Tr}(\Sigma_{d,t} \Sigma_0^{-1})]}{nL} \tag{27}$$

$$\gamma_i = \frac{\mathrm{Tr}[B_{i,t}^{-1}(\Sigma_{d,t}^{(i)} + \mu_t^{(i)} (\mu_t^{(i)})^T)]}{L}, \quad \forall i \tag{28}$$

where $\mu_t^{(i)}$ represents the i-th block of μ^t, or the estimated drift value of sensor i, and $\Sigma_{d,t}^{(i)}$ represents the i-th principal diagonal block of $\Sigma_{d,t}$, or the covariance matrix of the drift of sensor i.

For $B_{i,t}$, Zhang and Rao recommended to assign the same value B_t for each block i to avoid overfitting [25]. The learning rule of B_t is

$$B = \frac{1}{n} \sum_{i=1}^{n} \frac{\Sigma_{d,t}^{(i)} + \mu_t^{(i)} (\mu_t^{(i)})^T}{\gamma_i}. \tag{29}$$

Algorithm 1. Drift Estimation Using T-SBL

```
Input  : $y$: sensor measurement, $\Phi$: observation matrix
Output: $\hat{d}$: estimated sensor drift
1  $\Phi_L \leftarrow \Phi \otimes I_L$
2  for $t \leftarrow L$ to $\infty$ step $L$ do
3      $z \leftarrow \Phi_L \cdot \mathrm{vec}(y_{t-L+1:t})$                // drift observation
4      $\gamma \leftarrow 1_{n\times 1}$, $\mu \leftarrow 0_{nL\times 1}$      // parameter initialization
5      $B \leftarrow I_L$, $\lambda \leftarrow 0.01$
6      $e \leftarrow \infty$                                     // T-SBL iteration
7      while $e < \epsilon$ do
8          $\mu_o \leftarrow \mu$
9          $\Sigma_0 \leftarrow \mathrm{diag}(\gamma) \otimes B$
10         $\mu \leftarrow \Sigma_0 \Phi_L^T (\lambda I + \Phi_L \Sigma_0 \Phi_L^T)^{-1} z$
11         $\Sigma \leftarrow \Sigma_0 - \Sigma_0 \Phi_L^T (\lambda I + \Phi_L \Sigma_0 \Phi_L^T)^{-1} \Phi_L \Sigma_0$
12         $B_s \leftarrow 0_{L\times L}$
13         for $i \leftarrow 1$ to $n$ do
14             $B_s \leftarrow B_s + (\Sigma^{(i)} + \mu^{(i)}(\mu^{(i)})^T)/\gamma_i$
15         $B \leftarrow B_s/n$
16         for $i \leftarrow 1$ to $n$ do
17             $\gamma_i \leftarrow \mathrm{Tr}[B^{-1}(\Sigma^{(i)} + \mu^{(i)}(\mu^{(i)})^T)]/L$
18         $\lambda \leftarrow \frac{1}{nL}\{\|z - \Phi_L \mu\|_2^2 + \lambda[pL - \mathrm{Tr}(\Sigma \Sigma_0^{-1})]\}$
19         $e \leftarrow \|\mu_o - \mu\|_\infty$
20     for $\tau \leftarrow 1$ to $L$ do
21         $\hat{d}_{t-L+\tau} \leftarrow \mu[(\tau-1)n+1 : \tau n]$
```

The T-SBL learning rules estimating sensor drift modeled by Eq. (16) are given by equations (18), (22), (23), (27), (28) and (29).

The drift estimation process is given in algorithm 1. The blocked observation matrix Φ_L is initialized first. On line 2, we let the time-variable t increase by step L. At each iteration, we use L samples of measurements to construct the observation vector on line 3, and then initialize the hyperparameters. The T-SBL algorithm runs through lines 7 to 19. The parameter μ and Σ are first estimated on lines 10, 11, and the hyperparameters B, γ and λ are estimated on lines 12-15, lines 16 to 17 and line 18, respectively, corresponding to Eq. (29), (28) and (27). T-SBL iteration stops when parameter μ converges. Finally, the MAP estimation of sensor drift $\hat{d}$ is given by parameter μ on lines 20 and 21.

V. Evaluation

In this section, we use both simulated and real-world datasets to evaluate the performance of the proposed algorithm. The performance is evaluated using two indexes: *recovery rate* and *mean square error (MSE)*.

For a sensor network with n sensors and T samples, let $D \in \mathbb{R}^{n\times T}$ denote the drift matrix, $D_{i\cdot}$ denote the i-th row or the drift of sensor i, and $\hat{D}$ denote the estimated sensor drift matrix. The MSE of drift estimation is defined as $\|\hat{D} - D\|_F^2/\|D\|_F^2$, where $\|\cdot\|_F$ is the Frobenius norm [37] of a matrix.

Before defining the term *recovery rate*, we first give the definition of a successful recovery. If m sensors are drifted, let $\mathbb{D}$ represent the set of the drifted sensors, and in the estimated sensor drift matrix $\hat{D}$, let $\hat{\mathbb{D}}$ denote the set of rows of $\hat{D}$ with m largest ℓ_2 norms. The drift recovery is successful only if $\mathbb{D} = \hat{\mathbb{D}}$. In the simulation, we independently enumerate many combinations of drifted sensors and run different drift estimation algorithms on each combination. Let $\mathbb{T}$ be the set of trials of one algorithm on different combinations, and $\mathbb{T}_s \overset{\text{def}}{=} \{\mathbb{T} | \mathbb{D} = \hat{\mathbb{D}}\}$ which is the successful recovery trials. The *recovery rate* is defined as $\|\mathbb{T}_s\|/\|\mathbb{T}\|$, i.e. the ratio of successful trials among all.

