A New Distributed Method for Clock Synchronization in Sensor Networks

Yong Qiao[1], Wenlun Yang[1], Minyue Fu[2,1]

1. State Key Laboratory of Industrial Control Technology and School of Control Science and Engineering, Zhejiang University, Hangzhou, 310027, P. R. China
E-mail: yongqiao@zju.edu.cn, yangwenlun@zju.edu.cn

2. School of Electrical Engineering and Computer Science, University of Newcastle, NSW 2308 Australia
E-mail: minyue.fu@newcastle.edu.au

Abstract: In this paper, we present a new distributed protocol to achieve the synchronization of time variations and initial times simultaneously. The protocol combines the design methods of controller and estimator to obtain more precise synchronization with higher robustness against noisy inputs mainly generated by crystal oscillators of clocks. Furthermore, the control input is ensured bounded while achieving the clock synchronization, which makes our protocol more applicable in practice. The implementation of the protocol allows sampling in the receiving end to be event-triggered with a specifically designed communication scheme. Numerical simulations are shown to illustrate the performance of the protocol.

Key Words: Distributed Algorithm, Clock Synchronization, Consensus Control, Sensor Networks

1 INTRODUCTION

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 and peer-to-peer synchronization. A master-to-slave protocol assigns one node as the master and other nodes as slaves. Reference Broadcast Synchronization (RBS) reduces the non-deterministic time-delay by dividing the net into multiple clusters with a master node in each one. And energy is conserved via post-facto communication scheme [2]. Time-Diffusion Protocol (TDP) [3] achieves a balanced time based on an election and reelection cycle, in which nodes become masters or diffused-leaders on their own decision. And then a radial tree structure can be formed to execute the iterative, weighted averaging process. Timing-sync Protocol for Sensor Networks (TPSN) [4] achieves pair-wise synchronization along the branches of a spanning tree built in the preparation phase of the protocol. Master-node failure and packet losses which influence the performance of wireless sensor networks (WSNs), are common problems with which above methods are confronted. Flooding Time Synchronization Protocol (FTSP) [5] can achieve robustness to the root failure and variations of dynamic topology to some extent.

In contrast, a peer-to-peer protocol requires no center nodes, i.e., it is a distributed protocol. Average Time Sync (ATS) [6] uses a cascade of two consensus algorithms to make all the clocks converge to a virtual reference one by tuning compensation parameters for each node. Distributed Time Sync [7] realizes clock synchronization by solving a gradient-descent optimization problem. Any node can act as a reference node without adjusting its own physical clock during the process and neighboring nodes exchange one-to-one time-stamp packets. The peer-to-peer methods can roughly be classified into internal synchronization and external synchronization. In most researches mentioned above, this pattern is not explained explicitly in the modeling part. Since the latter method seeks to achieve synchronization without resetting or correcting each sensor's physical clock, external or virtual clocks should be modeled at first. FLOPSYNC-2 [8] and CCTS [9] present relatively specific relationships between the actual and virtual clocks. And our paper focuses on the peer-to-peer external synchronization for the reason that in most application environment of WSNs, the local clocks of sensors cannot be adjusted physically.

Our research has been aiming to develop a new distributed protocol for clock synchronization. This is motivated by the fact that most existing peer-to-peer protocols achieve the consensus of 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 a high communication and computation overhead. In a previous paper, we combined these two tasks and solved them jointly. However, it is a theoretical method which actually relies on the knowledge of global physical time with the communication scheme proposed previously. Also, it is implicitly assumed that the clock rates are constant. In this paper, our protocol is modified to

This work is supported by National Natural Science Foundation of China under Grant 61273113 and Zhejiang Provincial Natural Science Foundation of China LR13F030002.

978-1-5090-4657-7/17/$31.00 ©2017 IEEE

combine the advantages of the controller-design method as in [8] with those of the estimator-based one as in [6] to gain better robustness against clock rate drifts and noisy measurements. And previous communication scheme is corrected to be a specifically designed one to enable the protocol to be independent on the global time, which ensures its practical application. Furthermore, the control inputs used to adjust the local time are proved to be bounded. Following the implementation ideas in [6], we have also addressed the problem with asynchronous communication in the actual implementation of the protocol. This also enhances the robustness to the node failure and packet losses.

