2016 14th International Conference on Control, Automation, Robotics & Vision
Phuket, Thailand, 13-15th November 2016 (ICARCV 2016)

A New Distributed Algorithm of Clock Synchronization for Static Sensor Networks

Yong Qiao
School of Control Science and Engineering
Zhejiang University
Hangzhou 310027, P. R. China.
Email: yongqiao@zju.edu.cn

Wenlun Yang
School of Control Science and Engineering
Zhejiang University
Hangzhou 310027, P. R. China.
Email: yangwenlun@zju.edu.cn

Minyue Fu
School of Electrical Engineering and Computer Science
University of Newcastle
NSW 2308 Australia.
minyue.fu@newcastle.edu.au

***Abstract*—In this paper, a new distributed algorithm is presented to achieve the consensus of time variations and that of initial time simultaneously. By combining both controller and estimator design methods, it obtains higher synchronization precision with stronger robustness against noisy inputs resulted from crystal oscillators. Furthermore, the control input is ensured bounded to make our algorithm realizable in practical implementation. The implementation of the algorithm allows the receiving end to be event-triggered, while the transmitting end is executed periodically. The performance of the algorithm is illustrated by the given numerical simulations.**

I. Introduction

Recent years have witnessed technological advances in the development of low-cost programmable sensors that are capable of sensing, data processing and storing, and communication via wireless channels. Wireless sensor networks (WSNs) can be applied in a wide range of cases, such as climate monitoring, target positioning and tracking, and event detection [2]. However, clock drifting due to low-cost oscillators in sensor nodes has posed a serious problem in time critical applications. How to achieve good clock synchronization among sensors with low overhead in communication and computation is an important and interesting research problem.

There are two kinds of clock synchronization approaches [1]: master-to-slave synchronization and peer-to-peer synchronization. A master-to-slave method assigns one node as the master and other nodes as slaves. The Reference Broadcast Sync (RBS) method reduces the non-deterministic latency by using partial aggregation synchronization and preserves energy via the post-facto scheme [3]. The network is divided into multiple clusters and a master node is required for each cluster. The Time-Diffusion Protocol (TDP) [4], using an iterative, weighted averaging technique, achieves a network-wide equilibrium time. It is based on a message diffusion involving all nodes in the synchronization process. Nodes periodically self-determine to become a master/diffused-leader, using an election/re-election procedure to form a radial tree structure. The Timing-Sync Protocol for Sensor Networks (TPSN) [5] divides the clock synchronization into two phases: the building of a spanning tree and the synchronization phase. Pair-wise synchronization is performed along tree branches by using sender-receiver handshake exchanges. Node failure and packet losses which can significantly influence the performance of wireless networks, are common problems with which these methods are confronted. The Flooding Time Sync Protocol (FTSP) [6] can achieve robustness to root and link failure, and dynamic topology variations to some extent.

In contrast, a peer-to-peer protocol requires no center nodes, i.e., it is a distributed method. Average Time Sync (ATS) [7] uses a cascade of two consensus algorithms to make all nodes converge to a virtual reference clock by tuning compensation parameters. It obtains the consensus of clock rates and that of offsets separately. In Distributed Time Sync (DTS) [8], neighboring nodes exchange packets containing time-stamps on a one-to-one basis. Any participating node can become a reference node by not adapting its own clock during the synchronization procedure, which converts the problem into a decentralized optimization problem.

Synchronization methods can also be roughly divided into internal synchronization and external synchronization. Since the clock physical time which depends on the oscillator frequency cannot be easily adjusted, the external synchronization is more preferred to be achieved. However, in most researches mentioned above, this pattern is not explained explicitly in the modeling part. External clocks, also called virtual clocks, should be modeled to achieve the synchronization, as the algorithm named FLOPSYNC-2 [9] does. It presents a specific relationship between the actual and virtual clock. Our paper focuses on the external synchronization for the reason that in most application environments of WSNs, local clocks of sensors cannot be adjusted physically.

In this paper, a new distributed algorithm of clock synchronization is to be developed with a peer-to-peer approach. This is motivated by the fact that most existing peer-to-peer algorithms achieve the consensus of virtual clock rates and that of initial offsets separately to complete the clock synchronization. This separation makes the problem conceptually easier but convergence slower, resulting in high communication and computation overheads. We will combine these two tasks and solve them jointly. Existing synchronization algorithms use either a controller-design method as in [9] or a estimator-based one as in [7]. We combine these two design methods to gain better robustness against noisy measurements and clock rate

978-1-5090-3549-6/16/$31.00

drifts. In our approach, the record of receiving sampling time is triggered by receiving synchronization messages from other nodes while the transmitting occurs periodically according to the local clock of each node. Furthermore, control inputs used to adjust the virtual time of local clocks are proven to be bounded. Following implementation ideas in [7], the issue with asynchronous communication is addressed for our algorithm, which also enhances the robustness to node failure and packet losses.

