Title	Distributed Clock Parameters Tracking in Wireless Sensor Network
Author(s)	Luo, B; Wu, YC
Citation	IEEE Transactions on Wireless Communications, 2013, v. 12 n. 12, p. 6464-6475
Issued Date	2013
URL	http://hdl.handle.net/10722/199093
Rights	IEEE Transactions on Wireless Communications. Copyright © IEEE.

Distributed Clock Parameters Tracking in Wireless Sensor Network

Bin Luo and Yik Chung Wu

***Abstract*—Clock parameters (skew and offset) in sensor network are inherently time-varying due to imperfect oscillator circuits. This paper develops a distributed Kalman filter for clock parameters tracking. The proposed algorithm only requires each node to exchange limited information with its direct neighbors, thus is energy efficient, scalable with network size, and is robust to changes in network connectivity. A low-complexity distributed algorithm based on Coordinate-Descent with Bootstrap (CD-BS) is also proposed to provide rapid initialization to the tracking algorithm. Simulation results show that the performance of the proposed distributed tracking algorithm maintains long-term clock parameters accuracy close to the Bayesian Cramer-Rao Lower Bound.**

***Index Terms*—Distributed clock synchronization, wireless sensor networks (WSNs), distributed Kalman filter, Bayesian Cramer-Rao lower bound.**

I. Introduction

WIRELESS Sensor Networks (WSNs) typically consist of inexpensive, small-sized, power-limited terminals (known as senor nodes) capable of onboard sensing, computing and communications. WSNs are used to monitor data that would be difficult or inconvenient to monitor using wired equipment. These applications include monitoring habitat environments, controlling industrial machines and home appliances, object tracking and event detection, etc. [1], [2]. Most of these applications require collaborative execution of a distributed task amongst a set of synchronized sensor nodes. Moreover, data fusion, power management, transmission scheduling, localization and tracking protocols demand all the nodes running on a common time frame. However, each sensor in a WSN has its own clock. Different clocks will drift from each other over time owning to imperfection in oscillator circuits. This necessitates synchronization algorithms that achieve and maintain *global clock synchronization*.

Over the last decade, a wide variety of clock synchronization protocols have been proposed. Existing synchronization protocols can be divided into two categories depending on how synchronization is executed: pairwise-based and fully distributed. In the pairwise-based protocols, clock synchronization is achieved by building a hierarchical network structure (spanning tree or cluster) and performing pairwise synchronization between adjacent levels or clusters. Two of the most representative protocols in this category are Time synchronization Protocol for Sensor Network (TPSN) [3], and Reference Broadcast Synchronization (RBS) [4]. Some other similar algorithms include Flooding Time Synchronization Protocol (FTSP) [5], Lightweight Tree-based Synchronization (LTS) [6], Tiny-sync [7], Pairwise Broadcast Synchronization (PBS) [8], Delay Measurement Time Synchronization [9], and Hierarchy Referencing Time Synchronization (HRTS) [10]. The disadvantages of this kind of approach are that it requires large overhead to maintain the hierarchical structure and rapid accumulation of synchronization error as distance from reference node increases.

On the other hand, for fully distributed synchronization algorithms, there is no special network structure. All sensors only have to communicate with their neighboring nodes, thus these protocols are robust to dynamic networks and are scalable with network size. This kind of algorithms can be further divided into two subclasses: pulse-coupled based and packet-coupled based. For the former, sensors are synchronized using physical layer pulses [14]–[16]. Despite the easy implementation and elegant theoretical support, pulse-coupled synchronization only provides a unified ticking rhythm but not precise clock reading. On the other hand, for the latter, timing messages between any two nodes are exchanged in the form of data package. Examples in this class include the average consensus principle based clock synchronization [11]–[13], and belief propagation based methods [23], [24]. Unfortunately, in consensus based methods, message delays are not considered, which causes large mean-square-error in converged clocks, while for existing belief propagation based methods, only clock offset is considered, resulting in the need of frequent re-synchronization.

Even after global synchronization in sensor network, individual clock would drift away from each other, and eventually call for re-synchronization. It is obvious that we can re-perform the distributed synchronization algorithms mentioned above. However, due to the slow-varying nature of clock parameters, the previously estimated clock parameters are useful in predicting the clock parameters in re-synchronization. Therefore, instead of discarding the previous estimated clock parameters, clock parameter tracking received some attentions recently. By assuming clock skew and clock offset can be directly observed subjected to noise, clock skews are tracked by Kalman filter in [21], while both clock skew and offset are tracked in [22]. Recently, graphical models are used in [25], [26] to derive a message-passing method for the clock offsets tracking in the presence of exponential family distributed random delays. However, all the above tracking algorithms were derived for synchronizing a pair of nodes only.

Manuscript received May 6, 2013; revised August 14, 2013; accepted October 8, 2013. The associate editor coordinating the review of this paper and approving it for publication was L. Lai.

This work was supported in part by the HKU seed Funding Programme, Project No. 201210159035.

The authors are with the Department of Electrical and Electronic Engineering, the University of Hong Kong, Hong Kong (e-mail: {luobin, ycwu}@eee.hku.hk).

Digital Object Identifier 10.1109/TWC.2013.103013.130811

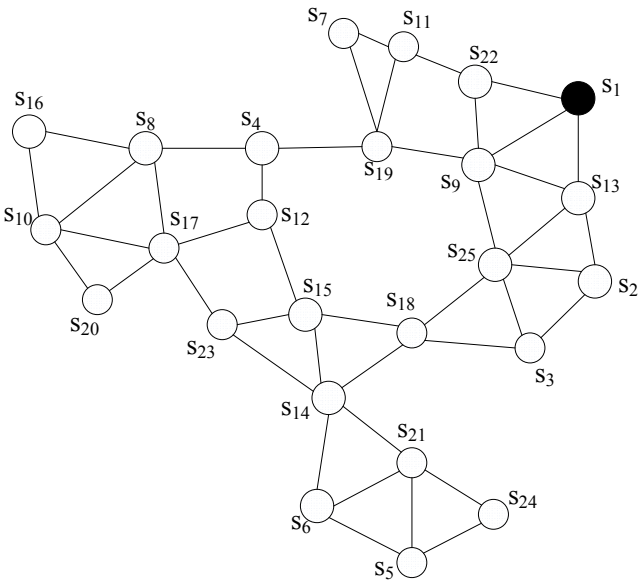


Fig. 1: Topology of a network with 25 sensors $\{S_i\}_{i=1}^{25}$.

In this paper, with clock parameters evolution equations developed based on the oscillator phase noise model, a fully-distributed clock offset and skew tracking algorithm using Kalman filtering is proposed. The distributed Kalman filter can achieve global synchronization in a distributed way, and the accuracy of clock parameters can be maintained close to Bayesian Cramer-Rao Lower Bound (Bayesian CRLB). It also performs well even when there is node failure, packet loss or new node joining in.

The rest of the paper is organized as follows. In Section II, the state-space model for the synchronization problem is developed. In Section III, distributed clock parameters tracking algorithm based on Kalman filter is proposed with a low-complexity initialization presented in Section IV. The Bayesian CRLB for global clock parameters tracking is derived in Section V. Simulation results are presented in Section VI. Finally, conclusions are drawn in Section VII.

Notation: The operator $\mathrm{Tr}\{\mathbf{A}\}$ takes the trace of matrix $\mathbf{A}$ and the operator $\mathrm{vec}(\mathbf{A})$ represents the vectorization of matrix $\mathbf{A}$. Superscript $(\cdot)^T$ denotes the transpose operator and $\mathbf{I}_N$ indicates an $N \times N$ identity matrix. Notation $\mathrm{E}\{\cdot\}$ takes the expectation. $\mathcal{N}_i$ denotes the set of neighbors of node S_i, with $\mathcal{N}_i(j)$ indicates the jth element in set $\mathcal{N}_i$. $\mathcal{N}_{[i]}$ describes the subsystem formed by the node S_i and the nodes in its neighbor set. Finally, $\otimes$ stands for the Kronecker product.

II. System Model

Consider a network with N sensor nodes $\{S_1, S_2, \cdots, S_N\}$. These sensors are randomly distributed in the field and can be self-organized into a network by establishing connections between neighbor nodes lying within each other's communication range. An example of 25 sensor nodes is shown in Figure 1, where each edge represents the ability to transmit and receive packets between the pair of nodes. Each sensor S_i has an analog clock characterized by an oscillator [16]:

$$\rho_i(t) = \cos \Phi_i(t), \tag{1}$$

where $\Phi_i(t)$ is the instantaneous phase, which evolves as:

$$\Phi_i(t) = 2\pi(f_0 + \Delta f_i)t + \Phi_i(0) + \varsigma_i(t), \tag{2}$$

where f_0 is the center frequency; Δf_i is the frequency offset that depends on hardware imperfections; $\Phi_i(0)$ is the initial phase; $\varsigma_i(t) = 2\pi f_0 \sqrt{p_i} B(t)$ is a random process modelling phase noise, with $B(t)$ represents the standard *Wiener* process [30], and p_i is a parameter describing degree of phase noise. In particular, p_i can be computed based on phase noise level $\mathcal{L}(f) = 10\log_{10}\left(p_i f_0^2 / f^2\right)$ at certain frequency offset f with respect to the oscillator center frequency f_0, which is available in datasheet. On the other hand, p_i can also be computed based on the RMS period jitter or RMS phase jitter. Details have been given in Appendix A. From (2), the clock reading evolves as:

$$\begin{aligned} c_i(t) = \frac{\Phi_i(t)}{2\pi f_0} &= \frac{f_0 + \Delta f}{f_0} t + \frac{\Phi_i(0)}{2\pi f_0} + \frac{\varsigma_i(t)}{2\pi f_0} \\ &\triangleq \xi_i t + \theta_i^0 + \sqrt{p_i} B(t) \end{aligned} \tag{3}$$

where ξ_i is the normalized frequency, and θ_i^0 represents the initial clock offset of node S_i.

The above clock reading model can also be expressed in terms of a time-varying skew and initial clock offset as:

$$c_i(t) = \int_0^t \beta_i(\tau) d\tau + \theta_i^0. \tag{4}$$

Comparing (3) and (4), the time-varying clock skew and phase noise are related by: $\int_0^t \beta_i(\tau) d\tau = \xi_i t + \sqrt{p_i} B(t)$, and then differentiating both sides with respect to t, we can obtain

$$\beta_i(t) = \xi_i + \sqrt{p_i} B^{'}(t). \tag{5}$$

After sampling with sampling period τ_0, (4) can be approximated by[1]

$$\begin{aligned} c_i(l) &= \sum_{m=1}^{l} \beta_i(m)\tau_0 + \theta_i^0 \\ &= l\tau_0 + \underbrace{\sum_{m=1}^{l-1} [\beta_i(m) - 1]\tau_0 + \theta_i^0}_{\vartheta_i(l-1)} + [\beta_i(l) - 1]\tau_0 \\ &= l\tau_0 + \vartheta_i(l-1) + [\beta_i(l) - 1]\tau_0 \end{aligned} \tag{6}$$

where $\vartheta_i(l)$ and $\beta_i(l)$ are the accumulated clock offset and instantaneous clock skew at the lth sample, respectively.

In order to achieve global clock synchronization, all $c_i(l)$ must be adjusted to be a common value. Without loss of generality, suppose S_1 is selected as the reference node with accurate clock (i.e., $\beta_1(l) = 1$ and $\vartheta_1(l) = 0$), then based on (6) the task of global clock synchronization is to track time-varying clock skews $\{\beta_i(l)\}_{i=2}^N$ and accumulated offsets $\{\vartheta_i(l)\}_{i=2}^N$ with respect to the reference node. Before presenting the distributed tracking algorithm, we first set up the clock skew and accumulated clock offset evolution models, and then localized timestamp measurement model.

Remark 1: If the clock skew is not time-varying, (4) can be written as: $c_i(t) = \beta_i t + \theta_i^0$, which is the first order model widely used in the literature [9], [13], [21], [23].

[1]The same symbols $c_i(\cdot)$ and $\beta_i(\cdot)$ are used for both continuous and discrete quantities but t and τ are reserved exclusively for continuous time argument.