We also compare the proposed method with other drift calibration methods with some modifications, including:

- SVR-KF: a prediction-based drift calibration algorithm proposed in [21]
- SVR-KF-oracle: we assume that there exists an oracle which can directly detect drifted sensors, and use SVR-KF to calibrate them.
- SSP-KF-oracle: a modified version of SSP-KF proposed in [24], where the drift detection algorithm is replaced by an oracle.
- Lasso: we replace T-SBL in our proposed method with the widely used ℓ_1 solver Lasso [38].

The SVR-KF-oracle and SSP-KF-oracle are employed as best cases, since the uncertainty of drift sensor detection is eliminated by the oracle. The term *oracle* is an imaginary drift detector that can always correctly detect the drifted subset of sensors.

The experiments are implemented using Python and its numeric calculation libraries including SciPy [39] and NumPy [40]. The SVR and Lasso algorithms are provided by the scikit-learn library [41].

A. Datasets for Evaluation

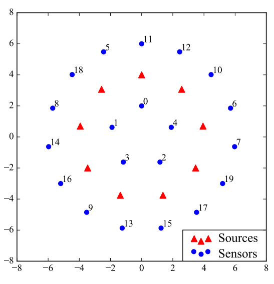
Although there are a number of open sensor datasets, they can only provide sensor measurements while the ground truth is unknown. We can not validate the correctness of calibration results without ground truth.

In our experiments, we first use simulated sensing fields with various deployments to evaluate the calibration algorithms under different conditions. We also deploy a simple sensor network with both cheap commodity-level and highly accurate commercial thermometers to obtain an almost drift-free dataset, and use it to validate calibration algorithms.

1) Simulated Sensing Field: To evaluate the performance of our algorithm under different circumstances, we simulate two different sensing fields, each of which has r signal sources and n sensors. One sensing field is named as *regular field*, where both signal sources and sensors are placed in a regular shape; the other is *random field*, where the sources and sensors are randomly placed.

We choose $r = 9$ and $n = 20$ in our experiment, and the sensing field is a circle with radius 6. The two sensing fields are shown in Fig. 1.

The source signals are generated using 8000 samples of lowpass-filtered ARMA processes, and the series are independent.


(a)

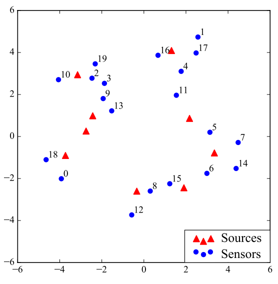

(b)

Fig. 1. Postions of signal sources and sensors in the two simulated sensing fields. (a) Regular Field. (b) Random Field.

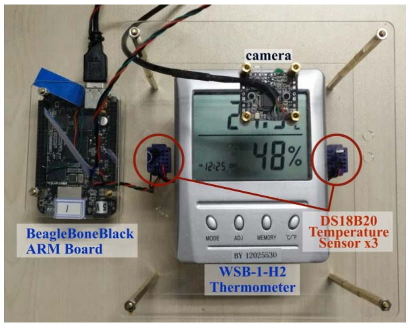


Fig. 2. Sensing unit: one WSB-1-H2 commercial thermometer, three DS18B20 temperature sensors (two on the right and one on the left of the thermometer), one BeagleBoneBlack ARM board and one camera.

The ground-truth measurement of each sensor is a linear combination of each source. For sensor i at time instant t, its ground-truth measurement is

$$x_{i,t} = \sum_{j=1}^{r} a_{i,j} s_{j,t}$$

where $s_{j,t}$ is the signal value of source j at time instant t, $a_{i,j}$ is the contribution of source j to sensor i, which is decided by the distance between the sensor and the source:

$$a_{i,j} = (\delta_{i,j} + 1)^{-1.5}$$

where $\delta_{i,j}$ is the distance between sensor i and source j. Hence, the signal subspace is totally determined by the geographic location of sensors and sources.

The simulated sensor measurements are shown in Fig. 6(a) and 6(b). From the figures, we can see that the ground-truth measurements of each sensor are correlated, but different from each other.

2) Real-World Dataset: We deploy six WSB-1-H2 commercial thermometers ($50 each) in our lab, and at the same location of each thermometer, three DS18B20 cheap temperature sensors ($0.5 each) are deployed. So in total, we deploy 24 temperature sensors in our lab. Each DS18B20 sensor measures the temperature every 30 seconds. We also use a camera to collect the reading of each thermometer every 5 minutes. The sensing unit is shown in Fig. 2.

Before deploying the sensing units, we first put all the thermometers in the same location, and the difference of the temperature readings is less than 0.1 °C, so the thermometers

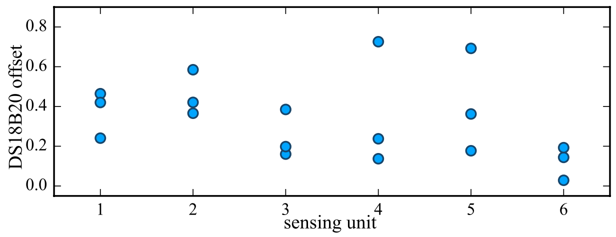


Fig. 3. Offsets of DS18B20 sensors to WSB-1-H2 thermometers in different sensing units. The offsets varies from 0.1 °C to 0.7 °C, so measurements from DS18B20 sensor without calibration is not accurate enough.

