International Journal of Robotics and Automation, Vol. 33, No. 6, 2018

QUICK TWO-WAY TIME MESSAGE EXCHANGE FOR TIME SYNCHRONIZATION IN ROBOT NETWORKS

Fanrong Shi,* Xianguo Tuo,*,** Jing Lu,* Zhenwen Ren,* and Lili Ran*

Abstract

Time synchronization is important for coordination and control in multiple robot networks. Two-way time message exchange (TTME) time synchronization is efficient, but due to variable response latency, it does not completely meet the requirements for quadrotor robot groups. Aiming to provide accurate time synchronization for robots, the proposed quick TTME synchronization starts a *downlink* after an *uplink* quickly and tries to maintain a fixed clock offset for the estimation. A timeout constraint is used to filter invalid observations and optimize clock offset estimation. This avoids the increasing clock offset caused by large software latencies and communication link delays, guarantees a fixed clock offset for TTME, and provides precise and stable clock offset estimation for time synchronization in robot networks.

Key Words

Synchronization, multiple robot, quadrotor, clock offset estimation, wireless sensor networks, quick two-way time message exchange

1. Introduction

Synchronization is an important part of key components in robot systems, such as cooperative control [1], [2], feedback control [3], [4], multi-robot coverage [5], trajectory tracking [6], [7] and formation control [8]. In this paper, we employ robot as wireless sensor nodes and discuss its time synchronization problem in the context of lightweight wireless sensor networks (WSNs). The proposed time synchronization algorithm works for coordination task and formation control in quadrotor networks. In distribute systems, the logic timing among the nodes is diverse; therefore, differences exist among the various nodes due to the initialization of their logic circuits at random moments. This also results in clock offset being introduced among the nodes. Furthermore, clock skew and offset are continuously changing due to the random clock source frequency offset. In this paper, a time synchronization algorithm is proposed to correct the local time information of nodes and drive the logic timing of networks in a consistent manner.

Almost all the time synchronization algorithms proposed for WSNs need to estimate and correct clock offset among the nodes [9], [10]. Many of these algorithms are improved by introducing clock skew compensation. Benefiting from this improvement, the negative effect from clock frequency offset can be reduced effectively so that it is possible to achieve long-term time synchronization [11]–[15]. The related literature shows that clock skew estimation is always derived from a series of accurate clock offset estimations, which is an important and difficult task.

As the hardware resources and computing capabilities of WSNs nodes are limited, clock frequency cannot be measured directly. The time synchronization algorithm always estimates the clock offset through time information transmission. The two-way time message exchange (TTME) for WSNs' time synchronization was first proposed by Ganeriwal *et al.* [9], followed by similar algorithms to improve its precision [10], [12], [16], [17]. TTME employs a pair of time information transmission channels, defined as *uplink* and *downlink*, to create four timestamps, which are then used to estimate the clock's offset. There is an important assumption here that if the clock offset and delay among the nodes are fixed during *uplink* and *downlink*, then the former can be directly determined using TTME timestamps.

Due to the clock frequency offset, the hardware clock offset is continuously increasing. As the time interval between TTME *uplink* and *downlink* increases, there is also a variable clock offset introduced to the fixed clock offset estimation, which leads the deterioration of time synchronization. On the other hand, the logic time notion of nodes is a discrete clock counter driven by a hardware clock. Its precision depends on the clock granularity; so in a limited time period, there is a fixed logic clock offset. In addition, in WSNs applications in complex environments, there is clock frequency jitter due to the

* Robot Technology Used for Special Environment Key Laboratory of Sichuan Province, Southwest University of Science and Technology, Mianyang, Si'chuan, CN; e-mail: {sfr_swust, rzw}@swust.edu.cn, lujing_017@live.cn, lili_ran@126.com

** College of Chemistry and Environment, Sichuan University of Science and Engineering, Zigong, Sichuan, CN; e-mail: tuoxg@cdut.edu.cn

Corresponding author: Xianguo Tuo

Recommended by Prof. Anmin Zhu
(DOI: 10.2316/Journal.206.2018.6.206-5160)

change of node voltage [18], temperature [19], and high propagation latency [20].

The TTME time synchronization algorithm can be improved to satisfy the synchronization requirements of quadrotor networks. These networks cover all the nodes in a single cluster-like star topology to avoid large transmission delays of control commands, while the central node (reference) may need to simultaneously respond to more than one TTME request of its neighbours. Additionally, the quadrotors execute multiple real-time tasks are controlled using real-time embedded operating systems (OS). Due to the scheduling mechanism, lower priority tasks may be blocked and even hung. The tasks pertaining to the flight control algorithm of quadrotor must have the highest priority, while time synchronization tasks have a lower priority.