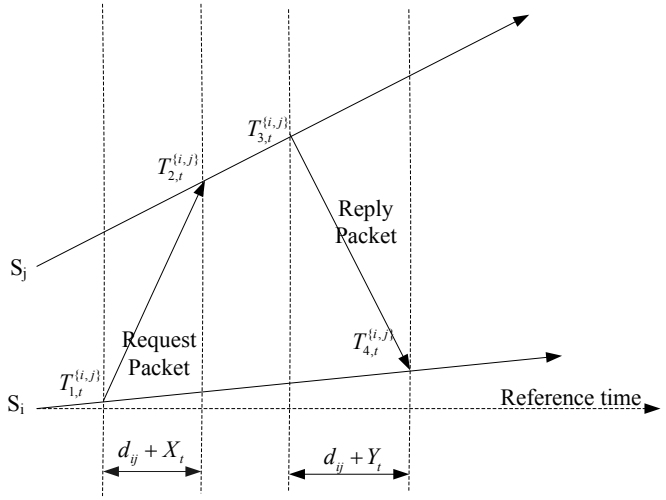


Fig. 2: Two-way time-stamp exchange between nodes i and j at standard time t.

A. Accumulated Clock Offset and Skew Evolution Model

After sampling (5), we can obtain: $\beta_i(l) = \xi_i + \sqrt{p_i}B^{'}(l)$, where $B^{'}(l)$ stands for the derivative of $B(t)$ at the lth sample time. Since the derivative of standard *Wiener* process is Gaussian white noise, we can further take the Gauss-Markov model to describe the time-varying skew of node S_i:

$$\beta_i(l) = \beta_i(l-1) + u_i(l), \tag{7}$$

where $u_i(l) = \sqrt{p_i}[B^{'}(l) - B^{'}(l-1)]$ is Gaussian noise with mean 0 and variance $\sigma_{u_i}^2 = 2p_i$.

On the other hand, based on the definition of accumulated clock offset in (6), we can rewrite $\vartheta_i(l)$ in a recursive form:

$$\vartheta_i(l) = \vartheta_i(l-1) + (\beta_i(l) - 1)\tau_0. \tag{8}$$

Substituting (7) into (8) gives:

$$\vartheta_i(l) = \vartheta_i(l-1) + \tau_0 \cdot \beta_i(l-1) + \tau_0 \cdot u_i(l) - \tau_0. \tag{9}$$

Defining $\mathbf{x}_i(l) = [\beta_i(l)\ \vartheta_i(l)]^T$ and combining (7) and (9), the state evolution model for clock parameters of S_i can be written in a matrix form:

$$\mathbf{x}_i(l) = \begin{bmatrix} 1 & 0 \\ \tau_0 & 1 \end{bmatrix} \mathbf{x}_i(l-1) + \begin{bmatrix} u_i(l) \\ \tau_0 u_i(l) \end{bmatrix} + \begin{bmatrix} 0 \\ -\tau_0 \end{bmatrix}. \tag{10}$$

B. Localized Timestamp Measurement Model

In order to establish clock relationship between two neighboring nodes, two-way time-stamp exchange is performed. The time-stamp exchange model between S_i and S_j is shown in Figure 2. In the time-stamp exchange process, node S_i sends a synchronization message to node S_j with its sending time $T_1^{\{i,j\}}$, S_j records its time $T_2^{\{i,j\}}$ at the reception of that message and replies S_i at time $T_3^{\{i,j\}}$. The replied message contains both $T_2^{\{i,j\}}$ and $T_3^{\{i,j\}}$. Then S_i records the reception time of S_j's reply as $T_4^{\{i,j\}}$. Since the total time elapsed in one round of time-stamp exchange is very small [5], we denote the set of time stamps in one round of message exchange as $\left\{T_{1,t}^{\{i,j\}}, T_{2,t}^{\{i,j\}}, T_{3,t}^{\{i,j\}}, T_{4,t}^{\{i,j\}}\right\}$, where t is the reference time at the message exchange, and clock parameters do not change within one round of time-stamp exchange.

Now, expressing the clock model (4) in terms of reference time and accumulated clock offset as:

$$\begin{aligned} c_i(t) &= \int_0^t \beta_i(\tau)d\tau + \theta_i^0 = t + \int_0^t [\beta_i(\tau) - 1]d\tau + \theta_i^0 \\ &= t + \vartheta_i(t). \end{aligned} \tag{11}$$

With (11), the above time-stamp exchange procedure can be modeled as:

$$T_{2,t}^{\{i,j\}} - \vartheta_j(t) = T_{1,t}^{\{i,j\}} - \vartheta_i(t) + d_{ij} + X_t^{\{i,j\}} \tag{12}$$

$$T_{3,t}^{\{i,j\}} - \vartheta_j(t) = T_{4,t}^{\{i,j\}} - \vartheta_i(t) - d_{ij} - Y_t^{\{i,j\}} \tag{13}$$

where d_{ij} stands for the fixed portion of message delay between S_i and S_j; $X_t^{\{i,j\}}$ and $Y_t^{\{i,j\}}$ are variable portions of the message delay. Considering $X_t^{\{i,j\}}$ and $Y_t^{\{i,j\}}$ are due to numerous independent random processes, it is assumed that $X_t^{\{i,j\}}$ and $Y_t^{\{i,j\}}$ are independent and identically distributed (i.i.d.) Gaussian random variables with zero mean and variance σ^2, and this assumption was experimentally verified in [4].

Adding (12) to (13), defining $V_t^{\{i,j\}} \triangleq X_t^{\{i,j\}} - Y_t^{\{i,j\}}$, $T_{s,t}^{\{i,j\}} \triangleq T_{1,t}^{\{i,j\}} + T_{4,t}^{\{i,j\}}$ and $T_{r,t}^{\{i,j\}} \triangleq T_{2,t}^{\{i,j\}} + T_{3,t}^{\{i,j\}}$, and after sampling, we obtain the discrete-time localized measurement model as:

$$T_{r,l}^{\{i,j\}} - T_{s,l}^{\{i,j\}} = 2\vartheta_j(l) - 2\vartheta_i(l) + V_l^{\{i,j\}}, \tag{14}$$

where l is the sample index. Stacking (14) for all $j \in \mathcal{N}_i$ and defining $\mathbf{x}(l) = [\mathbf{x}_2^T(l)\ \mathbf{x}_3^T(l)\ \cdots\ \mathbf{x}_N^T(l)]^T$, we have

$$\mathbf{z}_{i,l} = \mathbf{C}_{i,l}\mathbf{x}(l) + \mathbf{v}_i(l), \tag{15}$$

where $\mathbf{z}_{i,l}(j) = T_{r,l}^{\{i,\mathcal{N}_i(j)\}} - T_{s,l}^{\{i,\mathcal{N}_i(j)\}}$ with $j \in \{1, \cdots, \lambda_i\}$ ($\lambda_i = |\mathcal{N}_i|$ is the number of neighbors of S_i), and the elements of $\mathbf{C}_{i,l} \in \mathbb{R}^{\lambda_i \times 2(N-1)}$ are represented as:

$$\mathbf{C}_{i,l}(j,m) = \begin{cases} -2 & \text{if } m = 2i-2, \\ 2 & \text{if } m = 2\mathcal{N}_i(j) - 2, \\ 0 & \text{otherwise,} \end{cases}$$

with $j \in \{1, \cdots, \lambda_i\}$ and $m \in \{1, \cdots, 2(N-1)\}$. Furthermore, $\mathbf{v}_i(l) \in \mathbb{R}^{\lambda_i \times 1}$ is the measurement noise, and clearly its mean is zero and the covariance is $\mathbf{R}_i = \mathrm{E}\left[\mathbf{v}_i(l)\mathbf{v}_i^T(l)\right] = 2\sigma^2\mathbf{I}_{\lambda_i}$.

The measurement model (15) can also be described in terms of local state vector as

$$\mathbf{z}_{i,l} = \widetilde{\mathbf{C}}_{i,l}\mathbf{x}_{\mathcal{N}_{[i]}}(l) + \mathbf{v}_i(l), \tag{16}$$

where $\mathbf{x}_{\mathcal{N}_{[i]}}(l) = \mathbf{\Lambda}_i\mathbf{x}(l)$ is the clock parameters vector of local subsystem including the node S_i and all its neighbors (except reference node), and $\widetilde{\mathbf{C}}_{i,l} = \mathbf{C}_{i,l}\mathbf{\Lambda}_i^T$ is the reduced matrix excluding columns of $\mathbf{C}_{i,l}$ corresponding to non-neighbors of node i, and $\mathbf{\Lambda}_i$ is the selection matrix with $\mathbf{\Lambda}_i^T\mathbf{\Lambda}_i = \mathbf{I}$.

III. Distributed Clock Parameters Tracking Algorithm

In wireless sensor network, clock skews and offsets are time-varying. This calls for frequent resynchronization. In this section we will design distributed Kalman filer (DKF) to track

the clock parameters. Since clock synchronization should only occupy a small portion of the resource in WSNs, it is assumed that one round of Kalman filter is executed every Δ unit of τ_0 with $\Delta \gg 1$.

A. Distributed Kalman Filtering (DKF) Approach

Define $l_k = \Delta k$ and based on (10) and (16), we can obtain the state-space equations for local subsystem of node S_i as

$$\begin{cases} \mathbf{x}_i(l_k) = \mathbf{A}_i \mathbf{x}_i(l_{k-1}) + \mathbf{w}_i(l_k) + \mathbf{b}_i \\ \mathbf{z}_{i,l_k} = \widetilde{\mathbf{C}}_{i,l_k} \mathbf{x}_{\mathcal{N}_{[i]}}(l_k) + \mathbf{v}_i(l_k) \end{cases} \tag{17}$$

with

$$\mathbf{A}_i = \begin{bmatrix} 1 & 0 \\ \Delta\tau_0 & 1 \end{bmatrix},\ \mathbf{w}_i(l_k) = \begin{bmatrix} \sqrt{p_i}[B^{'}(l_k) - B^{'}(l_{k-1})] \\ \sum_{m=l_{k-1}+1}^{l_k} (l_k - m + 1)\tau_0 u_i(m) \end{bmatrix}$$

and $\mathbf{b}_i = [0\ \ -\Delta\tau_0]^T$, where $\mathbf{w}_i(l_k)$ can be interpreted as the random disturbance in the evolution equation. It clearly has a zero mean and the covariance matrix is:

$$\boldsymbol{Q}_i(l_k) = \mathrm{E}\left[\mathbf{w}_i(l_k)\mathbf{w}_i^T(l_k)\right] = \sigma_{u_i}^2 \begin{bmatrix} 1 & 0 \\ 0 & \frac{\Delta(1+\Delta)(2\Delta+1)}{6}\tau_0^2 \end{bmatrix}. \tag{18}$$

The goal is to track the time-varying clock skews $\{\beta_i(l_k)\}_{i=2}^N$ and accumulated offsets $\{\vartheta_i(l_k)\}_{i=2}^N$, based on local information (17). The optimal solution is the Kalman filter, which requires gathering of (17) for all S_i in a central processing unit, resulting the dynamic equation:

$$\begin{cases} \mathbf{x}(l_k) = \mathbf{A}\mathbf{x}(l_{k-1}) + \mathbf{w}(l_k) + \mathbf{b} \\ \mathbf{z}_{l_k} = \mathbf{C}_{l_k} \cdot \mathbf{x}(l_k) + \mathbf{v}(l_k) \end{cases} \tag{19}$$

where $\mathbf{A} = \mathrm{diag}(\mathbf{A}_2, \cdots, \mathbf{A}_N)$; $\mathbf{w}(l_k) = [\mathbf{w}_2^T(l_k)\ \cdots\ \mathbf{w}_N^T(l_k)]^T \in \mathbb{R}^{2\times(N-1)}$ with $\mathrm{E}[\mathbf{w}(l_k)] = \mathbf{0}$ and $\mathbf{Q}(l_k) = \mathrm{E}\left[\mathbf{w}(l_k)\mathbf{w}^T(l_k)\right] = \mathrm{diag}(\mathbf{Q}_2(l_k), \cdots, \mathbf{Q}_N(l_k))$; $\mathbf{b} = [\mathbf{b}_2^T\ \cdots\ \mathbf{b}_N^T]^T$; $\mathbf{z}_{l_k} = [\mathbf{z}_{2,l_k}^T\ \cdots\ \mathbf{z}_{N,l_k}^T]^T$; $\mathbf{C}_{l_k} = [\mathbf{C}_{2,l_k}^T\ \cdots\ \mathbf{C}_{N,l_k}^T]^T \in \mathbb{R}^{\lambda\times 2(N-1)}$ ($\lambda = \sum_{i=2}^N \lambda_i$); and $\mathbf{v}(l_k) = [\mathbf{v}_2^T(l_k)\ \cdots\ \mathbf{v}_N^T(l_k)]^T \sim \mathcal{N}(\mathbf{0},\ \mathbf{R})$ with $\mathbf{R} = 2\sigma^2\mathbf{I}_\lambda$.