II. Notation and graph theory

An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is composed of a node set $\mathcal{V} = \{1, 2, \cdots, N\}$ and an edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ where an edge of $\mathcal{G}$ is a pair of un-ordered nodes. $\mathcal{N}_i$ denotes the in-neighbor set of node i, i.e., $\mathcal{N}_i = \{j : (j, i) \in \mathcal{E}\}$. An undirected graph is called connected if any two nodes in $\mathcal{V}$ can be connected by a path of edges in $\mathcal{E}$. Each edge (j, i) is associated with a weight $\xi_{ij} = \xi_{ji} \neq 0$. Then we can introduce a Laplacian matrix L whose ijth entry is given as

$$L(i,j) = \begin{cases} -\xi_{ij} & i \neq j, j \in \mathcal{N}_i \\ 0 & i \neq j, j \notin \mathcal{N}_i \\ \sum\limits_{j \in \mathcal{N}_i} \xi_{ij} & i = j \end{cases}$$

It is well known that 0 is an eigenvalue of L with $\mathbf{1}$ as a corresponding right eigenvector, and all nonzero eigenvalues have positive real parts. For an undirected graph, zero is a simple eigenvalue if and only if the graph is connected.

If relationships between neighboring nodes only exist at each node's event time, such a graph is called as an *extended neighbour graph*, denoted as $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$. $\mathcal{D}(t)$ is a directed graph with at most one arc between each ordered pair of nodes and with exactly one self-loop at each vertex at time t. If $\mathcal{G} = \bigcup_{t_0}^{t_0+kT} \{\mathcal{D}(t)\}$ where k is a positive integer and $t \in [t_0, t_0 + kT]$, then j belongs to $\mathcal{N}_i$ in $\mathcal{G}$ as long as j is the in-neighbour of i in $\mathcal{D}(t)$ at least once in that time interval.

III. Clock Models and Proposed Synchronization Algorithm

In this section, a new algorithm which can achieve the synchronization of time variations and that of initial time simultaneously is proposed. Before presenting our models, certain following symbols have to be introduced. Let $\{\bar{t}_{ik}^j\}_{k=0}^{+\infty}$ stand for the set of sampling instants as well as that of update instants of local clock i in the global timescale when receiving the message from node j. And let $\{t_{ik}\}_{k=0}^{+\infty}$ be the set of discrete transmitting instants of node i from the global view. Then, t_{ik} should be a monotonically increasing function of k. So $T_i^k = t_{i(k+1)} - t_{ik}$ can represent the transmitting period of node i. Since global physical time is unavailable to all nodes, the execution of receiving sampling during the synchronization process is event-triggered in practice. Said differently, it is when node j receives the packet containing the synchronization message of node i that it records its local time as the receiving sampling time. Therefore t_{ik} usually does not equate with $\bar{t}_{jk}^i$ when $i \neq j$. However, techniques of MAC-layer time-stamping help to safely assume that broadcasting of the packet by the transmitting node and receiving of the same packet by the transmitting node's neighbours happen simultaneously. For the sake of simplicity of our model and algorithm expressions, we let $t+1, T$ stand for $\bar{t}_{i(k+1)}^j, T_i^k$ respectively.

Above all, the changing principle satisfied by each local clock's physical time is presented as

$$\tau_i(t+1) = \tau_i(t) + \Delta_i(t, T), \tag{1}$$

where $\tau_i(t)$ represents the actual time of local clock i on the global time instant t. And $\Delta_i(t, T) > 0$ denotes the time variation in clock i's local timescale, corresponding to the sampling period T. Here, it has to be assumed that values of $\Delta_i(t, T)$ are uniformly bounded. As mentioned in the previous part, the physical clock rate cannot be easily adjusted. Every node has to generate a virtual local clock whose time can be adjusted directly by a control input $v_i(t)$. And activities asking for the clock consensus all refer to the virtual local time. In this way, only virtual clock synchronization needs achieving for each node. Then the model of virtual clock with a control effect can be rewritten as

$$\begin{aligned} \overline{\tau}_i(t+1) =& \overline{\tau}_i(t) + [\tau_i(t+1) - \tau_i(t)] + v_i(t) \\ =& \overline{\tau}_i(t) + \Delta_i(t, T) + v_i(t) \end{aligned} \tag{2}$$

where $\overline{\tau}_i(t)$ denotes the virtual time of local clock i and we let $\overline{\tau}_i(0) = \tau_i(0)$. Therefore, the problem of clock synchronization can be converted to designing an algorithm whose output is $v_i(t)$ to assist all local clocks in obtaining virtual common time.

The algorithm in a discrete-time form is designed as

$$\begin{cases} \omega_i(t+1) = \varepsilon \sum\limits_{j \in \mathcal{N}_i} d_{ij}(\overline{\tau}_j(t) - \overline{\tau}_i(t)) \\ v_i(t+1) = -\sum\limits_{j \in \mathcal{N}_i} d_{ij}(\overline{\tau}_j(t) - \overline{\tau}_i(t)) + v_i(t) + \omega_i(t+1) \\ \qquad\qquad - \alpha\varepsilon \sum\limits_{j \in \mathcal{N}_i} d_{ij}(\overline{\tau}_j(t-1) - \overline{\tau}_i(t-1)) \end{cases}$$

where d_{ij} is the weight associated to the edge (j, i). Parameters ε and α will be chosen in the next section to guarantee stability and consensus. The above equations can be expressed in a vector form as

$$\begin{cases} \boldsymbol{\omega}(t+1) = -\varepsilon D\overline{\boldsymbol{\tau}}(t) \\ \boldsymbol{v}(t+1) = D\overline{\boldsymbol{\tau}}(t) + \boldsymbol{v}(t) + \boldsymbol{\omega}(t+1) - \alpha\boldsymbol{\omega}(t) \end{cases} \tag{3}$$

where D consisting of d_{ij} is a Laplacian matrix. It is assumed that the message channel topology of the whole network is connected undirected. Then under this assumption, D is symmetric with zero as its simple eigenvalue.