2 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: Its ijth entry is given by

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

It is well known that zero is an eigenvalue of L with $\mathbf{1}$ as its 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. An *extended neighbour graph*, denoted as $\mathcal{D}(t) = (\mathcal{V}, \mathcal{E}(t))$, is a digraph with at most one arc between each ordered pair of nodes and with exactly one self-loop at each vertex at event time t. If all the extended neighbor graphs in a time interval make up $\mathcal{G} = \bigcup_{t_0}^{t_1} \{\mathcal{D}(t)\}$, then vertex $j \in \mathcal{N}_i$ in $\mathcal{G}$ as long as this establishes in $\mathcal{D}(t)$ at least once in the interval mentioned above.

3 Clock Models and Proposed Synchronization Protocol

Above all, the model of each local clock's physical time is given as follows:

$$\tau_i(t) = \alpha_i(t)t + o_i \tag{1}$$

where $\alpha_i(t) > 0$ denotes the physical clock skew of node i. It is worth pointing out that in previous literature α_i was always simplified as a constant value to make the model more convenient for analysis. However, here $\alpha_i(t)$ is time-varying, which is more practical. t denotes the physical time in the global timescale and $\tau_i(t)$ denotes the local physical time of node i corresponding with the global time instant t. o_i is the initial time error between $\tau_i(t)$ and t when $t = 0$, i.e., $o_i = \tau_i(0)$.

According to the model, $\tau_i(t)$ is a monotonically increasing function of t and there is a one-to-many mapping relationship between t and $\tau_i(t), i = 1, 2, \cdots, N$. Said differently, the local physical time of all nodes may be different from each other on the same global time instant t without clock synchronization. Therefore, the purpose of the clock synchronization is to adjust all the local time to gain the same one-to-one mapping relationship with t eventually. Then, after having achieved the synchronization, the local time value of all nodes should equate with each other on the same physical instant in the global timescale.

However, continuous clock synchronization poses a high demand for power supply, which is not practical for wireless sensor networks. So we give out a sampling model of each local clock in a discrete-time expression. The time of each local clock, instead of local clock skew and initial time error, is modeled directly as follows.

$$\tau_i(k+1) = \tau_i(k) + \Delta_i(k, T_k), \tag{2}$$

where we assume that all the nodes have the same sampling period T_k in the global timescale to have the form of expression established. $\tau_i(k + 1)$ denotes node i's local time after one sampling period from the global sampling instant t_k. And $\Delta_i(k, T_k) > 0$ corresponding with the sampling period T_k, denotes the time variation of local clock i between two neighbouring sampling instants. Again, T_k has a one-to-many mapping relationship with $\Delta_i(k, T_k), i = 1, 2, \cdots, N$ for the fact that time of different local clocks varies at a distinct speed. Said differently, corresponding with the same sampling period in the global timescale, usually $\Delta_i(k, T_k) \neq \Delta_j(k, T_k)$, when $i \neq j$.

Remark 1 *It is of much significance in pointing out that the assumption that all nodes have the same sampling period from the global view is realizable in practical application. This can be explained from two aspects. On one hand, if the application is under the condition of synchronous communication which means all nodes have the same packet transmission period in the global timescale, then just let the sampling period equate with the transmission one.*