Algorithm 2. Virtual Sensor Data Generation

```
Input  : x: sensor measurements
Output: v: virtual sensor measurements
// iterate over locations, then sensors
for $i \leftarrow 0$ to 5 do
    for $j \leftarrow 1$ to 3 do
        $k \leftarrow (i + j)\mathrm{mod}6$          // choose a neighbour sensor
        $\boldsymbol{v}_{i,j} \leftarrow (\boldsymbol{x}_{k,j} + \boldsymbol{x}_{i,j})/2$
```

have a consensus on measurements. We use the sensor data collected in 15 days from Oct. 26 to Nov. 11 2015, and re-sample the data at 3-minute intervals, so 7200 samples of measurements are used in our experiment. To make use of the collected data, we need to calibrate the DS18B20 sensors with their corresponding thermometers. We take the first 400 samples of measurements, and for each sensing unit, we calculate the mean offset between DS18B20 measurements and that of their corresponding thermometer. The calculated offsets are scattered in Fig. 3, most of which is larger than 0.2 °C and some are as large as 0.7 °C.

After pre-calibration, the readings of the sensors and the thermometer from each sensing unit match well, as shown in Fig. 6(c), so the dataset can be considered drift-free.

Due to the limitation of this method to obtain drift-free data, although this dataset contains measurements of 24 sensors, they come from only 6 locations, and the data correlation between different locations might be limited. Hence, we use this dataset to further generate a virtual dataset containing data from thermometers and *virtual sensors*. For each DS18B20 sensor, we assume that a "virtual sensor" is located in the middle of the real sensor and one of its neighbours. The measurement of this virtual sensor is set to be the mean of its two corresponding sensors. The data of virtual sensors and thermometers construct the generated dataset. The detailed process is shown in algorithm 2.

Fig. 5 shows the locations of deployed sensors and thermometers, and the imaginary locations of virtual sensors. We select sensor group 1 and 2 and and plot their measurements in Fig. 6(c) in the real-world dataset, and Fig. 6(d) in the virtual dataset. As shown in the figures, the measurements in the real-world dataset are grouped in two lines. Therefore, in the real-world dataset, measurements from the thermometer and DS18B20 sensors matches well, and the dataset can be considered to be drift-free. In the virtual dataset,

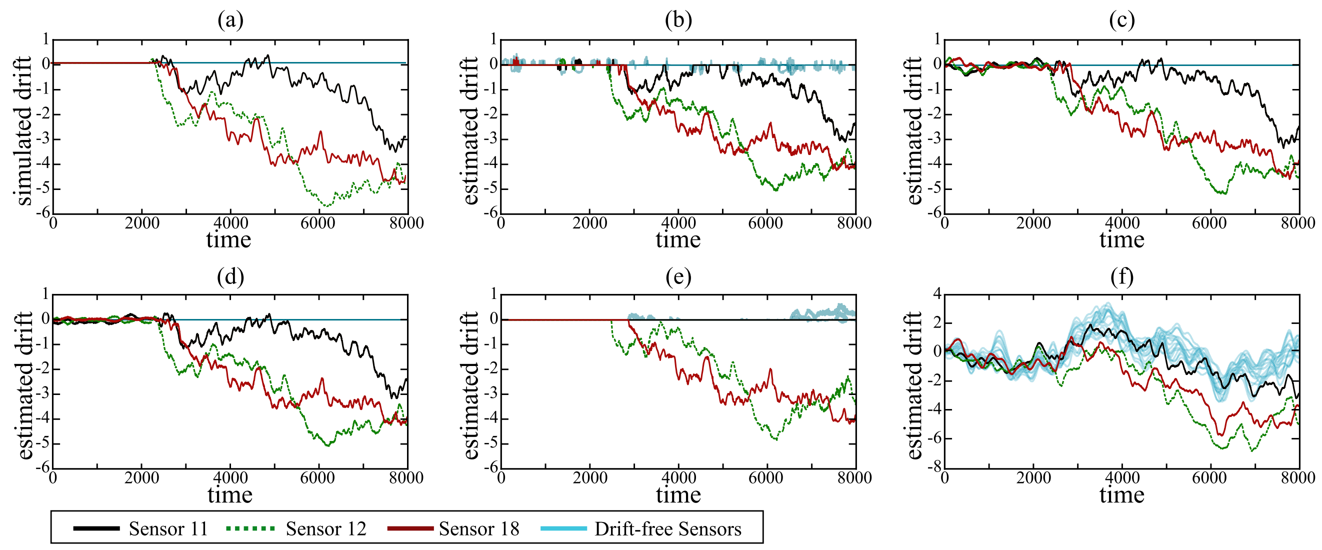


Fig. 4. Demonstration of calibration results from different algorithms on one case of simulations in regular field where sensors 11, 12 and 18 are drifted: (a) shows the simulated sensor drift; (b) T-SBL solution is very near to the ground-truth; (c) and (d) SVR-KF-oracle and SSP-KF-oracle have precise value estimation; (e) Lasso failed to recover all the drifts; and (f) the original SVR-KF obtained accumulative prediction error and failed to estimate drift.

measurements from virtual sensors are different but correlated with each other.

B. Results in Simulated Field

For simulated dataset, we randomly pick m sensors as drifted sensors, and generate a random walk process as emulated drift starting at time instance 2000, so the first 2000 samples of measurements are drift-free, and are used in the learning phase to estimate the observation matrix $\mathbf{\Phi}$ and train the SVR model used by SVR-KF method. The random walk process is generated by:

$$d_{i,1..2000} = 0$$
$$d_{i,t} = d_{i,t-1} + \delta_{i,t} \quad \delta_{i,t} \sim \mathcal{N}(0, \sigma^2) \tag{30}$$

where the variance of drift increment σ^2 is set to 0.03.

In the learning phase, the PCA threshold is set to 0.99. In both random and regular field dataset, the first 7 singular values have accumulated 99% of data energy, so we choose the corresponding 7 vectors as the estimation of basis vectors of the signal subspace, and the rest 13 vectors compose the observation matrix $\mathbf{\Phi}$.