Based on (19), the standard Kalman filter is

$$\text{Prediction step}: \quad \widehat{\mathbf{x}}(l_k|l_{k-1}) = \mathbf{A}\widehat{\mathbf{x}}(l_{k-1}|l_{k-1}) + \mathbf{b} \tag{20}$$

$$\text{Update step}: \quad \widehat{\mathbf{x}}(l_k|l_k) = \widehat{\mathbf{x}}(l_k|l_{k-1}) + \mathbf{K}(l_k) \cdot (\mathbf{z}_{l_k} - \mathbf{C}_{l_k}\widehat{\mathbf{x}}(l_k|l_{k-1})). \tag{21}$$

The covariance matrix $\mathbf{P}(l_k|l_{k-1})$ of prediction-step and covariance matrix $\mathbf{P}(l_k|l_k)$ of update-step are given by:

$$\mathbf{P}(l_k|l_{k-1}) = \mathbf{A}\mathbf{P}(l_{k-1}|l_{k-1})\mathbf{A}^T + \mathbf{Q}(l_k) \tag{22}$$

$$\mathbf{P}(l_k|l_k) = (\mathbf{I} - \mathbf{K}(l_k)\mathbf{C}_{l_k})\mathbf{P}(l_k|l_{k-1})(\mathbf{I} - \mathbf{C}_{l_k}^T\mathbf{K}^T(l_k)) + \mathbf{K}(l_k)\mathbf{R}\mathbf{K}^T(l_k) \tag{23}$$

where the global Kalman gain $\mathbf{K}(l_k)$ is chosen to minimize the $\mathrm{Tr}(\mathbf{P}(l_k|l_k))$.

Since the centralized optimal solution is not convenient in large scale systems, now we decompose (20) and (21) into distributed form. From (20) and (21), it is noticed that the state vector $\mathbf{x}_i(l_k)$ in $\mathbf{x}(l_k)$, observation $\mathbf{z}_{i,l_k}$ in $\mathbf{z}_{l_k}$, system matrices $\mathbf{A}_i$ in $\mathbf{A}$, $\widetilde{\mathbf{C}}_{i,l_k}$ in $\mathbf{C}_{l_k}$, $\mathbf{b}_i$ in $\mathbf{b}$ are all localized. Only the global Kalman gain $\mathbf{K}(l_k)$ is not localized and has to be computed in centralized way. In order to make the optimal solution decomposed into distributed form, we enforce an additional constraint that $\mathbf{K}(l_k)$ is a block diagonal matrix. With this additional constraint, the standard KF can be decomposed into following distributed form:

$$\begin{cases} \widehat{\mathbf{x}}_i(l_k|l_{k-1}) = \mathbf{A}_i\widehat{\mathbf{x}}_i(l_{k-1}|l_{k-1}) + \mathbf{b}_i \\ \widehat{\mathbf{x}}_i(l_k|l_k) = \widehat{\mathbf{x}}_i(l_k|l_{k-1}) + \mathbf{K}_i(l_k)(\mathbf{z}_{i,l_k} - \widetilde{\mathbf{C}}_{i,l_k}\widehat{\mathbf{x}}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})) \end{cases} \tag{24}$$

where $\mathbf{K}_i(l_k)$ is the local Kalman gain chosen as:

$$\mathbf{K}(l_k) = \arg\min_{\mathbf{K}(l_k)} \mathrm{Tr}\ \mathbf{P}(l_k|l_k) \quad \text{s.t.} \quad \mathbf{K}(l_k) = \sum_{i=2}^N \mathbf{U}_i^T\mathbf{K}_i(l_k)\boldsymbol{\Omega}_i, \tag{25}$$

where $\mathbf{U}_i = \left[\mathbf{0}_{2\times 2(i-2)}\ \mathbf{I}_{2\times 2}\ \mathbf{0}_{2\times 2(N-i-1)}\right]$ and $\boldsymbol{\Omega}_i = [\mathbf{0}_{\lambda_i\times\sum_{j=2}^{i-1}\lambda_j}\ \mathbf{I}_{\lambda_i\times\lambda_i}\ \mathbf{0}_{\lambda_i\times\sum_{j=i}^{N}\lambda_j}]$ are used to enforce the block diagonal structure of $\mathbf{K}(l_k)$.

To solve this optimization problem, the covariance matrix $\mathbf{P}(l_k|l_k)$ in (23) is written as:

$$\mathbf{P}(l_k|l_k) = \mathbf{L}_{11} + \mathbf{K}(l_k)\mathbf{L}_{12} + \mathbf{L}_{21}\mathbf{K}(l_k)^T + \mathbf{K}(l_k)\mathbf{L}_{22}\mathbf{K}(l_k)^T, \tag{26}$$

where $\mathbf{L}_{11} = \mathbf{P}(l_k|l_{k-1})$, $\mathbf{L}_{12} = -\mathbf{P}(l_k|l_{k-1})\mathbf{C}_{l_k}{}^T$, $\mathbf{L}_{21} = -\mathbf{C}_{l_k}\mathbf{P}(l_k|l_{k-1})$, and $\mathbf{L}_{22} = \mathbf{C}_{l_k}\mathbf{P}(l_k|l_{k-1})\mathbf{C}_{l_k}^T + \mathbf{R}$. With the matrix equality $\mathrm{Tr}(\mathbf{A}) = \mathrm{Tr}(\mathbf{A}^T)$ and the symmetry of $\mathbf{P}(l_k|l_k) = \mathbf{P}^T(l_k|l_k)$, $\mathrm{Tr}\ \mathbf{P}(l_k|l_k)$ becomes

$$\mathrm{Tr}\ \mathbf{P}(l_k|l_k) = \mathrm{Tr}(\mathbf{L}_{11}) + 2\mathrm{Tr}\left[\mathbf{L}_{21}\mathbf{K}^T(l_k)\right] + \mathrm{Tr}\left[\mathbf{K}(l_k)\mathbf{L}_{22}\mathbf{K}^T(l_k)\right].$$

Now differentiating $\mathrm{Tr}(\mathbf{P}(l_k|l_k))$ with respect to $\mathbf{K}_i(l_k)$ gives:

$$\mathrm{vec}\left[\frac{d\ \mathrm{Tr}\mathbf{P}(l_k|l_k)}{d\ \mathbf{K}_i(l_k)}\right] = \frac{d\ \mathrm{Tr}\mathbf{P}(l_k|l_k)}{d\ \mathrm{vec}[\mathbf{K}_i(l_k)]} = \frac{d\ \mathrm{Tr}\mathbf{P}(l_k|l_k)}{d\ \mathrm{vec}[\mathbf{K}(l_k)]}\frac{d\ \mathrm{vec}[\mathbf{K}(l_k)]}{d\ \mathrm{vec}[\mathbf{K}_i(l_k)]}. \tag{27}$$

With the matrix differentiation rules [29], the derivatives in (27) are given by

$$\begin{cases} \dfrac{d\ \mathrm{Tr}\mathbf{P}(l_k|l_k)}{d\ \mathrm{vec}[\mathbf{K}(l_k)]} = 2\left(\mathrm{vec}\left[\mathbf{L}_{12} + \mathbf{K}(l_k)\mathbf{L}_{22}\right]\right)^T \\ \dfrac{d\ \mathrm{vec}[\mathbf{K}(l_k)]}{d\ \mathrm{vec}[\mathbf{K}_i(l_k)]} = \left(\boldsymbol{\Omega}_i \otimes \mathbf{U}_i\right)^T \end{cases} \tag{28}$$

Putting (28) into (27), we obtain

$$\frac{d\ \mathrm{Tr}\mathbf{P}(l_k|l_k)}{d\ \mathrm{vec}[\mathbf{K}_i(l_k)]} = 2\left((\boldsymbol{\Omega}_i \otimes \mathbf{U}_i)\mathrm{vec}\left[\mathbf{L}_{12} + \mathbf{K}(l_k)\mathbf{L}_{22}\right]\right)^T = 2\left(\mathrm{vec}[\mathbf{U}_i(\mathbf{L}_{12} + \mathbf{K}(l_k)\mathbf{L}_{22})\boldsymbol{\Omega}_i^T]\right)^T.$$

The optimal $\mathbf{K}_i(l_k)$ can be obtained by setting the result to zero: $\mathbf{U}_i(\mathbf{L}_{12} + \mathbf{K}(l_k)\mathbf{L}_{22})\boldsymbol{\Omega}_i^T = \mathbf{0}$. Using (25) leads to

$$\mathbf{0} = \mathbf{U}_i(\mathbf{L}_{12})\boldsymbol{\Omega}_i^T + \sum_{j=2}^N \left(\underbrace{\mathbf{U}_i\mathbf{U}_j^T}_{=0\ if\ i\neq j}\mathbf{K}_j(l_k)\boldsymbol{\Omega}_j\mathbf{L}_{22}\boldsymbol{\Omega}_i^T\right) = \mathbf{U}_i(\mathbf{L}_{12})\boldsymbol{\Omega}_i^T + \mathbf{K}_i(l_k)\boldsymbol{\Omega}_i\mathbf{L}_{22}\boldsymbol{\Omega}_i^T. \tag{29}$$

Therefore, the optimal Kalman gain $\mathbf{K}_i(l_k)$ can be solved to be:

$$\mathbf{K}_i(l_k) = -\left[\mathbf{U}_i\mathbf{P}(l_k|l_{k-1})\mathbf{C}_{i,l_k}^T\right]\left[\mathbf{C}_{i,l_k}\mathbf{P}(l_k|l_{k-1})\mathbf{C}_{i,l_k}^T + \mathbf{R}_i\right]^{-1}. \tag{30}$$

To fully distribute the calculation of the Kalman gain $\mathbf{K}_i(l_k)$ in (30), it can be rewritten in the following alternative form:

$$\mathbf{K}_i(l_k) = -\left[\mathbf{p}_{[i]}(l_k|l_{k-1})\widetilde{\mathbf{C}}_{i,l_k}^T\right]\left[\widetilde{\mathbf{C}}_{i,l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})\widetilde{\mathbf{C}}_{i,l_k}^T + \mathbf{R}_i\right]^{-1} \tag{31}$$

where $\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}) = \mathbf{\Lambda}_i\mathbf{P}(l_k|l_{k-1})(\mathbf{\Lambda}_i)^T$ is the covariance matrix of the estimate $\mathbf{x}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})$ in local subsystem and $\mathbf{p}_{[i]}(l_k|l_{k-1})$ is the rows of $\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})$ corresponding to $\mathbf{x}_i$.

From (31), we can notice that $\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})$ is required. Now based on (22), left multiplying both sides by $\mathbf{\Lambda}_i$ and right multiplying both sides by $(\mathbf{\Lambda}_i)^T$, and with the matrix equality $\mathbf{\Lambda}_i(\mathbf{\Lambda}_i)^T = (\mathbf{\Lambda}_i)^T\mathbf{\Lambda}_i = \mathbf{I}$, we can obtain local update of $\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})$ as

$$\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}) = \mathbf{A}_{\mathcal{N}_{[i]}}\mathbf{P}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})\mathbf{A}_{\mathcal{N}_{[i]}}^T + \mathbf{Q}_{\mathcal{N}_{[i]}}(l_k), \tag{32}$$

where $\mathbf{P}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})$ is the covariance matrix of the estimate $\mathbf{x}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})$ in local subsystem, $\mathbf{A}_{\mathcal{N}_{[i]}} = \text{diag}(\mathbf{A}_{m_1}, \cdots, \mathbf{A}_{m_j}, \cdots, \mathbf{A}_{m_{(\lambda_i+1)}})$, and $\mathbf{Q}_{\mathcal{N}_{[i]}} = \text{diag}(\mathbf{Q}_{m_1}, \cdots, \mathbf{Q}_{m_j}, \cdots, \mathbf{Q}_{m_{(\lambda_i+1)}})$, where $m_j \in \{\mathcal{N}_i, i\}$. Without loss of generality, it is assumed that $m_1 < m_2 < \cdots < m_{(\lambda_i+1)}$.