On the other hand, a particular communication scheme can be designed. Before introducing it, certain symbols have to be defined as follows. Let $\{t_{ik}\}_{k=0}^{+\infty}$ *be the set of transmitting instants of node* i *in the global timescale.* $\{\bar{t}_{jk}^{i}\}_{k=0}^{+\infty}$ *stands for the set of node* j*'s receiving as well as sampling instants in the global timescale when it receives messages from node* i*. Since sampling of all nodes can not be executed at the same global time which is unavailable to them during the synchronization process, it is event-triggered in practice. Said differently, it is when node* j *receives the packet containing the time-stamp information of node* i *that it samples the packet's arrival time in its local timescale. Therefore usually* t_{ik} *does not equate with* $\bar{t}_{jk}^{i}$ *when* $i \neq j$ *in normal ways. However, the 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 almost simultaneously. In this way, the two symbols can stand for the same instant. Then, the communication scheme is designed as follows. First, one of the nodes wakes up from the sleeping mode (or low-power state) after having listened to the signal for clock synchronization. Without losing generality, we take node* 1 *as this* *node. Then it broadcasts its packets containing concerned information to its neighbours periodically according to its own transmission period Δ_1 in its local timescale. And the execution of all other nodes' packets transmitting and receiving is event-triggered. Specifically, when receiving the packet from node 1, node 1's neighbours sample their local time and simultaneously broadcast the packets containing their local-time information to their own neighbours. And all nodes sample and broadcast their packets only one time around the same instant. In this way, all the nodes can have the same sampling instants and period T_k as well as the same pseudo transmitting-period in the global timescale which also has a one-to-one mapping relationship with Δ_1. Actually, here we have an assumption that sampling for the in-packet and broadcasting of the out-packet of the same node are triggered at almost the same time, i.e., the time delay between them can be ignored compared to the period's timescale. However, when this time delay cannot be ignored for certain kinds of sensors or the delay accumulates strictly in a very large and sparse net, it will be an issue of applying our method with asynchronous communication, which will be discussed later in the implementation part. Additionally, since $\alpha_1(t)$ is time-varying, the same local transmission period Δ_1 of node 1 may lead to different intervals between different sampling instants in the global timescale with global physical time changing at the same pace, i.e., usually $T_k \neq T_{k+1}$. And that's why we take $\Delta_i(k, T_k)$ to denote the local time variations of all nodes. From the above statements, it is reasonable to assume that the values of $\Delta_i(k, T_k)$ are uniformly bounded.*

Since we cannot actually adjust the physical clock rate, a virtual local clock model is established for each node, whose time we can tune directly through a control input $u_i(k)$. Then only the synchronization of all virtual clocks is needed to generate the virtual common time that time critical events all refer to. With the control input, this virtual local time can be modeled as

$$\begin{aligned}\overline{\tau}_i(k+1) =& \overline{\tau}_i(k) + [\tau_i(k+1) - \tau_i(k)] + u_i(k) \\ =& \overline{\tau}_i(k) + \Delta_i(k, T_k) + u_i(k)\end{aligned} \tag{3}$$

where $\overline{\tau}_i(k)$ denotes the virtual time of local clock i corresponding with the global instant t_k and we let $\overline{\tau}_i(0) = \tau_i(0)$. Then the communication scheme still works with the packets containing virtual time-stamps. Therefore, the problem of clock synchronization can be converted to designing a protocol for $u_i(k)$ to assist all virtual local time in achieving the same value corresponding with the same global instant.

Remark 2 *Here k is just an auxiliary symbol for expression without requiring for its value information. Though the value of the global instant t_k is unavailable to all nodes, the difference $\overline{\tau}_j(k) - \overline{\tau}_i(k)$ can be obtained without the necessity in knowing the exact value of t_k, as long as $\overline{\tau}_i(k)$ and $\overline{\tau}_j(k)$ correspond with the same global instant. Therefore, that difference is the input controlled-variable in the following protocol.*

Then, the protocol which enables $u_i(k)$ to help all the virtual time achieve the synchronization of time variations and initial time errors simultaneously is proposed as follows.

$$\begin{cases}\omega_i(k+1) =\varepsilon \sum_{j\in\mathcal{N}_i} d_{ij}(\overline{\tau}_j(k) - \overline{\tau}_i(k)) \\ u_i(k+1) = \sum_{j\in\mathcal{N}_i} d_{ij}(\overline{\tau}_i(k) - \overline{\tau}_j(k)) + u_i(k) + \omega_i(k+1) \\ \qquad + \alpha\varepsilon \sum_{j\in\mathcal{N}_i} d_{ij}(\overline{\tau}_i(k-1) - \overline{\tau}_j(k-1))\end{cases}$$

where $\omega_i(0) = 0$ and d_{ij} is the edge (j, i)'s weight. Parameters ε and α are to be specified in the next section to guarantee the stability of the whole system. A vector form can be obtained from the above equations as

$$\begin{cases}\boldsymbol{\omega}(k+1) = -\varepsilon D\overline{\boldsymbol{\tau}}(k) \\ \boldsymbol{u}(k+1) = D\overline{\boldsymbol{\tau}}(k) + \boldsymbol{u}(k) + \boldsymbol{\omega}(k+1) - \alpha\boldsymbol{\omega}(k)\end{cases} \tag{4}$$

where D is a Laplacian matrix consisting of d_{ij}. Here, we have to assume that the underlying graph of the whole network's message-channel topology is connected undirected. Then $d_{ij} = d_{ji}$ establishes and D is symmetric with zero as its simple eigenvalue.