In this paper, the details of continuous clock offset for hardware clocks are analysed, and fixed clock offset in TTME is discussed and its constraints are derived. Then, a quick TTME protocol with time-out restraining time synchronization for quadrotor networks is proposed.

2. System Model

The quadrotors are defined as wireless nodes and employ local clock sources for logic operations. Multiple quadrotors compose a wireless quadrotor network, which we consider as a WSN and discuss its synchronization problem. Figure 1 shows a wireless quadrotor network, where the quadrotors exchange time information and estimate the relative clock offset to formulate a consistent logic time notion. The network's model is defined as a graph $G=(V,E)$, where the nodes' subset is defined as $V=\{1,\ldots,n\}$ and the bidirectional communication link (edge) subset is defined as $E\subseteq(V\times V)$. An arbitrary node i has a neighbourhood nodes' subset $\mathcal{N}_i=\{i,j\}\in E$ and can only communicate with these nodes. The distance of arbitrary nodes $\{i,j\}\in V$ is defined as the number of edges on the shortest path between these two nodes. The *diameter* is defined as the maximum *distance* of any two nodes in G. The behaviour of wireless quadrotor networks depends on the speed of information transmission; so, their topology should minimize the number of multiple hops and the

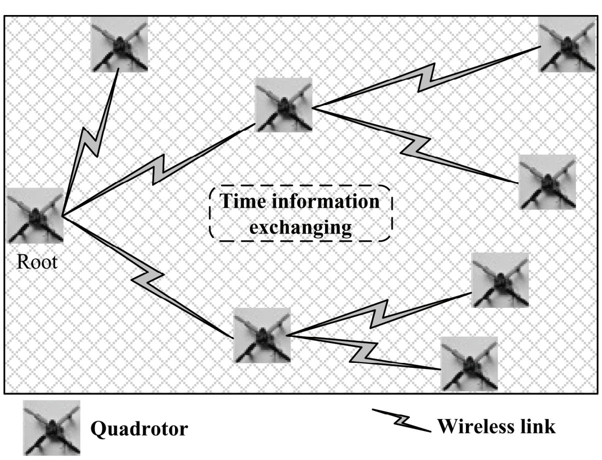

Figure 1. A wireless quadrotor network.

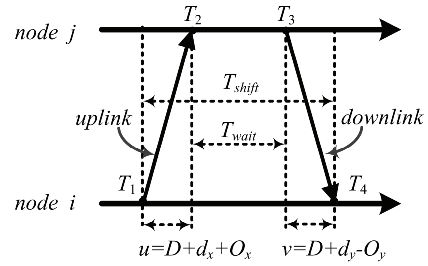

Figure 2. Two-way time message exchange between nodes i and j. There is a non-negligible difference between *uplink* clock offset O_x and *downlink* clock offset O_y.

diameter, so that the control and coordination commands reach their destination as quickly as possible.

To simplify the description, a TTME similar to TPSN will be referred to as traditional TTME, the details of which are shown in Fig. 2. Node i needs to be synchronized, while j is the reference node. TTME is performed in two steps. In the first step, i sends a short message to j; this message contains only the identity numbers of nodes i and j. Once the message is sent successfully, i will create a local logic time stamp T_1, while j creates the local logic time stamp T_2 once the message is received. This step is defined as the *uplink*.

The reference node j responds to the TTME at T_3 and sends another short message containing time stamps T_2 and T_3 to i, which in turn records the time of the message arrival as time stamp T_4. The second step aims to establish time stamps T_3 and T_4 and is referred to as the *downlink*. Then, the traditional TTME clock offset estimation $\hat{O}$ is given by:

$$\hat{O}=((T_2-T_1)-(T_4-T_3))/2 \tag{1}$$

The clock offset estimate obtained using (1) is rough. As in [17], we defined u as the duration time for TTME uplink, *i.e.*, $u\triangleq T_2-T_1=D+d_x+O_x$, while v is the duration time for TTME *uplink*, *i.e.*, $v\triangleq T_4-T_3=D+d_y-O_y$. D is defined as the fixed transmission delay between i and j; d_x and d_y are the variable transmission delays, while O_x and O_y are the clock offsets of i relative to j, where $O_x=O_y$. Then, the clock offset estimation $\hat{O}_x$ is given by (2):

$$O_x=\hat{O}+(d_x-d_y)/2 \tag{2}$$

In this estimation model, the estimation error is $(d_x-d_y)/2$. d_x is not equal to d_y and special probability distribution functions are employed to estimate them, *e.g.*, the exponential or Gaussian delay models. Noh *et al.* [10] and Chen *et al.* [11] employ maximum likelihood estimators to minimize this estimate error. Obviously the TTME clock offset estimation model of (2) is more accurate than (1), assuming other error sources are ignored. In our application's design, we found that the response speed of j influences the clock estimation precision. In particular, the quadrotors have multiple tasks and operate using real-time embedded OS. Real-time flight control tasks are more frequent and must be responded to quickly. The time synchronization tasks may be delayed until all other higher-priority tasks are finished.