Let the number of drifted sensors m vary from 1 to 8. At each drift number m, there exist $\binom{20}{m}$ combinations of different drifted sensors. However, this number is too large for us to enumerate and evaluate the calibration algorithms, so we randomly generate $20 + 10 \times (m - 1)$ different combinations of drifted sensors with independent drift. For each trial, we use different methods to estimate the sensor drift and compare their performance. According to the experiments in [25], the MMV parameter L of T-SBL is set to 5, since smaller L cannot make full use of temporal correlation. The iteration stop condition value ϵ is 0.01, and the ℓ_1-penalty coefficient of Lasso is set to 0.01.

Fig. 4 shows the calibration result of different algorithms in one trial with 3 drifted sensors. Fig. 4(a) is the ground-truth of sensor drift, and Fig. 4(b) is drift estimation of the proposed T-SBL based method. Fig. 4(c) and 4(d) are estimated using SVR-KF-oracle and SSP-KF-oracle respectively where the indexes of drifted sensors are known. Results of Lasso and SVR-KF are shown in Fig. 4(e) and Fig. 4(f). respectively.

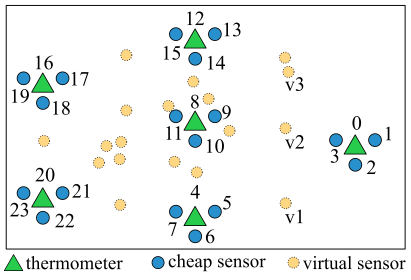


Fig. 5. Approximate locations of sensor and thermometers of the real-world dataset, the deployed sensors are grouped by 4 in 6 locations, the virtual sensors are in different locations.

The result demonstrates that 1) if the drifted sensors are known, both SVR-KF and SSP-KF can precisely estimate the sensor drift; 2) the result of the proposed T-SBL based method is also accurate but a little noisy; 3) Lasso fails to estimate the sensor drift 4) SVR-KF does not work out a stable solution since the prediction error accumulates and interferes each other.

The performance of different methods under different numbers of drifted sensors in random and regular field are shown in Fig. 7, where the blue line is obtained in the regular field and the red dashed line in the random field. In Fig. 7(a) we compare the recovery rate of T-SBL and Lasso. Fig. 7(b) shows the average MSE of each trial of T-SBL, Lasso, SVR-KF-oracle and SSP-KF-oracle. In the MSE comparison, only successful recoveries are taking into account.

This experiment shows that 1) with the increasing number of drifted sensors, the calibration MSE slightly increases while the recovery rate decrease; 2) the recovery rate of the T-SBL

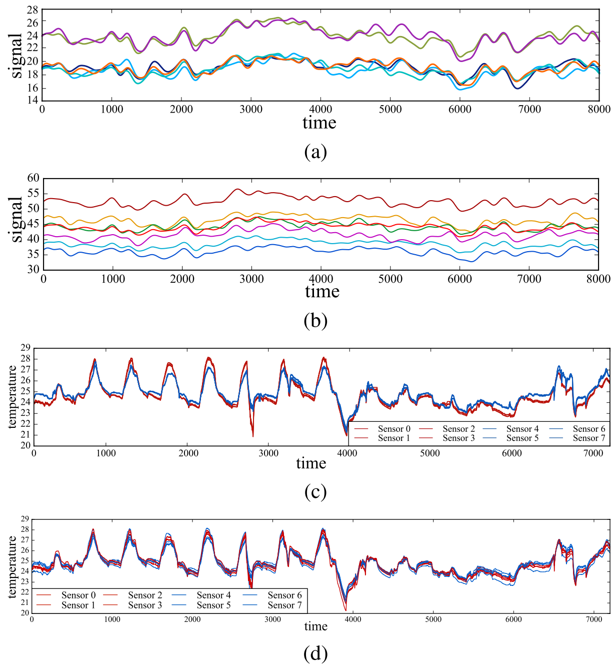


Fig. 6. Samples of dataset for evaluation: readings form each sensor are different but correlated. (a) Ground-truth measurement in regular field. (b) Ground-truth measurement in random field. (c) Measurements from real deployed temperature sensors: measurements are grouped into two lines. (d) Virtual measurements generate from real deployed temperature sensors: measurements from different sensors are correlated but different.

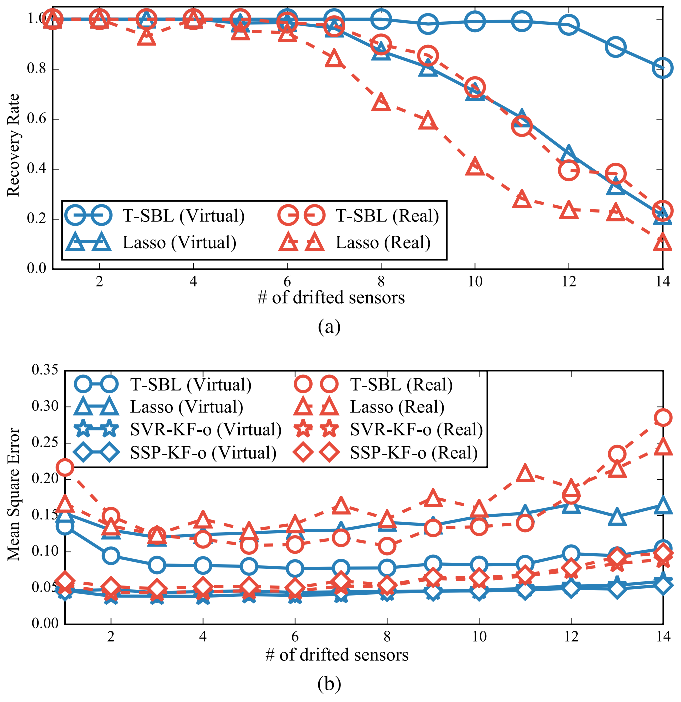


Fig. 8. Recovery rate and MSE of different methods in real-world and virtual dataset when the number of drifted sensors varies from 1 to 14. Failed recoveries are eliminated in MSE calculation. (a) Recovery rate of T-SBL and Lasso. (b) Mean square error of different methods.