Remark 1: We, inspired by the method of transfer function, take $D\overline{\boldsymbol{\tau}}(t)$ as the input and take $\boldsymbol{v}(t)$ as the output of the algorithm. The scheme of the algorithm is shown in Fig.1, where

$$Q(z) = -\frac{1}{z - \alpha}, G(z) = \frac{(\varepsilon - 1)z - \alpha\varepsilon}{(z-1)\varepsilon}, H(z) = z - 1.$$

In this way, the problem can be converted into solving a feedback control problem whose system is LTI (linear and

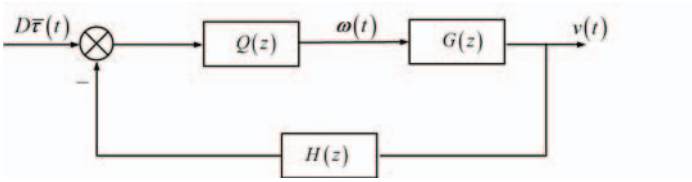

Fig. 1. The scheme of control algorithm

time-invariant) and free of model errors and uncertainties. Then, the closed-loop transfer function of the feedback system is obtained as

$$G_{cls}(z) = \frac{(1-\varepsilon)z + \alpha\varepsilon}{z(z-1)}$$

which figures out that the system is stable. In other words, as long as the synchronization message is exchanged regularly enough, the value range of the system output $\boldsymbol{v}(t)$ is bounded. It ensures the algorithm to adjust the local clocks' virtual time in a practical way.

Remark 2: Another form of expression can be derived from (3) as

$$\boldsymbol{v}(t+1) = \boldsymbol{v}(t) + (1-\varepsilon)D\overline{\boldsymbol{\tau}}(t) + \alpha\varepsilon D\overline{\boldsymbol{\tau}}(t-1). \quad (4)$$

It shows that $\boldsymbol{v}(t+1)$ can be treated as the update of an estimator. In other words, errors from the input $\overline{\tau}_j(t) - \overline{\tau}_i(t)$ do not affect $v_i(t+1)$ directly. Hence this algorithm provides a filtering effect of noisy inputs (short-term jitter of clock crystal, measurement and quantization errors, etc.) mainly from $\Delta_i(t,T)$, which is an advantage over algorithms of normal controllers. On the other hand, if we let $\boldsymbol{\zeta}(t)$ denote the error between the input and output of the algorithm, we can also get

$$\boldsymbol{v}(t+1) = a_1\boldsymbol{v}(t) + a_2\boldsymbol{v}(t-1) + a_3\boldsymbol{\zeta}(t) + a_2\boldsymbol{\zeta}(t-1)$$

where $a_1 = 2-\varepsilon, a_2 = 1-\varepsilon, a_3 = \alpha\varepsilon$ represent coefficients of the linear combination. This also shows that the algorithm is able to eliminate static errors.

IV. Stability and Consensus Analysis

To analyze the stability of the whole system after adding this algorithm, we can list the state-space expression of the discrete-time system. Since the Laplacian matrix D is symmetric, there should exist a unitary matrix U satisfying that

$$U^H DU = \Lambda = \mathrm{diag}\{\lambda_1, \cdots, \lambda_N\},$$

where U^H denotes Hermitian transpose of U and λ_i represents one of the eigenvalues of D. Since the underlying graph of the communication topology can be treated as a connected undirected graph, we can have the relation that $0 = \lambda_1 < \lambda_2 \cdots \leq \lambda_N$. According to the model presented in (2) and the synchronization algorithm (3), the whole dynamic system can be summarized in the following way:

$$\begin{pmatrix}\overline{\boldsymbol{\tau}}(t+1)\\ \boldsymbol{v}(t+1)\\ \boldsymbol{\omega}(t+1)\end{pmatrix} = \begin{pmatrix} I & I & \mathbf{0}\\ (1-\varepsilon)D & I & -\alpha I\\ -\varepsilon D & \mathbf{0} & \mathbf{0}\end{pmatrix}\begin{pmatrix}\overline{\boldsymbol{\tau}}(t)\\ \boldsymbol{v}(t)\\ \boldsymbol{\omega}(t)\end{pmatrix} + \begin{pmatrix}\boldsymbol{\Delta}(t,T)\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix},$$

where I denotes an N-dimension identity matrix.