To analyse the adverse effect of the TTME response speed, we introduce T_{shift} and T_{wait}. As Fig. 2 shows, T_{shift} is the time interval between T_1 and T_4 on node i, *i.e.*, the duration time of TTME, $T_{shift} \triangleq T_4 - T_1$. T_{wait} is the time interval between T_2 and T_3 on node j, which is the response delay of TTME, $T_{wait} \triangleq T_3 - T_2$. T_{wait} is the main constraint of TTME response speed and is a variable latency resulting from software delays, such as the interrupt response delay, priority queuing delay, or software blocking latencies. Furthermore, $T_{shift} = T_{wait} + u + v$. d_x and d_y can be described as random variables with exponential distribution. As we assume that T_{shift} is a random variable with exponential distribution, its mean is unknown. We use T_{shift} to describe the time cost of TTME.

As will be discussed in Section 3, the clock offset is increasing continuously; so there is clock offset O_{ij} for the TTME *uplink* and *downlink* among nodes ascribed to the clock frequency offset, *i.e.*, $O_y \triangleq O_x + O_{ij}, O_{ij} \geq 0$. O_x and O_y are the relative clock offsets for node i during TTME. Then, we rewrite the TTME clock offset estimation model as:

$$O_x = \hat{O}_x + (d_x - d_y)/2 + O_{ij}/2 \tag{3}$$

The $\hat{O}_x$ in (3) is the estimate of O_x; the former is used in TTME time synchronization to correct the local logic time rather than the latter. The estimation errors are mainly caused by d_x, d_y, and O_{ij}. Additionally, the O_{ij} is not equal to zero. Our work discusses the clock offset estimation errors caused by O_{ij} to reduce these errors.

We try to minimize the estimation error caused by O_{ij} and provide an accurate time synchronization approach for complex WSN applications that have multiple tasks, high propagation latency, and large response delays. The proposed time synchronization algorithm employs a spanning tree approach to maintain time synchronization. The root node is a global clock reference, while MAC layer timestamping [13] is employed to create time stamps. First, the root node initiates TTME to force its child nodes to synchronize with it. Then, the synchronized nodes function as reference nodes and synchronize their own child nodes until all nodes along the spanning tree have been synchronized. We use the clock offset estimation to perform linear regression and obtain the clock skew [9], [13]. The clock skew estimate error depends on the precision of the clock offset estimation; so in the following sections, we focus on accurate TTME clock offset estimation.

3. Preliminaries

In this section, the details of the hardware clock's continuous offset, which is caused by clock drift, are discussed, and the constraints for fixed clock offset TTME are deduced.

3.1 Continuous Clock Offset for Hardware Clocks

Clock sources are usually driven by rough crystal oscillators. The max frequency offset $a_{\max}$ could be up to dozens of ppms, even hundreds. If the clock frequency offset is relatively stable in a limited period, then the clock's offset rate of increase is fixed; thus, the max clock offset is calculated.

Assuming that the frequency of nominal clock source r is f Hz, where f is constant. The frequency offset of the arbitrary clock source i is a_i ppm. We define $\beta = 10^6$. The frequency of clock source i can be rewritten as f_i in (4). The exact values of a_i and f_i are unknown. We also let $T_{tick} = 1/f$ be the timing granularity of node i. Then,

$$f_i = f(1 + a_i/\beta) \tag{4}$$

This is the ideal clock period for a logic time counter but not the real period of clock i. We set the angular rate $w_i = 2\pi f_i$ and the real-time phase as $\varphi_i(t) = w_i t$. Then, the real-time phase difference $\varphi_r(t)$ between i and r can be written as (5).

$$\varphi_r(t) = \varphi_f(t) - \varphi_i(t) = 2\pi a_i f t/\beta \tag{5}$$

$$\gamma_r(t) = \varphi_r(t)/2\pi = a_i f t/\beta \tag{6}$$

$$O_r(t) = \gamma_r(t) T_{tick} = a_i t/\beta \tag{7}$$

$\varphi_f(t)$ is the phase of r at t, while $\varphi_i(t)$ is the phase of i at t. $\gamma_r(t)$ in (6) is the counts of differences for i at t *and* compares to r. $O_r(t)$ in (7) is the relative clock offset model of node i. The reference clock is r, so $O_r(t)$ is a continuous clock offset for the hardware clock, which is increasing continuously. $1 + a_i/\beta$ is the increasing speed of clock i due to the frequency offset. Therefore, there is a variable clock offset for TTME, which means that the estimate errors of clock offset $\hat{O}_x$ in (1) are unavoidable.