Remark 3 *Another form derived from protocol (4) can be expressed as $\boldsymbol{u}(k+1) = \boldsymbol{u}(k) + (1-\varepsilon)D\overline{\boldsymbol{\tau}}(k) + \alpha\varepsilon D\overline{\boldsymbol{\tau}}(k-1)$. Then $\boldsymbol{u}(k+1)$ can be regarded as the update of an estimator. Said differently, errors from the input $\overline{\tau}_j(k) - \overline{\tau}_i(k)$ would not directly affect $u_i(k+1)$. Hence the protocol provides a better filtering effect for noisy inputs (mainly from $\Delta_i(k, T_k)$, such as short-term jitter of clock crystals, measurement and quantization errors), which is an advantage over purely controller-based protocols. On the other hand, if we let $\boldsymbol{\zeta}(k)$ denote the error between the input and output of the protocol, we can get $\boldsymbol{u}(k+1) = a_1\boldsymbol{u}(k) + a_2\boldsymbol{u}(k-1) + a_3\boldsymbol{\zeta}(k) + a_2\boldsymbol{\zeta}(k-1)$ where $a_1 = 2-\varepsilon, a_2 = 1-\varepsilon, a_3 = \alpha\varepsilon$ represent the coefficients of the linear combination. This also shows that this protocol can eliminate the static errors.*

Remark 4 *It can be seen from following statements that the value of $\boldsymbol{u}(k)$ is bounded so as to ensure the applicability of the protocol in practice. We, inspired by the method of transfer function, take $D\overline{\boldsymbol{\tau}}(k)$ as the input and $\boldsymbol{u}(k)$ as the output of the protocol. Then, the scheme of the protocol can be obtained as Figure 1, where $Q(z) = -\frac{1}{z-\alpha}, G(z) = \frac{(\varepsilon-1)z-\alpha\varepsilon}{(z-1)\varepsilon}$ and $H(z) = z - 1$. In this way, designing the protocol can also be treated as solving a feedback control problem whose control system is linear, time-invariant and free of model error and uncertainty. Then, the closed-loop transfer function of the feedback system is $G_{cls}(z) = \frac{(1-\varepsilon)z+\alpha\varepsilon}{z(z-1)}$ which shows the stability of the control system. In other words, as long as the message exchanges are regular enough, the system output $\boldsymbol{u}(k)$ is bounded.*

4 Stability and Consensus Analysis

First, we analyze the stability of the whole system after adding the protocol. With D being symmetric, a unitary matrix U can be found so that $U^H D U = \Lambda =$

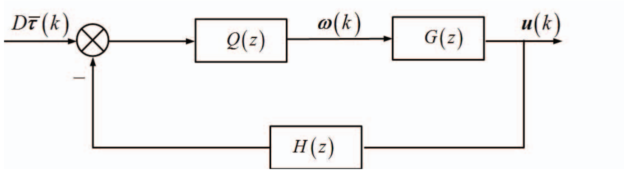

Figure 1: The scheme of protocol for the control input

$\mathrm{diag}\{\lambda_1, \cdots, \lambda_N\}$, where U^H denotes Hermitian transpose of U and λ_i represents one of the eigenvalues of the Laplacian matrix D. Owing to the connected undirected graph, all the eigenvalues of D can be labeled in such an order that $0 = \lambda_1 < \lambda_2 \le \cdots \le \lambda_N$. According to the model (3) and protocol in the vector form (4), the whole dynamic system can be expressed in the following way:

$$\begin{pmatrix} \overline{\boldsymbol{\tau}}(k+1) \\ \boldsymbol{u}(k+1) \\ \boldsymbol{\omega}(k+1) \end{pmatrix} = \begin{pmatrix} I & I & \mathbf{0} \\ (1-\varepsilon)D & I & -\alpha I \\ -\varepsilon D & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \overline{\boldsymbol{\tau}}(k) \\ \boldsymbol{u}(k) \\ \boldsymbol{\omega}(k) \end{pmatrix} + \begin{pmatrix} \boldsymbol{\Delta}(k,T_k)^T & \mathbf{0} & \mathbf{0} \end{pmatrix}^T. \tag{5}$$