Then, we utilize the unitary transformation to make the expression of the system decoupled by letting

$$\begin{cases} \hat{\boldsymbol{\tau}}(t) = U^H\overline{\boldsymbol{\tau}}(t), & \hat{\boldsymbol{\omega}}(t) = U^H\boldsymbol{\omega}(t)\\ \hat{\boldsymbol{v}}(t) = U^H\boldsymbol{v}(t), & \hat{\boldsymbol{\Delta}}(t,T) = U^H\boldsymbol{\Delta}(t,T)\end{cases}. \quad (5)$$

Let $\overline{\boldsymbol{U}}^H = \mathrm{diag}(U^H, U^H, U^H)$. Then, applying $\overline{\boldsymbol{U}}^H$ to the above expression of the system can yield that

$$\begin{aligned}\overline{\boldsymbol{U}}^H\begin{pmatrix}\overline{\boldsymbol{\tau}}(t+1)\\ \boldsymbol{v}(t+1)\\ \boldsymbol{\omega}(t+1)\end{pmatrix} =& \overline{\boldsymbol{U}}^H\begin{pmatrix} I & I & \mathbf{0}\\ (1-\varepsilon)D & I & -\alpha I\\ -\varepsilon D & \mathbf{0} & \mathbf{0}\end{pmatrix}\overline{\boldsymbol{U}}\overline{\boldsymbol{U}}^H\begin{pmatrix}\overline{\boldsymbol{\tau}}(t)\\ \boldsymbol{v}(t)\\ \boldsymbol{\omega}(t)\end{pmatrix}\\ &+ \overline{\boldsymbol{U}}^H\begin{pmatrix}\boldsymbol{\Delta}(t,T)\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix}\end{aligned}$$

which with (5) further infers a decoupled vector form as

$$\begin{pmatrix}\hat{\boldsymbol{\tau}}(t+1)\\ \hat{\boldsymbol{v}}(t+1)\\ \hat{\boldsymbol{\omega}}(t+1)\end{pmatrix} = \begin{pmatrix} I & I & \mathbf{0}\\ (1-\varepsilon)\Lambda & I & -\alpha I\\ -\varepsilon\Lambda & \mathbf{0} & \mathbf{0}\end{pmatrix}\begin{pmatrix}\hat{\boldsymbol{\tau}}(t)\\ \hat{\boldsymbol{v}}(t)\\ \hat{\boldsymbol{\omega}}(t)\end{pmatrix} + \begin{pmatrix}\hat{\boldsymbol{\Delta}}(t,T)\\ \mathbf{0}\\ \mathbf{0}\end{pmatrix}.$$

Then the component-wise equation can be expressed as

$$\begin{pmatrix}\hat{\tau}_i(t+1)\\ \hat{v}_i(t+1)\\ \hat{\omega}_i(t+1)\end{pmatrix} = \begin{pmatrix} 1 & 1 & 0\\ (1-\varepsilon)\lambda_i & 1 & -\alpha\\ -\varepsilon\lambda_i & 0 & 0\end{pmatrix}\begin{pmatrix}\hat{\tau}_i(t)\\ \hat{v}_i(t)\\ \hat{\omega}_i(t)\end{pmatrix} + \begin{pmatrix}\hat{\Delta}_i(t,T)\\ 0\\ 0\end{pmatrix}.$$

To ensure the stability of the whole discrete-time system, there are certain extra constraints for the choice of parameters in the designed algorithm. As is well known, with values of $\Delta_i(t,T)$ being bounded, the system will be input-to-state stable if all eigenvalues of the system matrix are strictly included in a unit circle. Then according to the condition, we can utilize the eigenvalue expression,

$$\begin{aligned} f(\delta) =& \begin{vmatrix}\delta-1 & -1 & 0\\ (\varepsilon-1)\lambda_i & \delta-1 & \alpha\\ \varepsilon\lambda_i & 0 & \delta\end{vmatrix}\\ =& \delta^3 - 2\delta^2 + [1+(\varepsilon-1)\lambda_i]\delta - \alpha\varepsilon\lambda_i.\end{aligned}$$

Let $f(\delta) = 0$, then the three eigenvalues δ_1, δ_2 and δ_3 are also roots of the cubic eigenvalue equation. According to the constraints that $|\delta_1| < 1, |\delta_2| < 1$ and $|\delta_3| < 1$, there are many methods of calculating the ranges of those parameters. Here we only provide one way, using the method of Jury Stability Criterion. Then the test chart can be obtained as Table I, where

$$b_0 = \begin{vmatrix}-\alpha\varepsilon\lambda_i & 1\\ 1 & -\alpha\varepsilon\lambda_i\end{vmatrix}, \quad b_1 = \begin{vmatrix}-\alpha\varepsilon\lambda_i & -2\\ 1 & 1+(\varepsilon-1)\lambda_i\end{vmatrix},$$
$$b_2 = \begin{vmatrix}-\alpha\varepsilon\lambda_i & 1+(\varepsilon-1)\lambda_i\\ 1 & -2\end{vmatrix}.$$

Then we can get the following equations as constraints due to the criterion

$$\begin{cases} f(1) > 0, f(-1) < 0\\ |-\alpha\varepsilon\lambda_i| < 1, |b_0| > |b_2|\end{cases}.$$

Therefore, the value range of ε can be figured out as $1 < \varepsilon < 1 + \frac{1}{\lambda_i}$. After choosing the value of ε, the range of α should be $0 < \alpha < 1 - \frac{1}{\varepsilon}$.