On the other hand, (32) depends on $\mathbf{P}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})$, which can be obtained by first expressing (23) in its alternative form: $\mathbf{P}(l_k|l_k) = \mathbf{P}(l_k|l_{k-1}) - \mathbf{K}(l_k)\mathbf{C}_{l_k}\mathbf{P}(l_k|l_{k-1})$ [17], and then left multiplying both sides by $\mathbf{\Lambda}_i$ and right multiplying both sides by $(\mathbf{\Lambda}_i)^T$, so the local update of $\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_k)$ is given by

$$\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_k) = \mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}) - \mathbf{K}_{\mathcal{N}_{[i]}}(l_k)\mathbf{C}_{\mathcal{N}_{[i]},l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}), \tag{33}$$

where the Kalman gain $\mathbf{K}_{\mathcal{N}_{[i]}}(l_k) = \text{diag}(\mathbf{K}_{m_1}(l_k), \cdots, \mathbf{K}_{m_j}(l_k), \cdots, \mathbf{K}_{m_{(\lambda_i+1)}}(l_k))$, and $\mathbf{C}_{\mathcal{N}_{[i]},l_k} = [\mathbf{C}_{m_1,l_k}^T, \cdots, \mathbf{C}_{m_j,l_k}^T, \cdots, \mathbf{C}_{m_{(\lambda_i+1)},l_k}^T]^T$.

B. Asynchronous Implementation, Handling Node Failure and New Neighbors

In practice, due to the broadcasting nature and the half-duplex operation of wireless nodes, some data packets may loss. Updating one's estimate only after getting information from all neighbors may not be advisable. But the proposed algorithm can be easily modified to work in an asynchronous way. More specifically, nodes will wait for a "time-out" period for receiving update information from their neighbors. If update information from some neighbors (say node j) does not arrive in this period of time, the previously stored estimate $\hat{\mathbf{x}}_j$ and its covariance matrix $\mathbf{P}_j$ will be used instead.

On the other hand, the proposed algorithm can also easily handle node failure during tracking operation. If node j is a neighbor of node i, and suddenly fails, it can simply be removed from the subsystem of node i, and the estimation updates can be carried out in the new subsystem. More specifically, local matrices $\mathbf{A}_{\mathcal{N}_{[i]}}$, $\mathbf{Q}_{\mathcal{N}_{[i]}}$, $\mathbf{C}_{\mathcal{N}_{[i]},l_k}$, $\widetilde{\mathbf{C}}_{i,l_k}$, local estimate $\widehat{\mathbf{x}}_{\mathcal{N}_{[i]}}$, local Kalman gain $\mathbf{K}_{\mathcal{N}_{[i]}}$, and local covariance matrix $\mathbf{P}_{\mathcal{N}_{[i]}}$ can be modified by deleting the rows and columns that correspond to node j. If the node j go online again, the connection between node i and j resumes to work, the local matrices will be modified by inserting rows and columns correspond to node j. More specifically, we modify $\mathbf{P}_{\mathcal{N}_{[i]}}$ as $\widetilde{\mathbf{P}}_{\mathcal{N}_{[i]}} = \text{diag}(\mathbf{P}_{\mathcal{N}_{[i]}}, \mathbf{P}_j)$, where $\mathbf{P}_j$ is node j's covariance matrix (possibly from previous estimate). For the case of new neighbors, if a new node m joins the neighborhood of node i, a new connection is established between node i and m. The local matrices can be updated as in the case of a missing node resume working. The only difference is that $\mathbf{P}_m$ is set as $\delta^{-1}\mathbf{I}_{2\times 2}$ with δ being a small value due to the absence of prior information about the new node.

The distributed accumulated clock offset $\vartheta_i(l_k)$ and skew $\beta_i(l_k)$ tracking algorithm is summarized in Algorithm 1. This algorithm is localized, implying that the nodes in WSN communicate only with their neighbors to obtain the desired results. This localized algorithm can also work under the conditions of node failures, packet loss, and new neighbors, and the communication overhead scales well with increasing network size.

C. Computational Complexity Analysis

The computational complexity of one iteration of the distributed tracking algorithm at node i depends mainly on the costs of four terms: $\mathbf{A}_{\mathcal{N}_{[i]}}\mathbf{P}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})\mathbf{A}_{\mathcal{N}_{[i]}}^T$ at covariance prediction, $\widetilde{\mathbf{C}}_{i,l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})\widetilde{\mathbf{C}}_{i,l_k}^T$, $\left[\widetilde{\mathbf{C}}_{i,l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})\widetilde{\mathbf{C}}_{i,l_k}^T + \mathbf{R}_i\right]^{-1}$ at Kalman filter gain calculations and $\mathbf{K}_{\mathcal{N}_{[i]}}(l_k)\mathbf{C}_{\mathcal{N}_{[i]},l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})$ at covariance update based on observations. Since the computational complexity order of these four terms are $\mathbb{O}(8(\lambda_i+1)^3)$, $\mathbb{O}(2\lambda_i(\lambda_i+1)(3\lambda_i+2))$, $\mathbb{O}(\lambda_i^3)$, and $\mathbb{O}(4(\lambda_i+1)^2[\lambda_i^2+2(\lambda_i+1)])$, respectively, the total cost of one iteration at node i can be approximately as $\mathbb{O}(4\lambda_i^4 + 31\lambda_i^3 + 66\lambda_i^2 + 52\lambda_i + 16)$.

IV. Low-Complexity Distributed Clock Parameters Initialization

It is noticed that in the distributed clock parameters tracking algorithm, we need initial values and initial covariance matrices to start the distributed Kalman filtering. In the absence of prior information, we can set $\widehat{\mathbf{x}}_i(0|0) = [1\ 0]^T$ and $\mathbf{P}(0|0) = \delta^{-1}\mathbf{I}$ with δ being a small value. However, such initialization may result in a slow convergence speed. Furthermore, this method involves lots of matrix multiplications and inversions. In this section, we propose a low-complexity distributed clock parameters initialization algorithm to obtain good initial values $\widehat{\mathbf{x}}_i(0|0)$ and the covariance matrix $\mathbf{P}_{\mathcal{N}_{[i]}}(0|0)$.

Since the accumulated offsets $\vartheta_i(l)$ varies sample by sample, it is not suitable for batch mode estimation. On the other hand, since the clock skews vary relatively slow, and can be considered constant if the elapse time of batch mode estimation is small. Therefore, in the initialization, we use the clock model mentioned in *Remark 1*: $c_i(t) = \beta_i t + \theta_i^0$, since the procedure of batch mode estimation is very short. Equating this model with (4), we have

$$c_i(l) - \vartheta_i(l) = \frac{c_i(l) - \theta_i^0}{\beta_i}. \tag{34}$$

Algorithm 1 Distributed accumulated offset and skew tracking algorithm at node i

Initialization:
Initialize with $\widehat{\mathbf{x}}_i(0|0)$ and $\mathbf{P}_{\mathcal{N}_{[i]}}(0|0)$.
Broadcast variance $\sigma_{u_i}^2$ to neighboring sensors;
Iteration:
for $k = 1, 2, \cdots$ **do**
  Run two-way timestamp exchange with neighbors and obtain new measurements $\mathbf{z}_{i,l_k}$;
  construct $\mathbf{A}_{\mathcal{N}_{[i]}}$, $\mathbf{Q}_{\mathcal{N}_{[i]}}$, $\widetilde{\mathbf{C}}_{i,l_k}$ and $\mathbf{P}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})$;
  Prediction step:
  Calculate

$$\begin{cases} \widehat{\mathbf{x}}_i(l_k|l_{k-1}) = \mathbf{A}_i\widehat{\mathbf{x}}_i(l_{k-1}|l_{k-1}) + \mathbf{b}_i \\ \mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}) = \mathbf{A}_{\mathcal{N}_{[i]}}\mathbf{P}_{\mathcal{N}_{[i]}}(l_{k-1}|l_{k-1})\mathbf{A}_{\mathcal{N}_{[i]}}^T + \mathbf{Q}_{\mathcal{N}_{[i]}} \end{cases}$$

  and

$$\mathbf{K}_i(l_k) = -\left[\mathbf{p}_{[i]}(l_k|l_{k-1})\widetilde{\mathbf{C}}_{i,l_k}^T\right]\left[\widetilde{\mathbf{C}}_{i,l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})\widetilde{\mathbf{C}}_{i,l_k}^T + \mathbf{R}_i\right]^{-1}$$

  Broadcast $\widehat{\mathbf{x}}_i(l_k|l_{k-1})$, $\mathbf{K}_i(l_k)$ and $\widetilde{\mathbf{C}}_{i,l_k}$ to neighboring sensors;
  Update step:
  Construct $\widehat{\mathbf{x}}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})$, $\mathbf{K}_{\mathcal{N}_{[i]}}(l_k)$ and $\mathbf{C}_{\mathcal{N}_{[i]},l_k}$;
  Update $\widehat{\mathbf{x}}_i(l_k|l_k)$ and $\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_k)$ according to

$$\begin{cases} \widehat{\mathbf{x}}_i(l_k|l_k) = \widehat{\mathbf{x}}_i(l_k|l_{k-1}) + \mathbf{K}_i(l_k)\left(\mathbf{z}_{i,l_k} - \widetilde{\mathbf{C}}_{i,l_k}\widehat{\mathbf{x}}_{\mathcal{N}_{[i]}}(l_k|l_{k-1})\right) \\ \mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_k) = \mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}) - \mathbf{K}_{\mathcal{N}_{[i]}}(l_k)\mathbf{C}_{\mathcal{N}_{[i]},l_k}\mathbf{P}_{\mathcal{N}_{[i]}}(l_k|l_{k-1}) \end{cases}$$

end for

Applying (34) into (12) and (13), and recognizing that $T_{1,t}^{i,j}, T_{4,t}^{i,j}$ are the clock reading of node i (i.e., $c_i(t)$) while $T_{2,t}^{i,j}, T_{3,t}^{i,j}$ are that of node j, the measurement model (14) can be written as

$$1/\beta_j \cdot \left[T_{r,l}^{\{i,j\}} - 2\theta_j^0\right] = 1/\beta_i \cdot \left[T_{s,l}^{\{i,j\}} - 2\theta_i^0\right] + V_l^{\{i,j\}}, \quad (35)$$

where $l \in \{1, \cdots, L_1\}$, $V_l^{\{i,j\}}$ are i.i.d. Gaussian random variables with zero mean and variance $2\sigma^2$. Based on (35) and suppose we have L_1 round of time-stamp exchanges between any pair of nodes in the network, the initial parameters estimation can be considered as the following optimization problem:

$$\min_{\theta_i^0, \beta_i} \sum_{l=1}^{L_1}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left(1/\beta_i \cdot \left[T_{s,l}^{\{i,j\}} - 2\theta_i^0\right] - 1/\beta_j \cdot \left[T_{r,l}^{\{i,j\}} - 2\theta_j^0\right]\right)^2, \quad (36)$$

However, we can easily notice that the above problem is not a convex optimization problem, thus it is difficult to obtain the global optimal solution.