3.2 Fixed Clock Offset for Logic Time

The fixed clock offset does not actually occur in the hardware clock, but rather in the logic time calculation. Nodes employ a counter or timer to set up the logic time; this time perception is not continuous but rather discrete time, which increases by integer multiples of granularity T_{tick}. We define L_i as the logic time of node i, N_i as the count for clock pulse of i at t, and T_i as the actual period of clock i. Then (8) and (9) can be derived by (4) and (5).

$$L_i(T) = N_i T_{tick} \tag{8}$$

$$N_i = \lfloor t/T_i \rfloor = \lfloor (1 + a_i/\beta) f t \rfloor \tag{9}$$

The time granularity of logic time L_i is T_{tick}, but not the period, as $T_i = 1/f_i$ and $T_{tick} \neq T_i$, while N_i is the rounded down value of t/T_i. The logic time L_i is a discrete representation for the real-time t. By rewriting (6) and (7), we have the logic time offset of nodes i and j in (10) and (11).

$$\gamma_L(t) = \lfloor a_i f t/\beta \rfloor \tag{10}$$

$$O_L(t) = L_i(t) - L_j(t) = T_{tick} \gamma_L(t) \tag{11}$$

where $\gamma_L(t) \in N$ is the number of cycles difference between nodes, while $O_L(t)$ is the logic time offset of nodes i and j. The logic time offset is different from the hardware clock offset. As (10) and (11) show, t and a_i are variable. For a

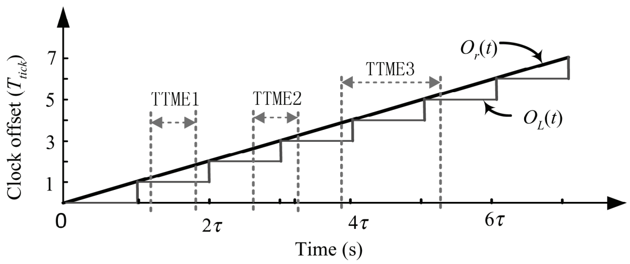

Figure 3. The hardware clock offset and logic clock offset. τ is the time cost that clock offset increases by a single T_{tick}. If we set $O_L(t) = T_{tick}$ in (9), then $\tau = t = \beta/\alpha_{\max} f$.

fixed t, the larger clock frequency offset leads to larger clock offset. If $\alpha_i < \alpha_j$ then:

$$\lim_{t\to\infty} O_L(t, \alpha_i) \ll \lim_{t\to\infty} O_L(t, \alpha_j) \tag{12}$$

In other words, in a bounded time t_{fixed}:

$$\lim_{a_i \to a_{\max}} O_L(t_{fixed}) = T_{tick} a_{\max} f t_{fixed}/\beta \tag{13}$$

$$\lim_{a_i \to 0} O_L(t_{fixed}) = 0 \tag{14}$$

As t increases, the hardware clock offset $O_r(t)$ is increasing continuously. The logic time offset $O_L(t)$ is gradually increasing as shown in Fig. 3. When $\gamma_L(t)$ is non-integer, $O_L(t)$ is a fixed value. The TTME clock offset estimation is $O_L(t)$ but not $O_r(t)$.

Assuming that the increment of $O_L(t)$ is not larger than 1 T_{tick}, then there is a fixed logic clock offset $O_L(t)$. The logic time offset is a discrete integer multiple time granularity. So, at the interval between $n\tau$ and $(n+1)\tau$ ($n \in N_+$), the increment of $O_L(t)$ is either 0 or 1 T_{tick}. As shown in Fig. 3, TTME1 and TTME2 have the same time cost T_{shift} and $T_{shift} < \tau$, while the clock offset increment is either 0 or 1 T_{tick}. This is due to the initialization time of TTME and the clock phase offset. TTME3 has $T_{shift} > \tau$, so its clock offset increment is 2 T_{tick}. Therefore, if the processing of TTME is fast enough, then the increment of $O_L(t)$ is not larger than 1 T_{tick}. Thus, a fixed clock offset exists for TTME.