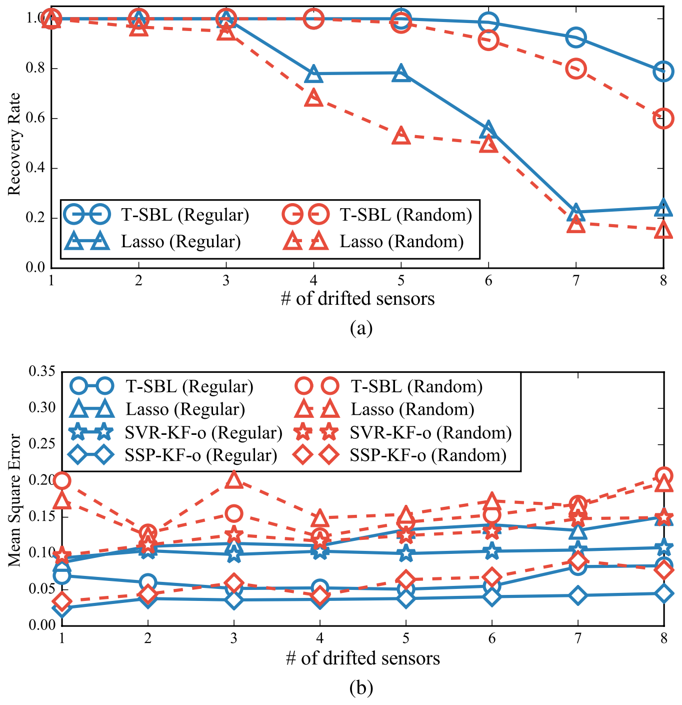


Fig. 7. Recovery rate and MSE of different methods in simulated random and regular field when the number of drifted sensors varies from 1 to 8. Failed recoveries are eliminated in MSE calculation. (a) Recovery rate of T-SBL and Lasso. (b) Mean square error of different methods.

calibrated result is significantly higher than the Lasso result in both the regular and random field; 3) the recovery rates of both the Lasso and T-SBL in the regular field is higher than that in the random field; 4) among all the calibration methods, SSP-KF-oracle has the lowest MSE, and Lasso has the highest MSE.

The experiments in the simulated dataset demonstrate that the proposed T-SBL based calibration method can successfully recover sensor drift in most cases where the number of drifted sensors is not too large. In the simulated regular field, when 40% sensors are drifted, T-SBL has the recovery rate at about 80%, while Lasso can only recovery about 60%. The original SVR-KF fails to accurately estimate sensor drift due to the prediction error accumulation.

C. Results in Real-World Dataset

Now we evaluate the performance of different algorithms in the real-world dataset and the generated virtual dataset. We also pick the first 2000 samples of measurements as the drift learning set to estimate observation matrix $\mathbf{\Phi}$ and train the SVR model. In the real-world dataset, the largest 5 singular values accumulated 99% of data energy, and in the virtual dataset, 99% data energy is accumulated by the largest 4 singular values. Here we can see that the virtual dataset has more correlation.

We set the number of drifted sensors m to vary from 1 to 14, and generate $24 + 10 \times (m - 1)$ different combinations of drifted sensors, then use different methods to calibrate sensor drift for each trial. For the T-SBL algorithm, we set the MMV parameter L to 5, and the iteration stop condition value ϵ is set to 0.01. For Lasso, the ℓ_1-penalty coefficient is set to 0.01. The recovery rate and MSE is shown are Fig. 8.

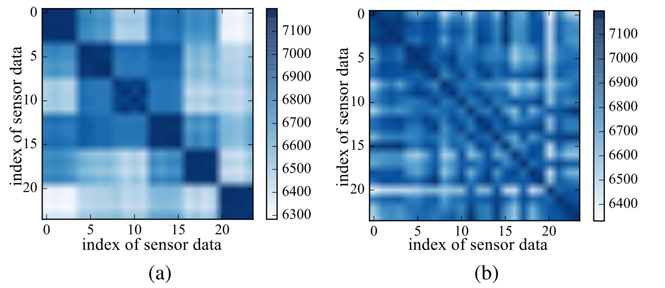


Fig. 9. Covariance matrix of the real-world dataset and the generated virtual dataset, the real-world dataset has a blocked covarience matrix. (a) Real-world dataset. (b) Virtual dataset.

Fig. 8(a) also shows that T-SBL has a higher recovery rate than Lasso. In the virtual dataset, the recovery rate is significantly higher than that in the real-world dataset. The reason for this is that although the real-world dataset is measured by 24 sensors, they are deployed as 6 groups and each group has 4 sensors deployed in the very same location, so the inter-location correlation is is not high enough. However, in the virtual dataset, the virtual sensors are distributed in different locations and the measurements are generated using different real sensors, and this generation method ensures high data correlation.

We plot the covariance matrix of the real-world and virtual dataset in Fig. 9. It is obvious that the covariance matrix of the real-world dataset has a blocked structure, where inter-block correlation is significantly lower than intra-block correlation. The size of each block is 4, as we deploy 4 sensors in each location. As a comparison, the virtual dataset has a more homogeneous covariance matrix, and the average correlation between sensors is higher than that of the real-world dataset.