On the other hand, with a simple transformation, $\alpha_i = 1/\beta_i$, $\gamma_i = \theta_i^0/\beta_i$, and since node 1 is selected as the reference, (i.e., $[\alpha_1, \gamma_1] = [1, 0]$), (36) can be transformed into

$$\begin{aligned} &\min_{\alpha_i, \gamma_i} LF\left(\{\alpha_i\}_{i=2}^N, \{\gamma_i\}_{i=2}^N\right) \\ &= \min_{\alpha_i, \gamma_i} \sum_{l=1}^{L_1}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\left(\alpha_i T_{s,l}^{\{i,j\}} - 2\gamma_i - \alpha_j T_{r,l}^{\{i,j\}} + 2\gamma_j\right)^2, \end{aligned} \quad (37)$$

and we can see that (37) is a convex Quadratic problem. We can then apply coordinate descent (CD) algorithm to iteratively minimize (37), which provides a fully distributed algorithm. More specifically, differentiating the objective function (37) w.r.t. variables α_i and γ_i respectively gives

$$\begin{aligned} \frac{\partial LF}{\partial \alpha_i} = \sum_{l=1}^{L_1}\sum_{j\in\mathcal{N}_i}\Big\{&2\left(\alpha_i T_{s,l}^{\{i,j\}} - 2\gamma_i - \alpha_j T_{r,l}^{\{i,j\}} + 2\gamma_j\right)\cdot T_{s,l}^{\{i,j\}} \\ &+2\left(\alpha_i T_{r,l}^{\{j,i\}} - 2\gamma_i - \alpha_j T_{s,l}^{\{j,i\}} + 2\gamma_j\right)\cdot T_{r,l}^{\{j,i\}}\Big\} \end{aligned} \quad (38)$$

$$\begin{aligned} \frac{\partial LF}{\partial \gamma_i} = \sum_{l=1}^{L_1}\sum_{j\in\mathcal{N}_i}\Big\{&4\left(\alpha_j T_{r,l}^{\{i,j\}} - 2\gamma_j - \alpha_i T_{s,l}^{\{i,j\}} + 2\gamma_i\right) \\ &+4\left(\alpha_j T_{s,l}^{\{j,i\}} - 2\gamma_j - \alpha_i T_{r,l}^{\{j,i\}} + 2\gamma_i\right)\Big\}. \end{aligned} \quad (39)$$

Setting (38) and (39) to zero yields the iteration formulas (40) and (41) shown at the bottom of the page. Notice that for $j = 1$, the variables α_1 and γ_1 correspond to that of reference node S_1, thus we have $\alpha_1 = 1, \gamma_1 = 0$ for all the iterations. During the procedure, each node update its estimates α_i and γ_i according to (40) and (41) until convergence. After convergence, the estimate of initial clock offset and clock skew can be calculated by the transformation: $\hat{\beta}_i = 1/\hat{\alpha}_i, \hat{\theta}_i^0 = \hat{\gamma}_i/\hat{\alpha}_i$. Finally, the initial values of accumulated offset and skew can be calculated as: $\hat{\mathbf{x}}_i(0|0) = [\hat{\beta}_i \ \ \hat{\theta}_i^0 + (\hat{\beta}_i - 1)\kappa_i]^T$, where $\kappa_i = [c_i(t_e) - c_i(t_s)]/\hat{\beta}_i$ is the elapsed time for node i initialization.

Since the objective function (37) is strictly convex and continuously differentiable, the coordinate descent based method converges to the global optimal solution. And from [18], [19], the convergence rate is at least linear.

On the other hand, we can further use the Bootstrap technique [20] to estimate the covariance matrix $\mathbf{P}_{\mathcal{N}_{[i]}}(0|0)$ based on L_1 rounds of time-stamp exchange during initialization. Denoting the L_1 rounds of time-stamp measurements

$$\widehat{\alpha}_i^{(m+1)} = \frac{\sum_{l=1}^{L_1}\sum_{j\in\mathcal{N}_i}\left[2\widehat{\gamma}_i^{(m)} - 2\widehat{\gamma}_j^{(m)}\right]\left[T_{s,l}^{\{i,j\}} + T_{r,l}^{\{j,i\}}\right] + \widehat{\alpha}_j^{(m)}\left[T_{r,l}^{\{i,j\}}\cdot T_{s,l}^{\{i,j\}} + T_{s,l}^{\{j,i\}}\cdot T_{r,l}^{\{j,i\}}\right]}{\sum_{l=1}^{L_1}\sum_{j\in\mathcal{N}_i}\left[\left(T_{s,l}^{\{i,j\}}\right)^2 + \left(T_{r,l}^{\{j,i\}}\right)^2\right]} \quad (40)$$

$$\widehat{\gamma}_i^{(m+1)} = \frac{1}{4L_1\cdot\lambda_i}\left\{\sum_{l=1}^{L_1}\sum_{j\in\mathcal{N}_i} 4\widehat{\gamma}_j^{(m)} + \widehat{\alpha}_i^{(m)}\left[T_{s,l}^{\{i,j\}} + T_{r,l}^{\{j,i\}}\right] - \widehat{\alpha}_j^{(m)}\left[T_{r,l}^{\{i,j\}} + T_{s,l}^{\{j,i\}}\right]\right\}. \quad (41)$$

TABLE I: Complexity Comparison During Initialization

Initialization method	pre-computation	per iteration	total cost
DKF	0	$\mathbb{O}(4\lambda_i^4+31\lambda_i^3+66\lambda_i^2+52\lambda_i+16)$	$\mathbb{O}(N_{KF}(4\lambda_i^4+31\lambda_i^3+66\lambda_i^2+52\lambda_i+16))$
CD-BS	$B(6L_1+3\lambda_i L_1)$	$B(9\lambda_i+2)$	$B[6L_1+3\lambda_i L_1+N_{CD}(9\lambda_i+2)]$

$\left\{T_{1,l}^{\{i,j\}},T_{2,l}^{\{i,j\}},T_{3,l}^{\{i,j\}},T_{4,l}^{\{i,j\}}\right\}_{l=1}^{L_1}$ as $\left\{\mathfrak{T}_l^{\{i,j\}}\right\}_{l=1}^{L_1}$, the procedure of Coordinate Descent with Bootstrap (CD-BS) for covariance matrix estimation is illustrated as follows:

- **Step 1.** *Resampling and repetition.* In each node, draw B random samples $\{\mathfrak{S}_1^{\{i,j\}},\cdots,\mathfrak{S}_B^{\{i,j\}}\}$ of size L_1, with replacement, from $\mathfrak{S}^{\{i,j\}}=\{\mathfrak{T}_1^{\{i,j\}},\cdots,\mathfrak{T}_{L_1}^{\{i,j\}}\}$.
- **Step 2.** *Calculation of the bootstrap estimates using CD.* Each node broadcasts B groups of current estimates $\{[\hat{\alpha}_i,\hat{\gamma}_i]_1,\cdots,[\hat{\alpha}_i,\hat{\gamma}_i]_B\}$ to its neighbors. After receiving the estimates, each node update its estimates according to (40) and (41) with the corresponding time-stamp sample $\{\mathfrak{S}_1^{\{i,j\}},\cdots,\mathfrak{S}_B^{\{i,j\}}\}$. This procedure iterates until convergence, and then we obtain a total of B bootstrap estimates $\hat{\mathbf{x}}_{\mathcal{N}_{[i]},1},\ldots,\hat{\mathbf{x}}_{\mathcal{N}_{[i]},B}$.
- **Step 3.** *Estimation of the covariance matrix* $\mathbf{P}_{\mathcal{N}_{[i]}}$. Estimate the covariance matrix of $\hat{\mathbf{x}}_{\mathcal{N}_{[i]}}$ by

$$\mathbf{P}_{\mathcal{N}_{[i]}}(0|0)=\frac{1}{B-1}\sum_{j=1}^{B}\left(\hat{\mathbf{x}}_{\mathcal{N}_{[i]},j}-\frac{1}{B}\sum_{k=1}^{B}\hat{\mathbf{x}}_{\mathcal{N}_{[i]},k}\right)^2.$$

In terms of computational complexity for node i, the method of Coordinate Descent with Bootstrap involves $B(4L_1+\lambda_i L_1)$ additions and $B(2L_1+2\lambda_i L_1)$ multiplications before iterations, and then for each iteration, $4B\lambda_i$ additions and $B(5\lambda_i+2)$ multiplications are required. Assuming that the computational costs of multiplication and addition operations are the same, the total cost for node i can be expressed as $B[6L_1+3\lambda_i L_1+N_{CD}(9\lambda_i+2)]$, where N_{CD} is the number of iterations for the convergence of CD-BS.

The computational complexities of CD-BS and DKF for initialization are listed in Table I for comparison. In case of DKF for initialization, we can set $\Delta=1$, and N_{KF} in Table I is the number of iteration of DKF to reach convergence during initialization. As shown in Table I, the computation of DKF method takes complexity order $\mathbb{O}(\lambda_i^4)$ while that of CD-BS is only $\mathbb{O}(\lambda_i)$. Detail complexity comparison will be presented in simulation section.

V. Bayesian Cramer-Rao Lower Bound

In this section, we derive the centralized Bayesian Cramer-Rao Lower Bound for the accumulated offsets and clock skews estimation which served as a benchmark for the distributed tracking algorithm.

Define $\mathbf{X}_{0:k}=\{\mathbf{x}(0),\mathbf{x}(l_1),\cdots,\mathbf{x}(l_k)\}$, $\mathbf{C}_{1:k}=\{\mathbf{C}_{l_1},\cdots,\mathbf{C}_{l_k}\}$, and $\mathbf{Z}_{1:k}=\{\mathbf{z}_{l_1},\cdots,\mathbf{z}_{l_k}\}$. The estimation covariance of $\widehat{\mathbf{x}}(l_k)$ is bounded below by $\mathbf{J}_k^{-1}$, i.e., $\mathbf{\Sigma}_{\widehat{\mathbf{x}}(l_k)}\geqslant\mathbf{J}_k^{-1}$, where $\mathbf{J}_k$ is the lower-right $[2(N-1)\times 2(N-1)]$ submatrix of the inverse of the Bayesian information matrix $\mathbf{J}(\mathbf{X}_{0:k})$ [27],

$$\mathbf{J}(\mathbf{X}_{0:k})=\begin{bmatrix}\mathrm{E}\left\{-\frac{\partial^2\log p_k}{\partial\mathbf{X}_{0:(k-1)}\partial\mathbf{X}_{0:(k-1)}^T}\right\} & \mathrm{E}\left\{-\frac{\partial^2\log p_k}{\partial\mathbf{X}_{0:(k-1)}\partial\mathbf{x}^T(l_k)}\right\}\\ \mathrm{E}\left\{-\frac{\partial^2\log p_k}{\partial\mathbf{x}(l_k)\partial\mathbf{X}_{0:(k-1)}^T}\right\} & \mathrm{E}\left\{-\frac{\partial^2\log p_k}{\partial\mathbf{x}(l_k)\partial\mathbf{x}^T(l_k)}\right\}\end{bmatrix}, \tag{42}$$

with the probability distribution $p_k=p(\mathbf{Z}_{1:k},\mathbf{C}_{1:k},\mathbf{X}_{0:k})=p(\mathbf{x}(0))\prod_{j=1}^{k}p\left(\mathbf{C}_{l_j},\mathbf{z}_{l_j}|\mathbf{x}(l_j)\right)\prod_{m=1}^{k}p\left(\mathbf{x}(l_m)|\mathbf{x}(l_{m-1})\right)$, and the expectation is taken with respect to the $\mathbf{X}_{0:k}$, $\mathbf{C}_{1:k}$ and $\mathbf{Z}_{1:k}$. It can be shown that [28] the submatrix $\mathbf{J}_k$ can be computed in a recursive way:

$$\mathbf{J}_{k+1}=\mathbf{D}_k^{22}-(\mathbf{D}_k^{12})^T(\mathbf{J}_k+\mathbf{D}_k^{11})^{-1}\mathbf{D}_k^{12}, \tag{43}$$

where

$$\begin{aligned}\mathbf{D}_k^{11}&=\mathrm{E}\left\{-\frac{\partial^2}{\partial\mathbf{x}(l_k)\partial\mathbf{x}^T(l_k)}\log p\left(\mathbf{x}(l_{k+1})|\mathbf{x}(l_k)\right)\right\}\\ \mathbf{D}_k^{12}&=\mathrm{E}\left\{-\frac{\partial^2}{\partial\mathbf{x}(l_k)\partial\mathbf{x}^T(l_{k+1})}\log p\left(\mathbf{x}(l_{k+1})|\mathbf{x}(l_k)\right)\right\}\\ \mathbf{D}_k^{22}&=\mathrm{E}\left\{-\frac{\partial^2}{\partial\mathbf{x}(l_{k+1})\partial\mathbf{x}^T(l_{k+1})}\log p\left(\mathbf{x}(l_{k+1})|\mathbf{x}(l_k)\right)\right\}\\ &\quad+\mathrm{E}\left\{-\frac{\partial^2}{\partial\mathbf{x}(l_{k+1})\partial\mathbf{x}^T(l_{k+1})}\log p\left(\mathbf{C}_{l_{k+1}},\mathbf{z}_{l_{k+1}}|\mathbf{x}(l_{k+1})\right)\right\}.\end{aligned} \tag{44}$$