Then to make the system decoupled, the unitary transformation can be utilized by letting

$$\begin{cases} \hat{\boldsymbol{\omega}}(k) = U^H \boldsymbol{\omega}(k), & \hat{\boldsymbol{u}}(k) = U^H \boldsymbol{u}(k) \\ \hat{\boldsymbol{\tau}}(k) = U^H \overline{\boldsymbol{\tau}}(k), & \hat{\boldsymbol{\Delta}}(k,T_k) = U^H \boldsymbol{\Delta}(k,T_k) \end{cases}. \tag{6}$$

Applying $\overline{\boldsymbol{U}} = \mathrm{diag}\{U, U, U\}$ to (5) yields that

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

In this way, system (5) is decoupled with D being replaced by Λ. Then, the component-wise form of the system after transformation is

$$\begin{pmatrix} \hat{\tau}_i(k+1) \\ \hat{u}_i(k+1) \\ \hat{\omega}_i(k+1) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ (1-\varepsilon)\lambda_i & 1 & -\alpha \\ -\varepsilon\lambda_i & 0 & 0 \end{pmatrix} \begin{pmatrix} \hat{\tau}_i(k) \\ \hat{u}_i(k) \\ \hat{\omega}_i(k) \end{pmatrix} + \begin{pmatrix} \hat{\Delta}_i(k,T_k) & 0 & 0 \end{pmatrix}^T. \tag{7}$$

As well known, with the values of $\hat{\Delta}_i(k,T_k)$ being bounded, the system can achieve input-to-state stability if the system matrix is schur-stable. This condition provides certain extra constraints for choosing the parameters in the system matrix. Then let the eigenvalue expression

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

equate with zero and $\delta_1, \delta_2, \delta_3$ denote the three roots of the equation, as well as the eigenvalues of the system matrix. Then the schur-stability of the system matrix is ensured as long as the constraints $|\delta_i| < 1$, for $i \in (1,2,3)$ establish. Here Jury stability criterion is utilized to calculate these value ranges of the parameters under the constraints. Then we can obtain the test chart shown as Table 1, where

Table 1: 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	

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

Also we can get the following constraints due to the criterion.

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

Thus, the solutions to the inequations can be figured out as $1 < \varepsilon < 1 + \frac{1}{\lambda_i}$ and $0 < \alpha < 1 - \frac{1}{\varepsilon}$. With the assumption that the maximum value of all eigenvalues can be obtained, we can get the two exact ranges by replacing λ_i with $\rho(D)$, where $\rho(D)$ is D's spectral radius.

Remark 5 *Certain researches like [10], show that methods are available to utilize only attributes of the underlying graph to decide the upper bound value* $\bar{\lambda}$ *of all eigenvalues. Then by choosing an* $\varepsilon > 1$ *so that* $0 < \bar{\lambda} < \frac{1}{\varepsilon-1}$, *the computation for parameters* ε, α *can be an off-line process without requesting for any global information of the Laplacian matrix.*
However, there is a tradeoff between the simplicity of designing parameters and stability performance. If one prefers attaching more emphasis on this index, the protocol can be modified as follows:

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

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

Next we show the asymptotic consensus attribute brought by the protocol of all virtual local time. Let $e_i(k) = \overline{\tau}_i(k) - \frac{1}{N}\sum_{j=1}^{N} \overline{\tau}_j(k)$, then we have

$$\boldsymbol{e}(k) = (I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)\overline{\boldsymbol{\tau}}(k). \tag{8}$$

From the expression of $\boldsymbol{e}(k)$, we can see that the asymptotical consensus problem of $\overline{\boldsymbol{\tau}}(k)$ can be solved if $\boldsymbol{e}(k)$ converges to zero. And it is easy to figure out that $I - \frac{1}{N}\mathbf{1}\mathbf{1}^T$ is also a symmetric Laplacian matrix with a special form. Then it also satisfies that $U^H(I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)U = \mathrm{diag}\{0, I_{N-1}\}$. Similarly, let $\hat{\boldsymbol{e}}(k) = U^H\boldsymbol{e}(k)$, then (8) can be transformed as