The MSE of different calibration results are shown in Fig. 8(b). From the figure, we can see that SVR-KF-oracle and SSP-KF-oracle both have the lowest MSE. The MSE of T-SBL is as high as that of Lasso in the real-world dataset. While in virtual dataset, T-SBL has much lower MSE value.

To summarize, we find that the deployed real-world sensor network does not provide enough redundancy for calibration. Using the proposed T-SBL method, when the number of drifted sensors is larger than 8 (of 24), the recovery rate is below 80%, and the average MSE is 3 times of the best cases. The generated virtual dataset has a higher correlation among sensors. The proposed calibration method has a higher recovery ability in this dataset. When 14 of 24 sensors are drifted, the recovery rate is still higher than 80%, and the MSE is about 2 times as large as the best cases.

VI. Discussion and Future Work

The experiment results have shown that the proposed T-SBL based calibration method has the ability to estimate sensor drift when multiple sensors are drifted in a sensor network. While some interesting phenomenons are found in the experiments, and some issues also exist. Next we discuss some of them in this section.

A. Sensing Field Matters

As shown in our experiments, in the simulated dataset, the recovery rate of our method in regular field is slightly higher than that in the random field, and the MSEs of all methods in the random field are higher than that in the regular field. In the real-world dataset, compared with the virtual dataset, all methods results in lower recovery rate and higher MSE.

In our model, the sensing field decides the observation matrix $\mathbf{\Phi}$, and in compressed sensing, the recovery probability is decided by the *restricted isometry principle* (RIP) [42] property of the observation matrix. In prediction-based methods, the correlation of sensory data also has a impact on the accuracy of predicted values. This means that the recovery ability of the proposed calibration method is decided by the sensing field.

Although we demonstrate that some cases of sensing fields have higher recovery ability, so far we can not find a quantitative model to describe the recoverability of a sensing field.

Obviously, the more the sensory data are correlated, the more possible sensor drift can be calibrated. Our real-world experiment shows that deploying a group of sensors in one location is not as redundant as spreading the sensors to the monitoring field, since the former deployment can only utilize the intra-group correlation while inter-group correlation might not be enough to be utilized in calibration.

On this issue, a future direction is to design a methodology of sensor deployment planning, including sensor number, placement and sample frequency, to seek for optimal calibration ability under the constraint of cost and physical limitations.

B. Detection and Calibration

Our experiment results show that if the drifted sensors are known, both SSP-KF and SVR-KF can obtain very accurate drift estimation; while without drift detection, the original SVR-KF produced erroneous estimation. This indicates that the uncertainty of drift sensor detection has more impact on drift value estimation.

Both T-SBL and Lasso have the ability to detect drifted sensors, but they cannot minimize MSE of estimation, since MSE is not the only optimization target of compressed sensing algorithms. One possible solution is to use compressed sensing to identify drifted sensors, and then use SSP-KF or SVR-KF to estimate the drift value. An existing work is the CSKF proposed in [43], but the problem of this method is that the detection error can cause the accumulative estimation error.

In our previous work [24], drift detection and identification is achieved using a naïve threshold-based method and brute force search. This is applicable when only one sensor has unknown drift.

Another issue of drift detection in the proposed algorithm is that drift can only be detected if its value is large enough because of the existence of observation noise. Although T-SBL outperforms other CS methods on noise robustness, the recovery rate significantly decreases at low SNR (drift-noise-ratio in our case) [25]. Ignoring this issue is acceptable, since sensor drift as small as noise should be tolerable in many applications.

One future direction is to separate the detection and calibrate process. Therefore, more information such as temporal correlation of sensory data, statistical features, or some other prior knowledge can be employed to achieve higher detection rate. Small drift is also possible to be detected because of the difference of statistical features of drift and noise. Once drifted sensors can be detected and identified, it is much easier to obtain a precise estimation of drift value.

C. Time-Variant Signal Subspace

One fundamental assumption of this work is that the signal subspace does not change over time. Although we have shown in the real-world experiment that this assumption is acceptable in some applications, in practice, the signal subspace is decided by both sensor locations and the field of interest, which may change slowly overtime. In this case, the accumulated change can influence the performance of calibration algorithm.

In order to satisfy more applications, future work on blind calibration with time-variant signal subspace should been done.

D. Engineering Guidelines

From engineering perspective, it is not necessary to seek the ultimate calibratability of WSNs. For planning a WSN project, more redundant sensors make sensory data more drift-tolerant. If the sensors have to be sparsely deployed, or data-correlation among sensors cannot be guaranteed, putting multiple sensors on a single node can be a possible solution.

According to our experiments, the recovery rate depends on multiple factors including physical environment, sensor placement, sensor density, etc. Since any calibration algorithm has its limitations, and the more the number of drifted sensors, the lower probability sensors can be calibrated, it is unwise to rely on calibration results when many sensors are drifted. For long-term monitoring applications, calibration algorithms can delay the process of unrecoverable drift, and the maintainers should get ready for field calibrating, or replacing bad sensors when multiple sensor starts drifting, so that continuous and accurate monitoring data can be ensured.

VII. Conclusion

The need of blind calibration of sensor networks has become urgent as large-scale and long-term sensor networks are deployed in various applications. Sensor drift is a fundamental problem of sensor networks which influences the trustworthiness of sensory data. In this paper, we proposed a blind calibration framework based subspace projection and sparse recovery (SPSR), and applied temporally-correlated sparse Bayesian learning to estimate the sensor drift from under-sampled drift observations. The proposed method needs neither the existence of a prior model of the measurand data, nor the dense deployment of the sensors.

By modeling the ground-truth of sensory data as a signal subspace to utilize the spatial correlation of the sensors, a low-rank observation of the sensor drift can be obtained, and using the T-SBL recovery algorithm which utilizes the temporal correlation and common-sparsity of the sensor drift in multiple time instants, the sensor drift can be estimated.