TABLE I
CHART OF JURY STABILITY CRITERION

Linage	δ^0	δ^1	δ^2	δ^3
1	$-\alpha\varepsilon\lambda_i$	$1+(\varepsilon-1)\lambda_i$	-2	1
2	1	-2	$1+(\varepsilon-1)\lambda_i$	$-\alpha\varepsilon\lambda_i$
3	b_0	b_1	b_2	

With the assumption that the maximum value of all eigenvalues can be obtained, we can get the two exact value ranges by letting $\lambda_{\max}$ take the place of λ_i.

Remark 3: Many methods are available in literature like [10] to obtain the upper bound value $\bar{\lambda}$ of the eigenvalues only according to attributes of the graph. We can choose an ε satisfying $\varepsilon > 1$ so that $0 < \bar{\lambda} < \frac{1}{\varepsilon-1}$. In this way, we can design the parameters ε, α off-line without requesting for eigenvalue information of the Laplacian matrix—kind of global information.

However, the simplicity of designing only two parameters is at the expense of stability performance. If one prefers attaching more emphasis on this index, our algorithm can be modified as follows:

$$\begin{cases} \boldsymbol{\omega}(t+1) = -\Upsilon D\boldsymbol{\tau}(t) \\ \boldsymbol{v}(t+1) = D\boldsymbol{\tau}(t) + \boldsymbol{v}(t) + \boldsymbol{\omega}(t+1) - \alpha\boldsymbol{\omega}(t) \end{cases}$$

where $\Upsilon = \mathrm{diag}\{\varepsilon_1, \cdots, \varepsilon_N\}$ and $\alpha = \mathrm{diag}\{\alpha_1, \cdots, \alpha_N\}$. In this way, we have to design more parameters satisfying $1 < \varepsilon_i < 1 + \frac{1}{\lambda_i}$ and $0 < \alpha_i < 1 - \frac{1}{\varepsilon_i}$. However, we can get better performance of stability.

Then we consider the consensus attribute brought about by the algorithm. Let

$$e_i(t) = \overline{\tau}_i(t) - \frac{1}{N}\sum_{j=1}^{N}\overline{\tau}_j(t).$$

We have

$$\boldsymbol{e}(t) = (I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\overline{\boldsymbol{\tau}}(t), \tag{6}$$

where $\mathbf{1}$ represents an N-dimension column vector whose all entries are 1. From the expression of $\boldsymbol{e}(t)$, we can see that the consensus problem of $\overline{\boldsymbol{\tau}}(t)$ can be solved if $\boldsymbol{e}(t)$ converges to zero. Then with the relation that $D\mathbf{1} = U\Lambda U^H\mathbf{1} = 0$,

$$\Lambda U^H\mathbf{1} = 0$$

can be obtained. It can be further inferred that

$$U^H\mathbf{1} = (x_1, 0, \cdots, 0)^T,$$

where x_1 denotes the first element of this column vector. Then the following expression can be obtained as

$$\begin{aligned} U^H(I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)U =& I - \frac{1}{N}U^H\mathbf{1}\mathbf{1}^TU \\ =& \mathrm{diag}\{1 - \frac{1}{N}x_1^2, I_{N-1}\}. \end{aligned}$$

Due to the fact that $I - \frac{1}{N}\mathbf{1}\mathbf{1}^T$ is also a symmetric Laplacian matrix with a special form, zero must be an eigenvalue of it. So, $1 - \frac{1}{N}x_1^2 = 0$, which helps to get the following equation

$$U^H(I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)U = \mathrm{diag}\{0, I_{N-1}\}.$$

With transformation $\hat{\boldsymbol{e}}(t) = U^H\boldsymbol{e}(t)$, the equation of $\boldsymbol{e}(t)$ can be transformed as

$$U^H\boldsymbol{e}(t) = U^H(I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)UU^H\overline{\boldsymbol{\tau}}(t),$$

which infers

$$\hat{\boldsymbol{e}}(t) = \mathrm{diag}\{0, I_{N-1}\}\hat{\boldsymbol{\tau}}(t). \tag{7}$$

From the above expression, we can see that $\hat{e}_1(t) \equiv 0$, while $\hat{e}_i(t) = \hat{\tau}_i(t)$ when $i \geq 2$.

On the other hand, when t tends to infinity, the following equations can be obtained according to the component-wise form of the state-space expression as

$$\begin{cases} \hat{v}_i(t+1) = (1-\varepsilon)\lambda_i\hat{\tau}_i(t) + \hat{v}_i(t) - \alpha\hat{\omega}_i(t), t \to \infty \\ \hat{\omega}_i(t+1) = -\varepsilon\lambda_i\hat{\tau}_i(t), t \to \infty \end{cases}.$$

From the equations, we can figure out that

$$(\alpha\varepsilon + 1 - \varepsilon)\lambda_i\hat{\tau}_i(t) = 0, t \to \infty.$$

Then the facts that $0 = \lambda_1 < \lambda_2 \cdots \leq \lambda_N$ and $\alpha\varepsilon + 1 - \varepsilon \neq 0$ help to conclude that $\hat{\tau}_i(t) \equiv 0$ when $i \geq 2, t \to \infty$. Combining this inference with the relation inferred from (7), we can get that $\boldsymbol{e}(t) = U\hat{\boldsymbol{e}}(t) \equiv 0, t \to \infty$. Then till now, it has been proved that virtual local time of all nodes under the algorithm (3) can achieve asymptotic consensus.