Let us consider the worst case, where the clock frequency offset is $\alpha_{\max}$. The requirement that TTME meets the above assumption means that TTME must be finished at time τ, where τ is given by:

$$\tau \le \beta/\alpha_{\max} f \tag{15}$$

So if $T_{shift} \le \tau$, the TTME will meet the assumption. The probabilities of the clock offset increment being either 0 or1 T_{tick} are exactly the same. If $T_{shift} \le \tau$ then the probability that the increment of $O_L(t)$ is not larger than 1 T_{tick} is 1.

4. Quick TTME

Based on the above analysis, the estimate error of (1) is given by O_{ij}. We define the time stamp T_1 of TTME as the time origin, *i.e.*, the point in time when $t = 0$. For the logic time perception, the clock offset increment is given by O_L but not O_r, so $O_{ij} = O_L$. Based on the assumption, the clock offset model is given by:

$$O_y = \begin{cases} O_x & \tau \le \beta/\alpha_{\max} f \\ O_x + O_l(\tau) & \tau > \beta/\alpha_{\max} f \end{cases} \tag{16}$$

As (11), (15) and (16) show, for a smaller τ, there is a greater probability that O_x is equal to O_y in (16) and the smaller estimate error for $\hat{O}_x$ is O_{ij}.

This paper proposes a quick TTME clock offset estimation protocol. If the TTME responses are as fast as possible, this will guarantee T_{wait} and T_{shift} small enough and generate a greater probability to maintain a fixed clock offset for time stamps.

Algorithm 1. *Quick TTME with timeout constraint.*

■ ***Initialization***
1. *set* $\rho\beta/\alpha_{\max} f \to T_{limit}$
■ ***TTME***
2. *i sending,* $T_1 \leftarrow$ *PACKET transmission done.*
3. *j receiving,* $T_2 \leftarrow$*PACKET reception done.*
4. *j sending,* $T_3 \leftarrow$*PACKET transmission done.*
5. *i receiving,* $T_4 \leftarrow$*PACKET reception done.*
■ ***Timeout detection***
6. $T_{wait} \leftarrow (T_3 - T_2)$.
7. *if* $T_{wait} < T_{limit}$ *then jump to 9.*
8. *else jump to 2, and try again.*
9. *save timestamps* $T_1, T_2, T_3,$ *and* T_4.

The quick TTME is proposed to keep the clock offset fixed for the time stamps T_1, T_2, T_3, T_4. Following the restriction of T_{shift} in (15), the timeout threshold T_{limit} of quick TTME is given by $\rho\beta/\alpha_{\max} f$ $(0 < \rho < 1)$ (Algorithm 1, line 1). T_{limit} is employed to restrict the time cost of TTME. We employ the "packet transmission done" interrupt to create time tamps T_1 and T_3 at the sender side (Algorithm 1, lines 2,4), and the "packet received" interrupt to create timestamps T_2 and T_4 at the receiver side (Algorithm 1, lines 3,5).

T_{wait} is random variable. It is mainly caused by the interrupt handling delay, software blocking latencies, and wireless channel access delay for *downlink*. A poor wireless communication environment and software task priority blocking cause the waiting time to increase unpredictably. Quick TTME uses T_{limit} to restrict the T_{wait} (Algorithm 1, lines 6,7). If T_{wait} is larger than T_{limit}, then the algorithm judges that there was a TTME timeout and considers the TTME invalid. In this case, another TTME will be initialized immediately (Algorithm 1, line 8).

The constant ρ needs to be set as an appropriate value (Algorithm 1, line 1). A smaller ρ leads to a smaller T_{limit} and a tighter restriction for timeout. There will be a smaller T_{shift} for a reliable TTME so that the TTME has a greater probability of fixed clock offset for time stamps and higher precision of clock offset estimation for time synchronization. Conversely, a larger ρ means corresponds to a loose constraint for TTME, as the probability for the fixed clock offset time stamps is smaller and there will be rough clock offset estimation for TTME. In addition, ρ does not only determine the precision of clock offset estimation but is also related to the efficiency of time synchronization. A tight timeout restriction leads to a greater probability of TTME retry (Algorithm 1, line 8), potentially even leading the TTME into an infinite loop (Algorithm 1, line 7). Therefore, a maximum retries' number should be employed to avoid this case. Thus, an exact ρ will balance the accuracy and convergence speed of the time synchronization algorithm. A big T_{shift} makes the TTME of (1) invalid, while the quick TTME holds the T_{shift} and avoids the additional error O_{ij}.