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

which implies $\hat{\boldsymbol{e}}(k) = \text{diag}\{0, I_{N-1}\}\hat{\boldsymbol{\tau}}(k)$. From this expression, we can see that

$$\hat{e}_1(k) \equiv 0, \qquad \hat{e}_i(k) = \hat{\tau}_i(k), i \geq 2. \tag{9}$$

Expression (7) also establishes when k tends to infinity, therefore we have

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

From the equations, we can figure out that $(\alpha\varepsilon + 1 - \varepsilon)\lambda_i\hat{\tau}_i(k) = 0, k \to \infty$. Then the facts that $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$ and $\alpha\varepsilon + 1 - \varepsilon < 0$ help to assert that $\hat{\tau}_i(k) \equiv 0, k \to \infty$ when $i \geq 2$. Combining this inference with (9) yields that $\hat{\boldsymbol{e}}(k) \equiv \mathbf{0}, k \to \infty$. Since the fact that zero does not belong to the spectrum of U^H, $\boldsymbol{e}(k) \equiv \mathbf{0}, k \to \infty$ can be concluded due to the relationship $\hat{\boldsymbol{e}}(k) = U^H\boldsymbol{e}(k)$. Then till now, it has been proved that asymptotic consensus of all virtual time can be achieved under protocol (4).

To show the virtual common time in an explicit form, substituting $i = 1$ and $\lambda_1 = 0$ into (7) yields that

$$\begin{cases} \hat{\tau}_1(k+1) = \hat{\tau}_1(k) + u_1(k) + \hat{\Delta}_1(k, T_k) \\ \hat{u}_1(k+1) = \hat{u}_1(k) - \alpha\hat{\omega}_1(k) \\ \hat{\omega}_1(k+1) = \hat{\omega}_1(k) = 0 \end{cases}.$$

From the above expressions, the recursion expression of $\hat{\tau}_1(k)$ can be obtained as

$$\hat{\tau}_1(k) = \sum_{j=0}^{k-1}\hat{\Delta}_1(j, T_j) + \hat{\tau}_1(0) + k\hat{u}_1(0).$$

It can be seen that $\hat{u}_1(0)$ is only a constant factor of k in the last term. We can remove the uncertainty by letting $\boldsymbol{u}(0) = 0$ when the protocol is initiated. Then a more compact expression can be shown as follows.

$$\hat{\tau}_1(k) = \sum_{j=0}^{k-1}\hat{\Delta}_1(j, T_j) + \hat{\tau}_1(0).$$

All above statements enable us to get the following process of inference.

$$\begin{aligned}
&\lim_{k\to\infty}\overline{\boldsymbol{\tau}}(k) = \lim_{k\to\infty}U\hat{\boldsymbol{\tau}}(k) \\
&= \lim_{k\to\infty}U\begin{pmatrix}\hat{\tau}_1(k) & \cdots & \hat{\tau}_N(k)\end{pmatrix}^T \\
&= U\begin{pmatrix}\sum_{j=0}^{k-1}\hat{\Delta}_1(j, T_j) + \hat{\tau}_1(0) & 0 & \cdots & 0\end{pmatrix}^T \\
&= U\{\text{diag}\{1, \mathbf{0}_{N-1}\}U^H[\sum_{j=0}^{k-1}\boldsymbol{\Delta}(j, T_j) + \overline{\boldsymbol{\tau}}(0)]\} \\
&= U\{(I - \text{diag}\{0, \boldsymbol{I}_{N-1}\})U^H[\sum_{j=0}^{k-1}\boldsymbol{\Delta}(j, T_j) + \overline{\boldsymbol{\tau}}(0)]\} \\
&= [I - (I - \frac{1}{N}\mathbf{1}\mathbf{1}^T)][\sum_{j=0}^{k-1}\boldsymbol{\Delta}(j, T_j) + \overline{\boldsymbol{\tau}}(0)] \\
&= \frac{1}{N}\mathbf{1}\mathbf{1}^T[\sum_{j=0}^{k-1}\boldsymbol{\Delta}(j, T_j) + \overline{\boldsymbol{\tau}}(0)].
\end{aligned}$$