To make the virtual reference time more precise, the case $i = 1$ is still the focus of the consensus problem. From the component-form state-space expression of the system, we have

$$\begin{cases} \hat{\tau}_1(t+1) = \hat{\tau}_1(t) + v_1(t) + \hat{\Delta}_1(t,T) \\ \hat{v}_1(t+1) = \hat{v}_1(t) - \alpha\hat{\omega}_1(t) \\ \hat{\omega}_1(t+1) = \hat{\omega}_1(t) = 0 \end{cases}.$$

From the above expressions, we can obtain the recursion expression of $\hat{\tau}_1(t)$ as

$$\hat{\tau}_1(t) = \sum_{j=0}^{t-T}\hat{\Delta}_1(j,T) + \hat{\tau}_1(0) + (t/T)\hat{v}_1(0).$$

It can be seen from the last term that $\hat{v}_1(0)$ is only a constant factor of t/T. To make the later analysis easier, we can remove the uncertainty by letting $\hat{v}_1(0) = 0$ when the algorithm is run. In this way, we get a more compact expression

$$\hat{\tau}_1(t) = \sum_{j=0}^{t-T}\hat{\Delta}_1(j,T) + \hat{\tau}_1(0).$$

Now, we can find out what the value of the virtual reference clock time is. According to above statements, we can get the process of inference as follows.

$$\begin{aligned} \lim_{t\to\infty}\overline{\boldsymbol{\tau}}(t) &= \lim_{t\to\infty}U\hat{\boldsymbol{\tau}}(t) \\ &= \lim_{t\to\infty}U\begin{pmatrix}\hat{\tau}_1(t)\\ \vdots \\ \hat{\tau}_N(t)\end{pmatrix} \end{aligned}$$

$$= U \begin{pmatrix} \sum_{j=0}^{t-T} \hat{\Delta}_1(j,T) + \hat{\tau}_1(0) \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$$= U\{\mathrm{diag}\{1, \mathbf{0}_{N-1}\} U^H [\sum_{j=0}^{t-T} \mathbf{\Delta}(j,T) + \overline{\boldsymbol{\tau}}(0)]\}$$

$$= U\{(I - \mathrm{diag}\{0, I_{N-1}\}) U^H [\sum_{j=0}^{t-T} \mathbf{\Delta}(j,T) + \overline{\boldsymbol{\tau}}(0)]\}$$

$$= [I - (I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)][\sum_{j=0}^{t-T} \mathbf{\Delta}(j,T) + \overline{\boldsymbol{\tau}}(0)]$$

$$= \frac{1}{N}\mathbf{1}\mathbf{1}^T [\sum_{j=0}^{t-T} \mathbf{\Delta}(j,T) + \overline{\boldsymbol{\tau}}(0)].$$

Then the virtual reference clock time can be denoted as

$$\tau_c(t) = \frac{1}{N} \sum_{i=1}^{N} (\sum_{j=0}^{t-T} \Delta_i(j,T) + \tau_i(0)). \tag{8}$$

Remark 4: From the result, we can see that time variations can achieve the consensus value $\frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{t-T}\Delta_i(j,T)$ eventually. Simultaneously, the initial time errors also converge to the virtual common value $\frac{1}{N}\sum_{i=1}^{N}\tau_i(0)$ under implementation of the algorithm. What is more, it is worth pointing out that since the algorithm focuses on the consensus of time variations of each local clock, whether clock skews change according to global physical time or not will not affect this consensus result.

It can be easily verified that the result can be applied in the model where the slope of local clock is constant. Then, the physical time model of local clock i can be expressed as $\tau_i(t) = \alpha_i t + o_i$, where α_i denotes the constant physical skew and $o_i = \tau_i(0)$ represents the initial offset of clock i. In this case, the virtual reference time can be simplified to the form as

$$\tau_c(t) = \frac{1}{N} \sum_{i=1}^{N} (\alpha_i T_i t + \tau_i(0)). \tag{9}$$

In other words, let $\alpha_c, T_c, \tau_c(0)$ denote the slope, transmission period and initial time of the virtual reference clock, respectively, then $\alpha_c = \frac{1}{NT_c}(\alpha_1 T_1 + \cdots + \alpha_N T_N)$ and $\tau_c(0) = \frac{1}{N}(\tau_1(0) + \cdots + \tau_N(0))$. So, under this condition, the consensus of local time variations also means the consensus of local clock skews.