5. Error and Efficiency Analysis

In this section, we discuss the clock offset estimation error and time synchronization efficiency under a random exponentially distributed variable delay. Let us assume that T_{shift} (the response delay) is a random variable with exponential distribution which has an unknown mean λ. Its probability density function (PDF) is:

$$fT_{shift}(k) = e^{-k/\lambda}/\lambda \tag{17}$$

where k is the sample value of T_{shift}. We define $Z \triangleq \{the\ TTME\ has\ a\ minimum\ clock\ offset\ O_{ij}\}$. As (15) shows, if $0 < k \le \rho\beta/\alpha_{\max}f$, then O_{ij} is not larger than T_{tick} and $Z = TRUE$. The probability of $Z = TRUE$ is written as:

$$\begin{aligned} P(Z) &= \int_0^{\rho\beta/\alpha_{\max}f} fT_{shift}(k)dk \\ &= 1 - e^{-\rho\beta/\alpha_{\max}f\lambda}\ \ (0 < k \le \beta/\alpha_{\max}f) \end{aligned} \tag{18}$$

Traditional TTME uses the time stamp observations to estimate the clock offset directly. It has the same probability as $P(Z)$ to introduce a large O_{ij} in the clock estimate error. If $\rho\beta/\alpha_{\max}f < k$, O_{ij} will be a couple of T_{tick} and even more. A larger O_{ij} leads to a larger estimate error for $\hat{O}_x$ in (3). The quick TTME employs a timeout constraint, which means that once $\rho\beta/\alpha_{\max}f < k$, the TTME observations will be discarded and another TTME will be started. O_{ij} could never be larger than a single T_{tick} for a quick TTME clock offset estimation. So, $P(Z) = 1$ for quick TTME, and the max O_{ij} for the $\hat{O}_x$ is a single T_{tick}.

As discussed in Section IV, when there is a tight constant ρ or a large λ, the quick TTME may be trapped into a loop. The efficiency model of quick TTME is thus deduced by ρ and λ. It is defined as a reference to help the time synchronization algorithm balance synchronization precision and convergence rate. $E(Z,R)$ is defined as the efficiency function of quick TTME, where R is the expectation of retry times. Then, $E(Z,R)$ is written as:

$$\begin{aligned} E(Z,R) &= \textstyle\sum_{R=1}^{+\infty} R \times P(Z) \times (1-P(Z))^{R-1} \\ &= P(Z)\textstyle\sum_{k=1}^{+\infty} R \times (1-P(Z))^{R-1} \end{aligned} \tag{19}$$

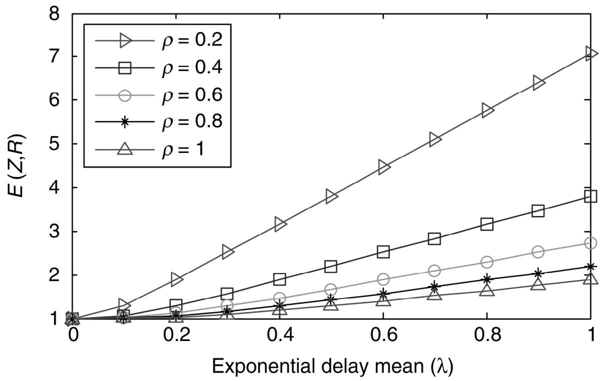

Figure 4. Plot of $E(Z,R)$. $a_{\max} = 40$ ppm.

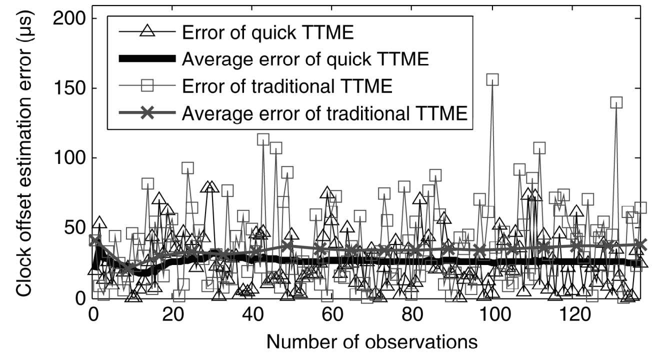

Figure 5. The clock offset estimation errors.

Now, as

$$\textstyle\sum_{R=1}^{+\infty} R \times (1-P(Z))^{R-1} = 1/P^2(Z) \tag{20}$$

$E(Z,R)$ can be rewritten as:

$$E(Z,R) = 1/P(Z) = 1/(1 - e^{-\rho\beta/\alpha_{\max}f\lambda}) \tag{21}$$

In (21), f is known, while ρ, a_i, and λ are unknown bounded constants. ρ is set by quick TTME. If the quick TTME uses an exact ρ based on (21), it will meet the requirements of high efficiency and high precision for time synchronization. Equation (21) is also the convergence model for quick TTME. As Fig. 4 shows, if $a_{\max}$ is constant, a tighter ρ leads to a larger retry time, and so it is the larger respond delay.