Based on the dynamic system model (19), the two conditional probability distribution in (44) are:

$$\begin{aligned}&-\log p\left(\mathbf{x}(l_{k+1})|\mathbf{x}(l_k)\right)\\ &=c_1+\frac{1}{2}[\mathbf{x}(l_{k+1})-\mathbf{A}\mathbf{x}(l_k)-\mathbf{b}]^T\mathbf{Q}^{-1}[\mathbf{x}(l_{k+1})-\mathbf{A}\mathbf{x}(l_k)-\mathbf{b}],\end{aligned} \tag{45}$$

$$\begin{aligned}&-\log p\left(\mathbf{C}_{l_{k+1}},\mathbf{z}_{l_{k+1}}|\mathbf{x}(l_{k+1})\right)\\ &=c_2+\frac{1}{2}\left[\mathbf{z}_{l_{k+1}}-\mathbf{C}_{l_{k+1}}\mathbf{x}(l_{k+1})\right]^T\mathbf{R}^{-1}\left[\mathbf{z}_{l_{k+1}}-\mathbf{C}_{l_{k+1}}\mathbf{x}(l_{k+1})\right],\end{aligned} \tag{46}$$

where c_1 and c_2 are constants. Substituting (45) and (46) into (44), we obtain $\mathbf{D}_k^{11}=\mathbf{A}^T\mathbf{Q}^{-1}\mathbf{A}$, $\mathbf{D}_k^{12}=-\mathbf{A}^T\mathbf{Q}^{-1}$, and $\mathbf{D}_k^{22}=\mathbf{Q}^{-1}+\mathbf{C}_{l_{k+1}}^T\mathbf{R}^{-1}\mathbf{C}_{l_{k+1}}$, and then the recursive formula (43) can be rewritten as:

$$\mathbf{J}_{k+1}=\mathbf{Q}^{-1}+\mathbf{C}_{l_{k+1}}^T\mathbf{R}^{-1}\mathbf{C}_{l_{k+1}}-\mathbf{Q}^{-T}\mathbf{A}(\mathbf{J}_k+\mathbf{A}^T\mathbf{Q}^{-1}\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Q}^{-1}. \tag{47}$$

After applying the matrix inversion lemma [32], it can be further simplified as:

$$\mathbf{J}_{k+1}=(\mathbf{Q}+\mathbf{A}\mathbf{J}_k^{-1}\mathbf{A}^T)^{-1}+\mathbf{C}_{l_{k+1}}^T\mathbf{R}^{-1}\mathbf{C}_{l_{k+1}}. \tag{48}$$

If we take the DKF as the initialization method, the initial information submatrix $\mathbf{J}_0$ can be set as: $\mathbf{J}_0=\delta\mathbf{I}$. On the other hand, if the CD-BS method is taken, $\mathbf{J}_0$ can be set as: $\mathbf{J}_0=[\mathrm{CRLB}(\mathbf{x})]^{-1}$, where $\mathrm{CRLB}(\mathbf{x})$ is the CRLB for the initial values $\mathbf{x}(0|0)$. Since $\hat{\mathbf{x}}_i(0|0)=\left[\hat{\beta}_i\ \ \hat{\theta}_i^0+(1-1/\hat{\beta}_i)[c_i(t_e)-c_i(t_s)]\right]^T$, we can define that

$\mathbf{x}(0|0) \triangleq \mathbf{g}([\boldsymbol{\beta}^T \ \boldsymbol{\theta}^T]^T)$, where $\boldsymbol{\beta} = [\beta_2, \cdots, \beta_N]^T$ and $\boldsymbol{\theta} = [\theta_2^0, \cdots, \theta_N^0]^T$. Thus the CRLB for $\mathbf{x}(0|0)$ can be calculated as:

$$\text{CRLB}(\mathbf{x}) = \mathbf{\Pi}\ \text{CRLB}\left([\boldsymbol{\beta}^T \ \boldsymbol{\theta}^T]^T\right)\mathbf{\Pi}^T, \qquad (49$$

where $\mathbf{\Pi} = [\frac{\partial \mathbf{g}}{\partial \boldsymbol{\beta}} \ \frac{\partial \mathbf{g}}{\partial \boldsymbol{\theta}}] = \left[\boldsymbol{\delta}_2 \oplus \cdots \oplus \boldsymbol{\delta}_N \ \ \mathbf{I}_{N-1} \otimes ([0\ 1]^T)\right]$ with $\boldsymbol{\delta}_i = \left[1 \ \ [c_i(t_e) - c_i(t_s)]/\beta_i^2\right]^T$ and $\oplus$ denotes the direct sum. The centralized CRLB for $[\boldsymbol{\beta}^T \ \boldsymbol{\theta}^T]^T$ was derived in Appendix B.

VI. Simulation Results and Discussions

In this section, numerical simulations will be presented to assess the performance of proposed clock parameters initialization and tracking algorithm in Wireless Sensor Networks. The measure of parameter estimate fidelity at time l is Root Average Mean Squared Error (RAMSE) of clock skew and accumulated offset over the whole network:

$$\text{RAMSE}(\zeta(l)) = \sqrt{\frac{1}{N-1}\sum_{i=2}^{N}(\hat{\zeta}_i(l) - \zeta_i(l))^2},$$

where $\zeta \in \{\beta, \vartheta\}$. Each sensor node is equipped with an oscillator having RMS period jitter 3ps and $f_0 = 150$MHz [33], with $\tau_0 = 0.1$s is assumed. Network of 25 nodes are randomly deployed in an area 5×5 with communication radius 1.5. 1000 independent networks are generated for averaging the RAMSE in the figures. In the simulations, initial clock skew, initial clock offsets and fixed delays are uniformly selected from ranges [0.9, 1.1], $[-5\tau_0, 5\tau_0]$ and $[0.01\tau_0, 0.02\tau_0]$, respectively. The variance of random delay is $0.5\tau_0$. For initialization, 5 rounds of two-way time-stamp exchange are performed. For all algorithms in the simulations, it is assumed that one iteration of distributed processing (including message exchanges and local computations) can be completed within τ_0.

A. Clock Parameters Initialization

The performance of the proposed CD-BS initialization algorithm, and DKF initialization (set $\Delta = 1$, $\hat{\mathbf{x}}_i(0|0) = [1\ 0]^T$ and $\mathbf{P}(0|0) = \delta^{-1}\mathbf{I}$ with $\delta = 0.01$) and consensus algorithm [13] are first compared. For consensus algorithm, it seeks to converge to the average value of all the nodes' clock parameters θ_i^0 and β_i. Therefore, the estimated clock parameters from consensus algorithm are transformed by $\hat{\mathbf{x}}_i(l) = \left[\hat{\beta}_i \ \ \hat{\theta}_i^0 + (1 - 1/\hat{\beta}_i)(l\tau_0)\right]^T$. Furthermore, the RAMSE for consensus algorithm is defined as

$$\text{RAMSE}(\zeta(l))_{con} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\hat{\zeta}_i(l) - \frac{1}{N}\sum_{i=1}^{N}\zeta_i(l)\right)^2}.$$

Finally, the CRLB$(\mathbf{x})$ in (49) is also plotted as performance limit.

The RAMSEs of $\hat{\vartheta}_i$ and $\hat{\beta}_i$ averaged over all nodes and all network topologies are shown in Figure 3. It can be seen that for the CD-based algorithm, as the number of iteration increases, RAMSE gradually decreases and finally approaches the batch mode CRLB, while there is a constant gap between

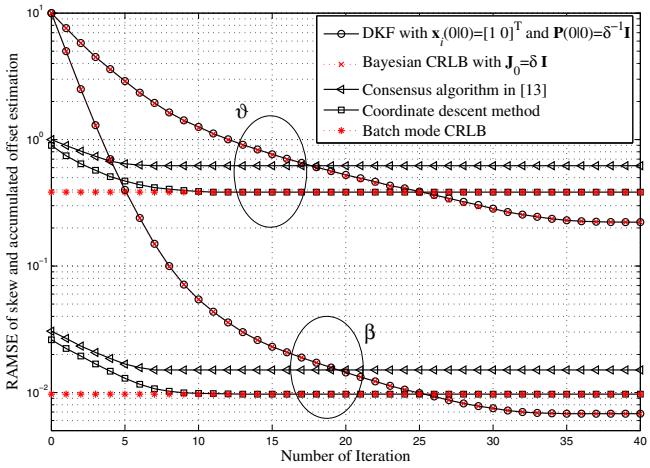


Fig. 3: RAMSE of accumulated clock offsets ϑ and skews β estimation.

the performance of the consensus algorithm and the CRLB even after consensus algorithm converged. On the other hand, DKF initialization has a slower convergence than CD-based method. It is noticed that for CD-based method, 13 iterations (in addition to 5 rounds of time-stamp exchange at the beginning) are performed till convergence while approximately 35 iterations (one iteration includes two-way time-stamp exchange and current estimate dissemination) are required for DKF method. However, DKF method, with performance coinciding with the Bayesian CRLB, has a smaller RAMSE than CD-based method after convergence. This shows that CD-based method is suitable for rapid initialization but not for long-term synchronization. In terms of complexity, the corresponding parameters in Table I are $L_1 = 5, N_{CD} = 13, N_{KF} = 35$. If we set $\lambda_i = 5$ and $B = 25$, the total cost for DKF at node i is $\mathbb{O}(290535)$, which is at least 17 times of 17275, the cost for CD-BS method.

On the other hand, the performance of covariance matrix estimation by CD-BS is measured by the Frobenius norm of the difference between estimated covariance matrix $\mathbf{P}_{\mathcal{N}_{[i]}}(0|0)$ and the centralized true value (the entries of CRLB$(\mathbf{x})$ in (49) corresponding to $\mathcal{N}_{[i]}$). From Figure 4, we can notice that the Frobenius norm errors become smaller as the number of iteration increases and finally converge to stable values. Furthermore, the larger the number of bootstrap samples, the smaller the Frobenius norm of error after convergence.

B. Distributed Clock Parameters Tracking

For assessing the performance of distributed tracking algorithm, after CD-BS initialization, $\Delta = 2000$ is set, i.e., re-synchronization using a single Kalman filter update every $\Delta \times \tau_0$. The idle periods between Kalman filter updates allows the sensor network to perform operations other than synchronization. Figures 5 and 6 show the performance of the tracked accumulated offsets and skews versus the number of Kalman filter iterations (notice that one iteration represents $\Delta \times \tau_0$), respectively. Both the prediction and posterior RAMSEs are shown, illustrating both the error due to pure prediction step and improvement due to observation updates. Firstly, it is noticed that as the number of iterations increases, the posterior RAMSE decreases, and finally touching the Bayesian CRLB.

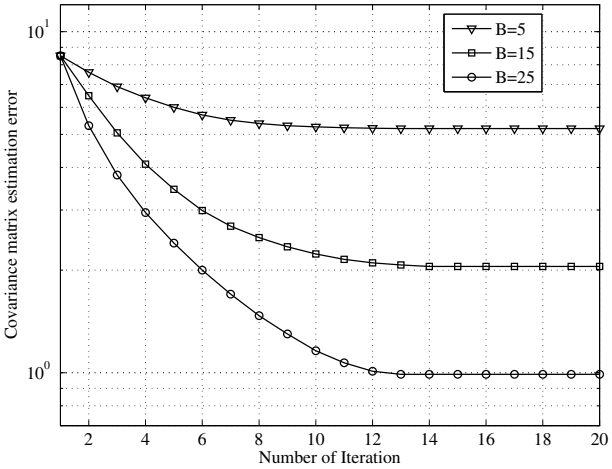


Fig. 4: Frobenius norm of error in estimated covariance matrix $\mathbf{P}_{N[i]}(0|0)$.

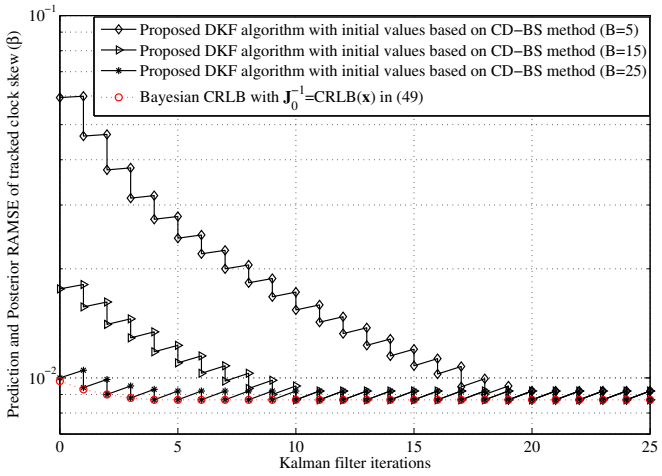


Fig. 6: RAMSE of tracked clock skews.