Before we focus on the case with time-varying skews, we have to know four main factors—crystal ageing, imperfections, short-term jitter and thermal stress—that affect the physical clock rate. However, the aging process is rather slow that it does not stay in the same time scale with the synchronization period. Imperfections contribute only one bounded constant value to the skew in a long period. Short-term jitter is a regularly fast process with a much smaller time scale than T, which means that it can be treated as one kind of noise. This effect can be reduced to certain extent by the filtering function of our algorithm. As for the temperature influence, we have the relation expression between the crystal frequency $f(t)$ and its temperature $\theta_x(t)$ as follows according to [12].

$$f(t) = f_0(1 + \frac{\beta}{10^6}(\theta_x(t) - \theta_0)^2)$$

where θ_0 is the nominal temperature corresponding with the nominal frequency f_0. $\beta < 0$ is a fixed parameter with a unit as $ppm/^\circ C^2$. Due to [11], $\theta_x(t)$'s value is bounded with slow variations, which infers that $f(t)$ is slowly and slightly changing in a bounded value interval. From the statements above, it can be implied that the value of $\sum_{j=0}^{t-T}\Delta_i(j,T)$ is bounded. Therefore, clocks can achieve a consensus value of virtual time with bounded control inputs, even if their physical rates vary according to time.

V. Implementation

In this section, we explain certain details of the algorithm in practical application. In one WSN, if two neighbouring nodes have enough power, a message channel can be established between them. So the assumption that the channel graph is undirected is reasonable. However, as for the communication graphs, they are usually unidirectional graphs at the sampling instants for the reason that each node cannot receive and transmit packets simultaneously. More precisely, the half duplex attribute of the channel makes asynchronous communication topologies exist as extended neighbor graphs at event instants. Then we have the following communication ways and explanations to make our algorithm applicable.

When being applied to WSN, this synchronization method can take a *pseudo*-flooding scheme as its communication means. Flooding means receiving nodes would be triggered by the incoming time-stamp to record their virtual local time of its arrival, then broadcast their own synchronization messages on to their neighbours on transmitting instants periodically, after updating the time-stamps. It is inferred that each node does not need to wait for all its neighbours' messages before updating its virtual time. This way of communication and updating raises up the robustness against node failure and packet losses in WSNs. The reason why we call it *pseudo* is that flooding method usually requests for a root/master node, while this algorithm presented in the paper does not. Naturally, this communication means generates non-symmetric stochastic matrices.

Considering that clock rates of distinct nodes are different, the sampling intervals of different local clocks are various, namely, $T_i^k \neq T_j^k$, when $i \neq j$. Besides, drifts of each local clock rate may lead to $T_i^k \neq T_i^{k+1}$. Confronted with the situations, implementation of the proposed algorithm with asynchronous communication can be proved practical in the following way. Let

$$\mathfrak{T} = \bigcup_k \{t_{jk}\}_{j=1}^N = \{\mathcal{T}_0, \mathcal{T}_1, \cdots, \mathcal{T}_\nu, \cdots\},$$

i.e., we rearrange all the transmitting instants of all nodes in such an order that $\mathcal{T}_\nu$ is a monotonically increasing function of ν. Since each local clock drifts at a different speed, there

should exist a $T^k_{\min}$ and a $T^k_{\max}$ in the kth transmission. And $T^{k+1}_{\max}$ may not equate with $T^k_{\max}$ for the fact that the physical rate of each node may also vary according to time. Then we can define $\bar{j}^k$ as $\bar{j}^k = \arg_j \max T^k_j$. The component which corresponds with $t_{\bar{j}^k k}$ can be found in the ordered set $\mathfrak{T}$, namely $\mathcal{T}_{\nu_k}$. Without loss of generality, we can translate the time window from $[t_{j0}, t_{j0} + kT]$ to $[0, kT]$. Then, $\nu_{k+1} - \nu_k$ must satisfy the lose constraints

$$N \le \nu_{k+1} - \nu_k \le [\frac{\max\{T^k_{\max}, T^{k+1}_{\max}\}}{\min\{T^k_{\min}, T^{k+1}_{\min}\}})N$$

where $[\cdot)$ means the smallest integer value no less than its element. Since we assume that drifts of local physical clock rates are slight, it can be ensured that every node transmits at least once in the time interval $[\mathcal{T}_{\nu_k}, \mathcal{T}_{\nu_{k+1}}]$, i.e., there exits $\overline{k}$ and $\overline{\nu}$ so that $\mathcal{T}_{\overline{\nu}} = t_{\bar{j}\overline{k}}$ for $\forall j, \forall k$, where $\nu_k \le \overline{\nu} \le \nu_{k+1}$. Then the combination of Theorem 1 in [7] and above statements helps infer that the union of instant communication graphs can be treated as one connected undirected graph in the time window $[\mathcal{T}_{\nu_k}, \mathcal{T}_{\nu_{k+1}}]$ when the algorithm can be applied.

VI. Numerical Simulations

In this section, we present certain simulation results to demonstrate the performance of the proposed algorithm. One WSN with six sensors is simulated. According to the choosing methods of parameters in the part of stability analysis, we can let $\varepsilon = 1.32$ and $\alpha = 0.22$. Then we can get Fig.2, which shows changes of virtual time of all local clocks according to the global physical time. From Fig. 2, we can find out that the initial time errors have been compensated and the clock synchronization has been achieved asymptotically. Fig. 3 shows the consensus of time variations of all local clocks.