Experiments show that the proposed method can accurately estimate multiple sensors' drift, while different sensing fields has different recovery ability. One important future work is to investigate the methodology of sensor network planning for optimal recoverability.

References

[1] P. Bellavista, G. Cardone, A. Corradi, and L. Foschini, "Convergence of MANET and WSN in IoT urban scenarios," *IEEE Sensors J.*, vol. 13, no. 10, pp. 3558–3567, Oct. 2013.

[2] M. Franceschinis, M. A. Spirito, R. Tomasi, G. Ossini, and M. Pidalà, "Using WSN technology for industrial monitoring: A real case," in *Proc. 2nd Int. Conf. Sensor Technol. Appl. (SENSORCOMM)*, Aug. 2008, pp. 282–287.

[3] H. Ghayvat, J. Liu, S. C. Mukhopadhyay, and X. Gui, "Wellness sensor networks: A proposal and implementation for smart home for assisted living," *IEEE Sensors J.*, vol. 15, no. 12, pp. 7341–7348, Dec. 2015.

[4] N. Kushalnagar, G. Montenegro, and C. P. Schumacher, "IPv6 over low-power wireless personal area networks (6LoWPANs): Overview, assumptions, problem statement, and goals," IETF Request for Comments (RFC) 4919, Aug. 2007. [Online]. Available: https://tools.ietf.org/html/rfc4919

[5] C. Bormann, A. P. Castellani, and Z. Shelby, "CoAP: An application protocol for billions of tiny internet nodes," *IEEE Internet Comput.*, vol. 16, no. 2, pp. 62–67, Mar./Apr. 2012.

[6] P. Levis *et al.*, "Tinyos: An operating system for sensor networks," in *Ambient Intelligence*. Heidelberg, Germany: Springer, 2005, pp. 115–148.

[7] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki—A lightweight and flexible operating system for tiny networked sensors," in *Proc. 29th Annu. IEEE Int. Conf. Local Comput. Netw.*, Nov. 2004, pp. 455–462.

[8] W. Liu *et al.*, "Design and implementation of a hybrid sensor network for Milu Deer monitoring," in *Proc. 14th Int. Conf. Adv. Commun. Technol. (ICACT)*, Feb. 2012, pp. 52–56.

[9] X. Fei *et al.*, "A reliable transfer protocol for multi-parameter data collecting in wireless sensor networks," in *Proc. 15th Int. Conf. Adv. Commun. Technol. (ICACT)*, Jan. 2013, pp. 569–573.

[10] W. Liu *et al.*, "Application specific sensor node architecture optimization—Experiences from field deployments," in *Proc. IEEE 17th Asia South Pacific Design Autom. Conf.*, Jan./Feb. 2012, pp. 389–394.

[11] K. Ni *et al.*, "Sensor network data fault types," *ACM Trans. Sensor Netw.*, vol. 5, no. 3, May 2009, Art. no. 25.

[12] R. Tan, G. Xing, Z. Yuan, X. Liu, and J. Yao, "System-level calibration for data fusion in wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 9, no. 3, May 2013, Art. no. 28.

[13] C. Xiang, P. Yang, C. Tian, H. Cai, and Y. Liu, "Calibrate without calibrating: An iterative approach in participatory sensing network," *IEEE Trans. Parallel Distrib. Syst.*, vol. 26, no. 2, pp. 351–361, Feb. 2015.

[14] V. Bychkovskiy, S. Megerian, D. Estrin, and M. Potkonjak, "A collaborative approach to in-place sensor calibration," in *Proc. 2nd Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, 2003, pp. 301–316.

[15] B.-T. Lee, S.-C. Son, and K. Kang, "A blind calibration scheme exploiting mutual calibration relationships for a dense mobile sensor network," *IEEE Sensors J.*, vol. 14, no. 5, pp. 1518–1526, May 2014.

[16] M. Takruri, K. Aboura, and S. Challa, "Distributed recursive algorithm for auto calibration in drift aware wireless sensor networks," in *Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering*, K. Elleithy, Ed. The Netherlands: Springer, 2008, pp. 21–25, doi: 10.1007/978-1-4020-8735-6_5.

[17] Y. Xiang *et al.*, "Collaborative calibration and sensor placement for mobile sensor networks," in *Proc. 11th Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, Apr. 2012, pp. 73–83.

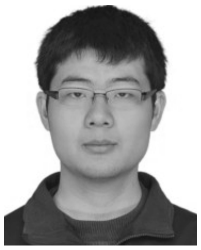
[18] O. Saukh, D. Hasenfratz, and L. Thiele, "Reducing multi-hop calibration errors in large-scale mobile sensor networks," in *Proc. 14th Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, 2015, pp. 274–285.

[19] L. Balzano and R. Nowak, "Blind calibration of sensor networks," in *Proc. 6th Int. Symp. Inf. Process. Sensor Netw.*, Apr. 2007, pp. 79–88.

[20] M. Takruri and S. Challa, "Drift aware wireless sensor networks," in *Proc. 10th Int. Conf. Inf. Fusion*, Jul. 2007, pp. 1–7.

[21] M. Takruri, S. Rajasegarar, S. Challa, C. Leckie, and M. Palaniswami, "Online drift correction in wireless sensor networks using spatio-temporal modeling," in *Proc. 11th Int. Conf. Inf. Fusion*, Jun./Jul. 2008, pp. 1–8.