Then the virtual common clock time can be denoted as

$$\tau_c(k) = \frac{1}{N}\sum_{i=1}^{N}(\sum_{j=0}^{k-1}\Delta_i(j, T_j) + \tau_i(0)). \tag{10}$$

Remark 6 *It can be seen from the virtual common time that the time variations converge to a common value $\frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{k-1}\Delta_i(j, T_j)$. Simultaneously, the initial time errors have also achieved the consensus value $\frac{1}{N}\sum_{i=1}^{N}\tau_i(0)$. Synchronization of time variations ensures the virtual time of all nodes to change at the same pace, which implies that local clocks do not need resynchronizing in a relative long period of global time. This reduces the workloads and energy consumption caused by data's transmission and computation.*

Additionally, no matter whether the local clock skews are time-invariant or time-varying, the virtual common time can have this synchronization form, which means wider application in practice. When the local clock skew α_i is constant, the form of virtual common time can be simplified to

$$\tau_c(k) = \frac{1}{N}\sum_{i=1}^{N}(k\alpha_i T + \tau_i(0)), \tag{11}$$

Let α_c, T, o_c denote the skew, transmission period and initial time of the virtual common clock respectively, then it can be obtained that $\alpha_c = \frac{1}{N}(\alpha_1 + \cdots + \alpha_N)$ and $o_c = \frac{1}{N}(\tau_1(0) + \cdots + \tau_N(0))$ according to model (1). So, under this condition, the synchronization of local time variations equates with that of local clock skews.

5 Implementation

In this section, we will explain certain details of the protocol in practical application. In a wireless sensor network, if two neighbouring nodes both have enough power, then a message channel can be established between them. So the assumption that the channel graph is connected undirected is reasonable. And if the channel is full duplex, the communication graph is also connected undirected. However, in some cases the channel has the attribute of half duplex, which means that each node cannot receive and transmit packets simultaneously. This makes the communication graph exist as a unidirectional graph on one global instant. In this case the following communication scheme and explanations are proposed to make our protocol applicable. A communication scheme modified from that in Remark 1 is applied as follows to make the protocol gain wider application as well as raise up the robustness to node failure and packet losses. Receiving nodes would be triggered by the incoming time-stamp to record the local physical time of its arrival and update their own virtual time-stamps immediately. This implies that each node does not need to wait for all its neighbours' messages before updating its virtual time. Then after a short period of delay-time from the message reception, the nodes broadcast their own current virtual local time to their neighbours. And the information of k contained in the packets helps to ensure each node to be triggered to broadcast their out-packet only one time in the kth transmission. This communication scheme is also the reason for the generation of extended neighbour graphs which generate non-symmetric stochastic matrices.

Now let $\mathfrak{T} = \bigcup_i \{t_{ik}\}_{k=0}^{+\infty} = \{\mathcal{T}_0, \mathcal{T}_1, \cdots, \mathcal{T}_\nu, \cdots\}$ denote the set where we rearrange all the transmitting instants of all nodes in such an order that $\mathcal{T}_\nu$ is a monotonically increasing function of ν. t_{ik} is the symbol mentioned in Remark 1. Also let ι_i denote the time delay which the broadcasting instant of node i has relative to that of node1, then there should exist a maximum delay value $\iota^k_{\max}$ in the kth transmission. We define $\bar{i} = \arg_{i\in N} \max \iota^k_i$. Then, the component corresponding with $t_{\bar{i}k}$ in the ordered set $\mathfrak{T}$ can be found and denoted as $\mathcal{T}_{\bar{\nu}_k}$. Since the transmission period T_k is determined by node 1 as mentioned in Remark 1 and generally the time scale of T_k is larger than that of time delay, $\bar{\nu}_{k+1} - \bar{\nu}_k$ satisfies the constraint $\bar{\nu}_{k+1} - \bar{\nu}_k = N$ under the above broadcasting scheme. Then, the transmitting instant of each node is ensured to appear once in the time interval $(\mathcal{T}_{\bar{\nu}_k}, \mathcal{T}_{\bar{\nu}_{k+1}}]$, i.e., for $\forall i$ there always exits k and l so that $\mathcal{T}_l = t_{i(k+1)}$, where $\bar{\nu}_k < l \le \bar{\nu}_{k+1}$, $k = 0, 1, \cdots$.