6. Simulation Results

A simulation platform based on the True-Time 2.0 toolbox was established to compare quick TTME and traditional TTME experimentally. We set the hardware clock frequency as 32.768 kHz, a_i had a variable value and $a_{\max} = 40$ ppm, so $-40 \le a_i \le 40$. The max clock drift was set to 0.2 ppm We compared the statistical properties of both traditional TTME and the quick TTME in TPSN.

Traditional TTME has a larger average estimate error, particularly when the variable processing latency T_{wait} is increasing. Let us assume that T_{wait} is a random variable with exponential distribution, while $\lambda = 1$. We set $\rho = 0.12$. Figure 5 shows that for traditional TTME the

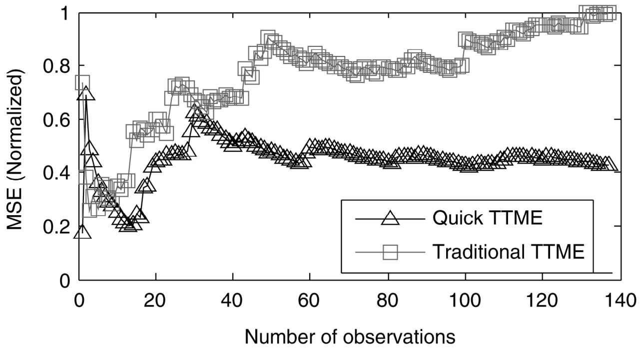

Figure 6. MSE of clock offset estimation.

Table 1
Probability Statistics of Clock Offset Estimation

Clock Offset Estimation Errors	Probability (%)	
	Traditional TTME	Quick TTME
≤15 µs	23.4	29.2
≤30 µs	46	59.1
≤45 µs	64.2	83.2

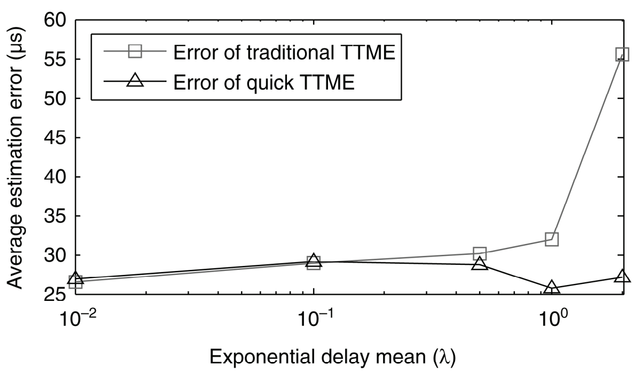

Figure 7. Clock offset estimation stability.

max clock offset estimation error is up to 156 µs, while the average estimation error is 38 µs. For the quick TTME the corresponding values were 76 µs, and 26 µs, respectively. Figure 6 shows that the normalized mean square error (MSE) of quick TTME is smaller and smoother than traditional TTME.

The probability statistics of traditional TTME and quick TTME are shown in Table 1. In our experiment, for $T_{tick} = 30.518\,\mu s$, the quick TTME clock offset estimation error has a 59.1% probability of being smaller than T_{tick}, which is larger than the corresponding value of traditional TTME by 13.1 percent.

As T_{wait} increases, there are more disadvantages for time synchronization. Figure 7 shows the stability of traditional TTME and quick TTME. As λ increases, the traditional TTME average clock offset estimate errors are increasing with T_{wait}. The quick TTME employs timeout limitation to maintain a fixed clock offset, which stabilizes clock offset estimation.

7. Conclusion

In this paper, we discussed quick TTME clock offset estimation for time synchronization in wireless quadrotor networks. To mitigate the deterioration of time synchronization due to continuous clock offset, the constraint for fixed clock offset TTME is derived. Based on the timeout constraint, quick TTME ensures a fixed clock offset for TTME observations and avoids the estimate errors caused by the continuous increase of the clock offset. Quick TTME can achieve reliable clock offset estimation for wireless quadrotor networks in complex environments. It is robust and accurate for clock offset estimation.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61601383, 41227802). This work was also supported and funded by Longshan Academic Talent Research Supporting Program of Southwest University of Science and Technology (17LZX650) and Fund of Robot Technology Used for Special Environment Key Laboratory of Sichuan Province.