[22] D. Kumar, S. Rajasegarar, and M. Palaniswami, “Automatic sensor drift detection and correction using spatial Kriging and Kalman filtering,” in *Proc. DCOSS*, May 2013, pp. 183–190.
[23] D. Kumar, S. Rajasegarar, and M. Palaniswami, “Geospatial estimation-based auto drift correction in wireless sensor networks,” *ACM Trans. Sensor Netw.*, vol. 11, no. 3, 2015, Art. no. 50.
[24] Y. Wang, A. Yang, Z. Li, P. Wang, and H. Yang, “Blind drift calibration of sensor networks using signal space projection and Kalman filter,” in *Proc. 10th ISSNIP*, Apr. 2015, pp. 1–6.
[25] Z. Zhang and B. D. Rao, “Sparse signal recovery with temporally correlated source vectors using sparse Bayesian learning,” *IEEE J. Sel. Topics Signal Process.*, vol. 5, no. 5, pp. 912–926, Sep. 2011.
[26] S. Ó. Buadhacháin and G. Provan, “A model-based control method for decentralized calibration of wireless sensor networks,” in *Proc. Amer. Control Conf. (ACC)*, Jun. 2013, pp. 6571–6576.
[27] J. Feng, S. Megerian, and M. Potkonjak, “Model-based calibration for sensor networks,” in *Proc. IEEE Sensors*, vol. 2, Oct. 2003, pp. 737–742.
[28] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[29] E. J. Candès, J. Romberg, and T. Tao, “Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information,” *IEEE Trans. Inf. Theory*, vol. 52, no. 2, pp. 489–509, Feb. 2006.
[30] A. Cichocki and S.-I. Amari, *Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications*. New York, NY, USA: Wiley, 2002.
[31] B. Eisenberg and R. Sullivan, “Why is the sum of independent normal random variables normal?” *Math. Mag.*, vol. 81, no. 5, pp. 362–366, 2008.
[32] S. F. Cotter, B. D. Rao, K. Engan, and K. Kreutz-Delgado, “Sparse solutions to linear inverse problems with multiple measurement vectors,” *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2477–2488, Jul. 2005.
[33] Y. C. Eldar and M. Mishali, “Robust recovery of signals from a structured union of subspaces,” *IEEE Trans. Inf. Theory*, vol. 55, no. 11, pp. 5302–5316, Nov. 2009.
[34] M. E. Tipping, “Sparse Bayesian learning and the relevance vector machine,” *J. Mach. Learn. Res.*, vol. 1, pp. 211–244, Jun. 2001.
[35] B. D. Rao, Z. Zhang, and Y. Jin, “Sparse signal recovery in the presence of intra-vector and inter-vector correlation,” in *Proc. IEEE Int. Conf. Signal Process. Commun. (SPCOM)*, Jul. 2012, pp. 1–5.
[36] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum likelihood from incomplete data via the EM algorithm,” *J. Roy. Statist. Soc. Ser. B (Methodol.)*, vol. 39, no. 1, pp. 1–38, 1977.
[37] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 1996.
[38] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *J. Roy. Statist. Soc. Ser. B (Methodol.)*, vol. 58, no. 1, pp. 267–288, 1996.
[39] Scipy Developers. *Scipy*, accessed on Jun. 25, 2016. [Online]. Available: http://www.scipy.org/
[40] Numpy Developers. *Numpy*, accessed on Jun. 25, 2016. [Online]. Available: http://www.numpy.org/
[41] F. Pedregosa *et al.*, “Scikit-learn: Machine learning in Python,” *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Oct. 2011.
[42] E. J. Candès and M. B. Wakin, “An introduction to compressive sampling,” *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 21–30, Mar. 2008.
[43] N. Vaswani, “Kalman filtered compressed sensing,” in *Proc. 15th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2008, pp. 893–896.



Yuzhi Wang (S’14) received the B.S. degree from the School of Telecommunication Engineering, Xidian University, Xi’an, China, in 2012. He is currently pursuing the Ph.D. degree with the Department of Electronic Engineering, Tsinghua University, Beijing, China, under the supervision of Prof. H. Yang. His research interests include wireless sensor networks, Internet of Things, network security, and machine learning.

Anqi Yang received the B.S. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2014, where he is currently pursuing the Ph.D. degree under the supervision of Prof. H. Yang. His current research interests include wireless sensor networks, signal processing, compressed sensing, and deep neural networks.

Zhan Li received the B.S. degree from the School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, China, in 2013, and the M.S. degree from the Department of Electronic Engineering, Tsinghua University, Beijing, in 2016. His research interests include wireless sensor networks and machine learning.

Xiaoming Chen (S’12–M’15) received the B.S. and Ph.D. degrees from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2009 and 2014, respectively. Since 2014, he has been a Post-Doctoral Researcher in electrical and computer engineering with Carnegie Mellon University, Pittsburgh, PA, USA. His current research interests include CAD algorithms for reliable and trusted circuit design and Internet of Things. He received the 2015 EDAA Outstanding Dissertation Award. He also received three best paper nominations at ISLPED’09, ASP-DAC’12, and ASP-DAC’14.

Pengjun Wang received the B.S. and Ph.D. degrees from the Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2006 and 2011, respectively. He is currently an Assistant Research Scientist with the Department of Electronic Engineering with Tsinghua University. Since 2014, he has also been the CTO of Smartbow Tech., Inc. His recent research mainly focuses on wireless sensor networks and structural health monitoring.

Huazhong Yang (M’97–SM’00) received the B.S. degree in microelectronics and the M.S. and Ph.D. degrees in circuits and systems from Tsinghua University, Beijing, China, in 1989, 1993, and 1998, respectively. Since 1993, he has been with the Department of Electronic Engineering, Tsinghua University, where he has been a Full Professor since 1998, and a specially appointed Professor of the Cheung Kong Scholars Program since 2011. His current research interests include wireless sensor networks, structural health monitoring, brain-inspired computing, nonvolatile processors, and energy-harvesting circuits. He has authored or co-authored over 300 technical papers and 70 granted patents.