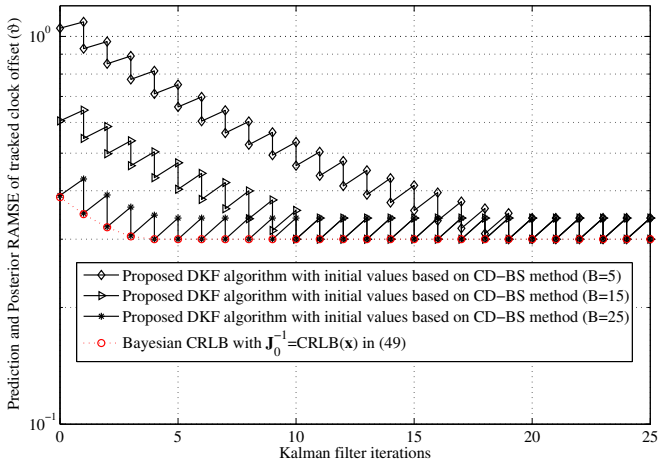


Fig. 5: RAMSE of tracked accumulated offsets.

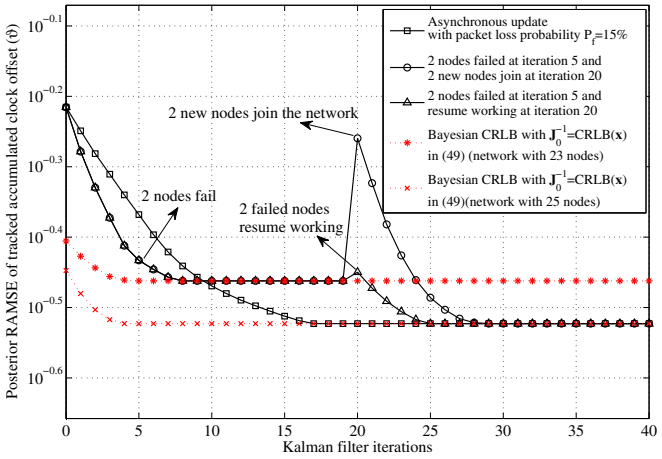


Fig. 7: Performance of accumulated offsets tracking in asynchronous scheduling, and in the presence of nodes failure and newly joined nodes.

But there is a significant difference in the convergence speed for different bootstrap samples used in the initialization. For $B = 25$, the posterior RAMSE basically touches the Bayesian CRLB in the first round of re-synchronization. This is not the case for smaller B. This is because in the first round of Kalman filter update, the estimated $\mathbf{P}(0|0)$ will be used as weighting for combining the prior estimate (which is an estimate touching the CRLB as shown in Figure 3) with that due to the new observations. If $\mathbf{P}(0|0)$ is not accurately estimated, it would degrade the RAMSE of the first re-synchronized clock parameters estimate. Only more observations are obtained, the effect of $\mathbf{P}(0|0)$ estimate becomes insignificant and then the RAMSE approaches the Bayesian CRLB. Furthermore, we can notice that the RAMSE can be maintained within a limited range from Bayesian CRLB after convergence. This is an important feature in the proposed method, as there is a guarantee in the RAMSE being kept close to Bayesian CRLB.

Finally, we conducted simulations to verify the algorithm is robust to nodes failure and new neighbors, and can also works in asynchronous scheduling. In the simulations, the network starts with 25 nodes at the beginning of tracking. Two nodes are chosen at random to fail at iteration 5, and then the failed nodes resume working or two newly joined nodes are added at iteration 20. Figures 7 and 8 show the posterior RAMSE (RAMSE after observation updates) of accumulated clock offset and skew respectively, versus the number of Kalman filter iterations (with $\Delta = 2000$, and $B = 15$ in CD-BS initialization). It can be seen that the Bayesian CRLB for network with 25 nodes would be lower than that of 23 nodes on average, since more timing information is present in the network with 25 nodes. Furthermore, we notice that with 2 nodes failure at iteration 5, the proposed DKF converges to the Bayesian CRLB for network with 23 nodes. If the two failed nodes resume working at iteration 20, the RAMSE can further decrease to approach the Bayesian CRLB of network with 25 nodes. On the other hand, if 2 new nodes join the network at iteration 20, the RAMSE initially shows a sharp increase since it is assumed that the newly joined nodes do not have any prior information on their clock parameters. But the RAMSE decreases quickly and finally touches the Bayesian CRLB of network with 25 nodes. Figures 7 and 8 also show the asynchronous implementation of DKF by imposing a 15% probability that any any local state exchange packet will be lost in the data transmission. From the figures, it is observed that the proposed DKF also works in asynchronous scheduling, and it can converge to the Bayesian CRLB with only a slightly slower convergence speed than synchronous scheduling.

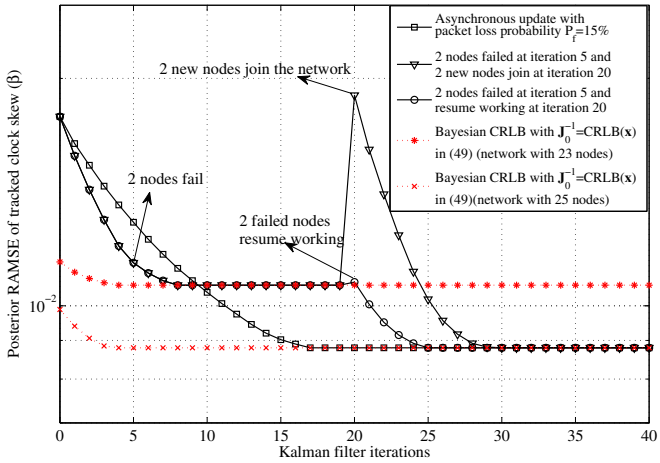


Fig. 8: Performance of skew tracking in asynchronous scheduling, and in the presence of nodes failure and newly joined nodes.

VII. Conclusions

In this paper, a fully-distributed Kalman filter for tracking the time-varying clock parameters in wireless sensor network was proposed. The proposed algorithm only requires communications between neighboring nodes and is scalable with network size. Furthermore, it can perform well in dynamic networks where there is node failure or new nodes joining in. A low-complexity Coordinate-Descent with bootstrap method was also proposed for rapid initialization for the tracking algorithm. Simulation results show that the proposed initialization method achieves higher accuracy than average consensus approach, and the proposed distributed Kalman filter maintains long-term clock parameters accuracy, and is robust to network topology changes during tracking process.

Appendix A
Calculation of the Oscillator Parameter

This Appendix derives the relationship between oscillator quality parameter p and the oscillator period jitter or phase jitter commonly available in data sheet.

A. Period Jitter

The relationship between the phase noise and RMS period jitter [31] can be expressed as:

$$J_{PER}=\sqrt{\frac{8T_0^2}{4\pi^2}\int_0^{\infty}10^{\frac{\mathcal{L}(f)}{10}}\left[\sin^2\left(\pi f T_0\right)\right]df}, \tag{50}$$

where f is the frequency offset with respect to the oscillator frequency f_0, $\mathcal{L}(f)$ is the phase noise power spectral density used to describe oscillator performance, and $T_0=1/f_0$. Since the relationship between $\mathcal{L}(f)$ and parameter p for free-running oscillator is given [30] by

$$\mathcal{L}(f)=10\log_{10}\left(pf_0^2/f^2\right), \tag{51}$$

the RMS period jitter can be derived as:

$$J_{PER}=\sqrt{\frac{8T_0^2}{4\pi^2}\int_0^{\infty}\frac{pf_0^2}{f^2}\left[\sin^2(\pi f T_0)\right]df}=\sqrt{pT_0}, \tag{52}$$

and the oscillator quality parameter p can be calculated as $p=J_{PER}^2 f_0$.

B. Phase Jitter

The relationship between the phase noise and RMS phase jitter [31] can be expressed as:

$$J_{PHA}=\frac{1}{2\pi f_0}\sqrt{2\int_{f_1}^{f_2}10^{\frac{\mathcal{L}(f)}{10}}df}, \tag{53}$$

where f_1 and f_2 are the lower and upper frequency offsets with respect to RMS phase jitter. Substituting the power spectral density (51) into (53), RMS phase jitter can be derived as:

$$J_{PHA}=\frac{1}{2\pi f_0}\sqrt{2\int_{f_1}^{f_2}\frac{pf_0^2}{f^2}df}=\frac{1}{2\pi}\sqrt{2p\left(\frac{1}{f_1}-\frac{1}{f_2}\right)}, \tag{54}$$

and then the oscillator quality parameter p can be calculated as $p=2\pi^2 J_{PHA}^2\cdot f_1 f_2/(f_2-f_1)$.

Appendix B
Centralized Cramer Rao Lower Bound for Initial Clock Offset and Skew Estimation

Based on (12), (13) and using (34), the centralized log-likelihood function for θ_i^0, β_i and d_{ij} is written in (55), shown at the bottom of the page, where N_t is the total number of rounds of time-stamp exchange in the entire network, and it is assumed that $d_{ij}=d_{ji}$.

Define

$$\varpi_{ij}=\begin{cases}1 & \text{if } j\in\mathcal{N}_i\\ 0 & \text{otherwise}\end{cases},$$

the Fisher Information Matrix (FIM) for $\boldsymbol{\beta}=[\beta_2,\cdots,\beta_N]^T$, $\boldsymbol{\theta}=[\theta_2^0,\cdots,\theta_N^0]^T$, and $\mathbf{d}$ ($\mathbf{d}$ is a vector containing d_{ij} as elements where $j\in\mathcal{N}_i$ and the indexes are arranged in ascending order on i and then on j) is given by [17]

$$\mathbf{F}=-\mathrm{E}\begin{pmatrix}\frac{\partial^2\ln f}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T} & \frac{\partial^2\ln f}{\partial\boldsymbol{\beta}\partial\boldsymbol{\theta}^T} & \frac{\partial^2\ln f}{\partial\boldsymbol{\beta}\partial\mathbf{d}^T}\\ \frac{\partial^2\ln f}{\partial\boldsymbol{\theta}\partial\boldsymbol{\beta}^T} & \frac{\partial^2\ln f}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T} & \frac{\partial^2\ln f}{\partial\boldsymbol{\theta}\partial\mathbf{d}^T}\\ \frac{\partial^2\ln f}{\partial\mathbf{d}\partial\boldsymbol{\beta}^T} & \frac{\partial^2\ln f}{\partial\mathbf{d}\partial\boldsymbol{\theta}^T} & \frac{\partial^2\ln f}{\partial\mathbf{d}\partial\mathbf{d}^T}\end{pmatrix}, \tag{56}$$

$$\begin{aligned}\ln f\left(\left\{T_{1,l}^{\{i,j\}},T_{2,l}^{\{i,j\}},T_{3,l}^{\{i,j\}},T_{4,l}^{\{i,j\}}\right\}_{l=1}^{L_1}|\theta_i^0,\beta_i,d_{ij}\right)&=\frac{N_t}{2}\cdot\ln\frac{1}{2\pi\sigma^2}-\frac{1}{2\sigma^2}\\ \times\sum_{l=1}^{L_1}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}&\left\{\left(1/\beta_j\cdot\left[T_{2,l}^{\{i,j\}}-\theta_j^0\right]-1/\beta_i\cdot\left[T_{1,l}^{\{i,j\}}-\theta_i^0\right]-d_{ij}\right)^2\right.\\ &\left.+\left(1/\beta_i\cdot\left[T_{4,l}^{\{i,j\}}-\theta_i^0\right]-1/\beta_j\cdot\left[T_{3,l}^{\{i,j\}}-\theta_j^0\right]-d_{ij}\right)^2\right\}\end{aligned} \tag{55}$$

where the expressions of different elements are shown at the bottom of the page. The centralized CRLB for $[\boldsymbol{\beta}^T\ \boldsymbol{\theta}^T]^T$ can be obtain as the upper-left $[2(N-1)\times 2(N-1)]$ submatrix of $\mathbf{F}^{-1}$.

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, Mar. 2002.

[2] N. Bulusu and S. Jha, *Wireless Sensor Networks: A Systems Perspective*. Artech House, 2005.

[3] S. Ganeriwal, R. Kumar, and M. B. Srivastava, "Time-sync protocol for sensor networks," Technical Reports, Center for Embedded Networking Sensing, UC Los Angeles, 2003.