VII. Conclusion and Future Work

This paper presented a new synchronization algorithm which can achieve the consensus of clock rates and that of initial time simultaneously. Its control input is bounded with robustness to noisy inputs. Also this method is robust against node failure and packet losses. Future work may include the optimization of the parameters choice to gain better performance of stability and consensus.

Acknowledgment

This work is supported by National Natural Science Foundation (NNSF) of China under Grant 61273113 and Zhejiang Provincial Natural Science Foundation of China L-R13F030002.

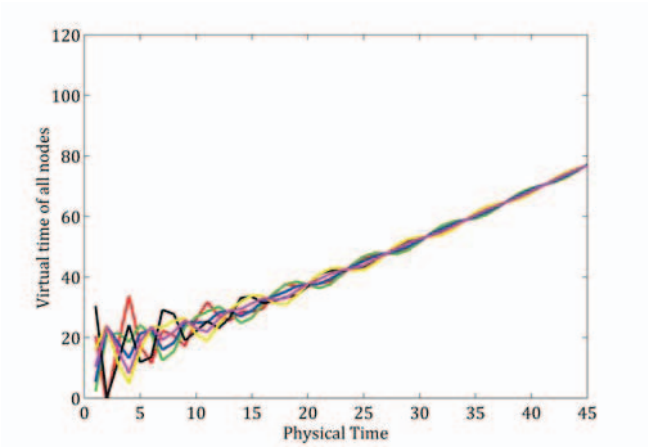

Fig. 2. Change of virtual time of all local clocks

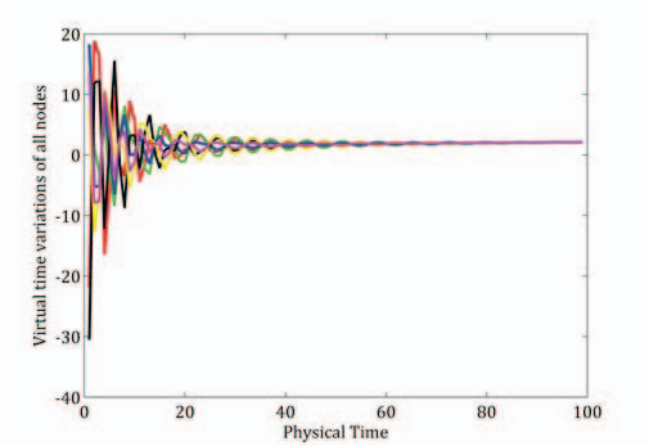

Fig. 3. Time variations of all local clocks

References

[1] Y. R. Faizulkhakov, "Time synchronization methods for wireless sensor networks: A survey," Programming and Computing Software, vol. 79, no. 4, pp. 214–226, 2007.

[2] B. Kaur, A. Kaur., "A survey of time synchronization protocols for wireless sensor networks," International Journal of Computer Science and Mobile Computing , vol. 2, no.9, pp. 100–106, 2013.

[3] J. Elson, L. Girod, and D. Estrin, "Finegrained network time synchronization using reference broadcasts," in Proc. 5th Symp. Operat. Syst. Design Implement., vol. 36. Dec. 2002, pp. 147–163.

[4] W. Su and I. Akyildiz, "Time-diffusion synchronization protocol for wireless sensor networks," IEEE/ACM Trans. Netw., vol. 13, no. 2, pp. 384–397, Apr. 2005.

[5] S. Ganeriwal, R. Kumar, and M. Srivastava, "Timing-sync protocol for sensor networks," in Proc. 1st ACM Conf. Embed. Netw. Sensor Syst., Nov. 2003, pp. 138–149.

[6] D. Mills, "Internet time synchronization: The network time protocol," IEEE Trans. Commun., vol. 39, no. 10, pp. 1482–1493, Oct. 1991.

[7] L. Schenato, F. Fiorentin, "Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks," Automatica, vol. 47, no. 9, pp. 1878–1886, 2011.

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

[9] F. Terraneo, L. Rinaldi, M. Maggio, A. V. Papadopoulos, and A. Leva, "FLOPSYNC-2: efficient monotonic clock synchronisation," in Proceedings of the 35th IEEE Real-Time Systems Symposium (RTSS), 2014, pp. 11–20.

[10] X. D. Zhang, "Two sharp upper bounds for the Laplacian eigenvalues," Linear Algebra and its Applications, vol. 376, no. 1, pp. 207-213, 2004.

[11] T. Schmid, P. Dutta, and M. B. Srivastava, "High-resolution, low-power time synchronization an oxymoron no more," in Proc. 9th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN). New York, NY, USA: ACM, 2010, pp. 151–161.

[12] M. Nakazawa, Y. Nakamura, and S. Miyashita, "Frequency-temperature characteristics of quartz crystal flexure bars and quartz crystal tuning forks," IEEE Transactions on Sonics and Ultrasonics, vol. 26, no. 5, pp. 369–376, 1979.