Taking the combination of Theorem 1 presented in [6] and the statements above into consideration, the union of the instant communication graphs can be treated as one connected undirected graph in the time window $(\mathcal{T}_{\bar{\nu}_k}, \mathcal{T}_{\bar{\nu}_{k+1}}]$ when the protocol can be applied.

6 Numerical Simulations

In this section, we present a simulation result of one WSN consisting of six sensors to demonstrate the performance of the proposed protocol. According to the constraints of choosing parameters, we let $\varepsilon = 1.30$ and $\alpha = 0.23$. Figure 2 shows the change of all virtual local time according to the global physical time. Figure 3 shows the synchronization of time variations of all local clocks.

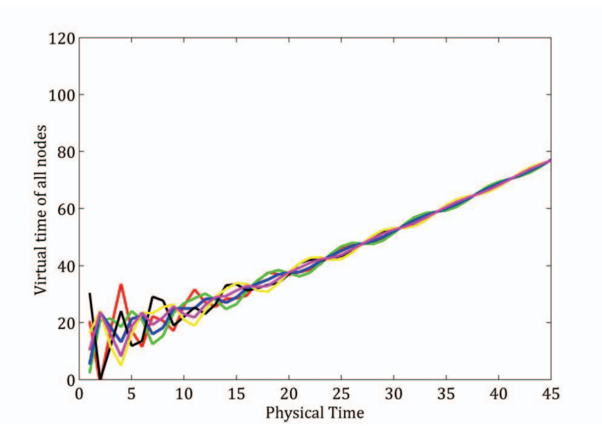

Figure 2: Change of virtual time of all local clocks

7 Conclusion and Future Work

This paper presents a new synchronization protocol which can achieve the consensus of time variations and that of initial times simultaneously. Its control input is bounded with robustness against noisy inputs. Also this method is robust against node failure and packet losses. Future work may include optimization of the parameters in this protocol to gain better performance of the system.

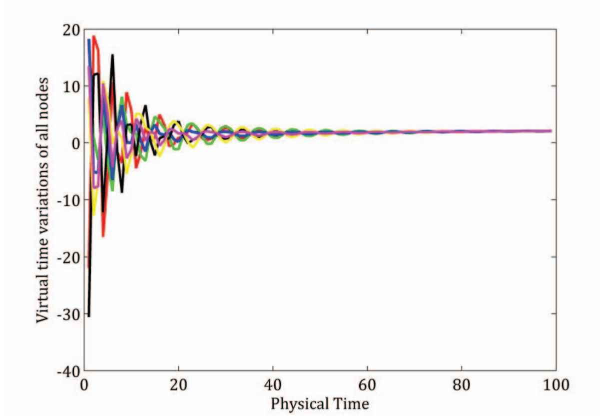

Figure 3: Time variation of all local clocks

REFERENCES

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

[2] J. Elson, L. Girod, and D. Estrin, Finegrained network time synchronization using reference broadcasts, in Proc. 5th Symp. Operat. Syst. Design Implement., New York, USA, 2002, pp. 147–163.

[3] W. Su and I. Akyildiz, Time-diffusion synchronization protocol for wireless sensor networks, IEEE/ACM Trans. Netw., Vol.13, No.2, 384–397, 2005.

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

[5] D. Mills, Internet time synchronization: The network time protocol, IEEE Trans. Commun., Vol.39, No.10, 1482–1493, 1991.

[6] L. Schenato and F. Fiorentin, Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks, Automatica, Vol.47, No.9, 1878–1886, 2011.

[7] 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, Article 8, 2007.

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

[9] J. Wu, L. Zhang, Y. Bai, and Y. Sun, Cluster-based consensus time synchronization for wireless sensor networks, IEEE Sensors Journal, Vol.15, No.3, 1404–1413, 2015.

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

[11] 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, 369–376, 1979.

[12] 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, USA, 2010, pp. 151–161.