[4] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," in *Proc. 2002 Symposium on Operating System Design and Implementation*, pp. 147–163.

[5] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The flooding time synchronization protocol," in *Proc. 2004 International Conference on Embedded Networked Sensor Systems*, pp. 39–49.

[6] J. V. Greunen and J. Rabaey, "Lightweight time synchronization for sensor network," in *Proc. 2003 International Workshop on Wireless Sensor Networks and Applications*.

[7] S. Yoon, C. Veerarittiphan, and M. L. Sichitiu, "Tiny-sync: tight time synchronization for wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 3, issue 2, no. 8, June 2007.

[8] K.-L. Noh, E. Serpedin, and K. Qaraqe, "A new approach for time synchronization in wireless sensor networks: pairwise broadcast synchronization," *IEEE Trans. Wireless Commun.*, vol. 7, no. 9, pp. 3318–3322, Dec. 2008.

[9] I. K. Rhee, Jaehan Lee, E. Serpedin, and Y. C. Wu, "Clock synchronization in wireless sensor networks: an overview," *Sensors* vol. 9, no. 1, pp. 56–85, 2009.

[10] H. Dai and R. Han, "TSync: a lightweight bidirectional time synchronization service for wireless sensor networks," *ACM Mobile Comput. Commun. Review*, vol. 8, no. 1, pp. 125–139, 2004.

[11] Q. Li and D. Rus, "Global clock synchronization in sensor networks," *IEEE Trans. Comput.*, vol. 55, no. 2, pp. 214–225, Feb. 2006.

[12] A. Giridhar and P. R. Kumar, "Distributed clock synchronization over wireless sensor networks: algorithms and analysis," in *Proc. 2006 IEEE Conference on Decision and Control*, pp. 4915–4920.

[13] L. Schenato and G. Gamba, "A distributed consensus protocol for clock synchronization in wireless sensor network," in *Proc. 2007 IEEE Conference on Decision and Control*, pp. 2289–2294.

[14] Y. W. Hong and A. Scaglione, "A scalable synchronization protocol for large scale sensor networks and its applications," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 5, pp. 1085–1099, May 2005.

[15] A. S. Hu and S. D. Servetto, "On the scalability of cooperative time synchronization in pulse-connected network," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2725–2748, 2006.

[16] O. Simeone and U. Spagnolini, "Distributed synchronization in wireless networks," *IEEE Signal Process. Mag.*, vol. 25, no. 5, pp. 81–97. Sept. 2008.

[17] S. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Prentice-Hall, 1993.

$$
-\mathrm{E}\left\{\frac{\partial^2 \ln f}{\partial\beta_i\partial\beta_j}\right\}=\begin{cases}
1/(\sigma^2\beta_i^4)\sum_{l=1}^{L_1}\sum_{k\in\mathcal{N}_i}\left\{\left[T_{1,l}^{\{i,k\}}-\theta_i^0\right]^2+\left[T_{3,l}^{\{k,i\}}-\theta_i^0\right]^2+2\beta_i^2\sigma^2\right.\\
\left.+\left(\frac{\beta_i}{\beta_k}\left[T_{3,l}^{\{i,k\}}-\theta_k^0\right]+\beta_i d_{ik}\right)^2+\left(\frac{\beta_i}{\beta_k}\left[T_{1,l}^{\{k,i\}}-\theta_k^0\right]+\beta_i d_{ik}\right)^2\right\}, & \text{if } i=j\\
\varpi_{ij}/(\sigma^2\beta_i^2\beta_j^2)\sum_{l=1}^{L_1}\left\{\left[\theta_i^0-T_{1,l}^{\{i,j\}}\right]\left(\beta_j/\beta_i\left[T_{1,l}^{\{i,j\}}-\theta_i^0\right]+\beta_j d_{ij}\right)\right.\\
+\left[\theta_j^0-T_{3,l}^{\{i,j\}}\right]\left(\beta_i/\beta_j\left[T_{3,l}^{\{i,j\}}-\theta_j^0\right]+\beta_i d_{ij}\right)\\
+\left[\theta_j^0-T_{1,l}^{\{j,i\}}\right]\left(\beta_i/\beta_j\left[T_{1,l}^{\{j,i\}}-\theta_j^0\right]+\beta_i d_{ij}\right)\\
\left.+\left[\theta_i^0-T_{3,l}^{\{j,i\}}\right]\left(\beta_j/\beta_i\left[T_{3,l}^{\{j,i\}}-\theta_i^0\right]+\beta_j d_{ij}\right)\right\}, & \text{otherwise}
\end{cases}
$$

$$
-\mathrm{E}\left\{\frac{\partial^2 \ln f}{\partial\theta_i^0\partial\theta_j^0}\right\}=\begin{cases}8L_1\lambda_i/(2\sigma^2\beta_i^2), & \text{if } i=j\\ -8(L_1\varpi_{ij})/(2\sigma^2\beta_i\beta_j), & \text{otherwise}\end{cases}
$$

$$
-\mathrm{E}\left\{\frac{\partial^2 \ln f}{\partial d_{ij}\partial d_{kl}}\right\}=\begin{cases}8L_1/(2\sigma^2), & \text{if } i=k, j=l\\ 0, & \text{otherwise}\end{cases}
$$

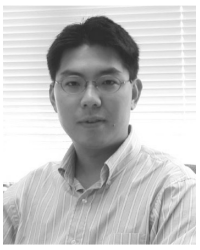
$$
-\mathrm{E}\left\{\frac{\partial^2 \ln f}{\partial\theta_i^0\partial\beta_j}\right\}=\begin{cases}
1/(\sigma^2\beta_i^3)\sum_{l=1}^{L_1}\sum_{k\in\mathcal{N}_i}\left\{\left[T_{1,l}^{\{i,k\}}-\theta_i^0\right]+\beta_i/\beta_k\left[T_{3,l}^{\{i,k\}}-\theta_k^0\right]\right.\\
\left.+\beta_i/\beta_k\left[T_{1,l}^{\{k,i\}}-\theta_k^0\right]+\left[T_{3,l}^{\{k,i\}}-\theta_i^0\right]+2\beta_i d_{ik}\right\}, & \text{if } i=j\\
(-1)/(\sigma^2\beta_i\beta_j^2)\sum_{l=1}^{L_1}\left\{\frac{\beta_j}{\beta_i}\left[T_{1,l}^{\{i,j\}}+T_{3,l}^{\{j,i\}}-2\theta_i^0\right]+2\beta_j d_{ij}\right.\\
\left.+\left[T_{1,l}^{\{j,i\}}+T_{3,l}^{\{j,i\}}-2\theta_j^0\right]\right\}, & \text{otherwise}
\end{cases}
$$

$$
-\mathrm{E}\left\{\frac{\partial^2 \ln f}{\partial\theta_i^0\partial d_{ij}}\right\}=0,
$$

$$
-\mathrm{E}\left\{\frac{\partial^2 \ln f}{\partial\beta_k\partial d_{ij}}\right\}=\begin{cases}
\frac{1}{\sigma^2\beta_i^2}\sum_{l=1}^{L_1}\left\{2\theta_i^0-T_{1,l}^{\{i,j\}}-T_{3,l}^{\{j,i\}}+2\beta_i d_{ij}+\frac{\beta_i}{\beta_j}\left[T_{1,l}^{\{j,i\}}+T_{3,l}^{\{i,j\}}-2\theta_j^0\right]\right\}, & \text{if } k=i\\
\frac{1}{\sigma^2\beta_j^2}\sum_{l=1}^{L_1}\left\{2\theta_j^0-T_{1,l}^{\{j,i\}}-T_{3,l}^{\{i,j\}}+2\beta_j d_{ij}+\frac{\beta_j}{\beta_i}\left[T_{1,l}^{\{i,j\}}+T_{3,l}^{\{j,i\}}-2\theta_i^0\right]\right\}, & \text{if } k=j\\
0 & \text{otherwise.}
\end{cases}
$$

[18] Z. Q. Luo and P. Tseng, "On the convergence of coordinate descent method for convex differentiable minimization," *J. Optim. Theory Appl.*, vol. 72, pp. 7–35, 1992.
[19] C. J. Hsieh, K. W. Chang, C. J. Lin, S. S. Keerthi, and S. Sundararajan, "A dual coordinate descent method for large-scale linear SVM," in *Proc. 2008 International Conference on Machine Learning*.
[20] A. M. Zoubir and D. R. Iskander, *Bootstrap Techniques for Signal Processing*. Cambridge University Press, 2004.
[21] N. Freris, V. Borkar, and P. Kumar, "A model-based approach to clock synchronization," in *Proc. 2009 IEEE Conference on Decision and Control, held jointly with the 2009 Chinese Control Conference (CDC/CCC)*, pp. 5744–5749.
[22] H. Kim, X. Ma, and B. E. Hamilton, "Tracking low-precision clocks with time-varying drifts using Kalman filtering," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 257–270, Feb. 2012.
[23] M. Leng and Y. C. Wu, "Distributed clock synchronization for wireless sensor networks using belief propagation," *IEEE Trans. Signal Process.*, vol. 59, no. 11, pp. 5404–5414, Nov. 2011.
[24] D. Zennaro, A. Ahmad, L. Vangelista, E. Serpedin, H. Nounou, and M. Nounou, "Network-wide clock synchronization via message passing with exponentially distributed link delays," *IEEE Trans. Coummun.*, vol. 61, no. 5, pp. 2012–2024, May 2013.
[25] A. Ahmad, D. Zennaro, E. Serpedin, and L. Vangelista, "Time-varying clock offset estimation in two-way timing message exchange in wireless sensor networks using factor graphs," in *Proc. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing*.
[26] A. Ahmad, D. Zennaro, E. Serpedin, and L. Vangelista, "A factor graph approach to clock offset estimation in wireless sensor networks," *IEEE Trans. Inf. Theory*, vol. 58, no. 7, pp. 4244–4260, July 2012.
[27] H. L. Van Trees and K. L. Bell, *Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking*. Wiley, 2007.
[28] P. Tichavsky, C. H. Muravchik, and A. Nehorai, "Posterior Cramer-Rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Process.*, vol. 46, no. 5, pp. 1386–1396, May 1998.
[29] Mike Brooks, *The Matrix Reference Manual*. Imperial College, 2005.
[30] A. Demir, A. Mehrotra, and J. Roychowdhury, "Phase noise in oscillators: a unifying theory and numerical methods for characterisation," *IEEE Trans. Circuits Syst. I*, vol. 47, no. 5, pp. 655–674, May 2000.
[31] B. Drakhlis, "Calculating oscillator jitter by using phase-noise analysis," *Microwaves & RF*, pp. 82–90, Jan. 2001.
[32] C. Liang, W. Lenan, and R. Piche, "Posterior Cramer-Rao lower bound for mobile tracking in mixed LOS/NLOS conditions," in *Proc. 2009 EUSIPCO*.
[33] Epson oscillator XG1000-CA/CB datasheet. Available: http://www.eea.epson.com/portal/pls/portal/docs/1/1539450.PDF.

Bin Luo received the B.Eng. degree from the Civil Aviation University of China, Tianjin, China, in 2006 and the M.Eng. degree from Institute of Electronics, Chinese Academy of Sciences (IECAS), Beijing, China, in 2009, both in electrical engineering. He is currently working towards the Ph.D. degree with the department of Electrical and Electronic Engineering, The University of Hong Kong, Hong Kong. His current research interests include distributed signal processing, distributed optimization, and machine learning, with applications to wireless sensor networks and wireless communication systems.



Yik-Chung Wu received the B.Eng. (EEE) degree in 1998 and the M.Phil. degree in 2001 from the University of Hong Kong (HKU). He received the Croucher Foundation scholarship in 2002 to study Ph.D. degree at Texas A&M University, College Station, and graduated in 2005. From August 2005 to August 2006, he was with the Thomson Corporate Research, Princeton, NJ, as a Member of Technical Staff. Since September 2006, he has been with HKU, currently as an Associate Professor. He was a visiting scholar at Princeton University, in summer 2011. His research interests are in general area of signal processing and communication systems, and in particular distributed signal processing and communications; optimization theories for communication systems; estimation and detection theories in transceiver designs; and smart grid. Dr. Wu served as an Editor for IEEE COMMUNICATIONS LETTERS, is currently an Editor for IEEE TRANSACTIONS ON COMMUNICATIONS and the *Journal of Communications and Networks*.