References

[1] N.E. Zoghby, V. Loscri, E. Natalizio, and V. Cherfaoui (eds.), *Wireless sensor and robot networks: From topology control to communication aspects*, 8 (Singapore: World Scientific Publishing Company, 2014), 168–201.

[2] C. Li, Z. Qu, and E. Pollak, Cooperative attitude synchronization for rigid-body spacecraft via varying communication topology, *International Journal of Robotics & Automation*, *26*(1), 2011, 110–119.

[3] K. Ohara and T. Tanikawa, Wireless time synchronization module for ubiquitous robot system, *2013 10th International Conf. on Ubiquitous Robots and Ambient Intelligence (URAI)*, Jeju, 2013, 375–377.

[4] S. Wei Wei, S. Cong, and S. Jiang, Synchronization control of a parallel manipulator with redundant actuation in the task space, *International Journal of Robotics & Automation*, *26*(4), 2011, 432–440.

[5] T.S. Kim, Y.H. Lee, J.K. Park, T.Y. Kuc, *et al.*, A time synchronized multi-robot coverage algorithm for unstructured environment, *2014 14th International Conf. on Control, Automation and Systems (ICCAS 2014)*, Seoul, 2014, 503–508.

[6] D. Sun, C. Wang, W. Shang, and G. Feng, A synchronization approach to trajectory tracking of multiple mobile robots while maintaining time-varying formations, *IEEE Transactions on Robotics*, *25*(5), 2009, 1074–1086.

[7] F. Rosales-Hernández, M. Velasco-Villa, R. Castro-Linares, B. del Muro-Cuéllar, *et al.*, Synchronization strategy for differentially driven mobile robots: Discrete-time approach, *International Journal of Robotics & Automation*, *30*(1), 2015, 50–59.

[8] I.M.H. Sanhoury, S.H.M. Amin, and A.R Husain, Synchronizing multi-robots in switching between different formations tasks while tracking a line, *Trends in Intelligent Robotics, Automation, and Manufacturing*, Berlin, 2012, 28–36.

[9] S. Ganeriwal, R. Kumar, and M.B. Srivastava, Timing-sync protocol for sensor networks, *SenSys '03 Proc. of the 1st International Conf. on Embedded Networked Sensor Systems*, Los Angeles, CA, 2004, 138–149.

[10] K.L. Noh, Q.M. Chaudhari, E. Serpedin, and B.W. Suter, Novel clock phase offset and skew estimation using two-way timing message exchanges for wireless sensor networks, *IEEE Transactions on Communications*, *55*(4), 2007, 766–777.

[11] J. Chen, Q. Yu, Y. Zhang, H.H. Chen, *et al.*, Feedback-based clock synchronization in wireless sensor networks: A

control theoretic approach, *IEEE Transactions on Vehicular Technology*, *59*(6), 2010, 2963–2973.

[12] H. Wang, H. Zeng, and P. Wang, Linear estimation of clock frequency offset for time synchronization based on overhearing in wireless sensor networks, *IEEE Communications Letters*, *20*(2), 2016, 288–291.

[13] J. Elson, L. Girod, and D. Estrin, Fine-grained network time synchronization using reference broadcasts, *OSDI '02: Proc. of the 5th Symposium on Operating Systems Design and Implementation*, Boston, Massachusetts, 2002, 147–163.

[14] K.S. Yildirim, A. Kantarci, Time synchronization based on slow-flooding in wireless sensor networks, *IEEE Transactions on Parallel and Distributed Systems*, *25*(1), 2014, 244–253.

[15] C. Lenzen, P. Sommer, and R. Wattenhofer, PulseSync: An efficient and scalable clock synchronization protocol, *IEEE/ACM Transactions on Networking*, *23*(3), 2015, 717–727.

[16] Q.M. Chaudhari, E. Serpedin, and K. Qaraqe, Some improved and generalized estimation schemes for clock synchronization of listening nodes in wireless sensor networks, *IEEE Transactions on Communications*, *58*(1), 2010, 63–67.

[17] D.R. Jeske, On maximum-likelihood estimation of clock offset, *IEEE Transactions on Communications*, *53*(1), 2005, 53–54.

[18] Z. Li, W. Chen, M. Li, and J. Lei, Incorporating energy heterogeneity into sensor network time synchronization, *IEEE Transactions on Parallel and Distributed Systems*, *26*(1), 2015, 163–173.

[19] M. Xu, W. Xu, T. Han, and Z. Lin, Energy-efficient time synchronization in wireless sensor networks via temperature-aware compensation, *ACM Transactions on Sensor Networks*, *12*(2), 2016, Article 12.

[20] J.S. Pettyjohn, D.R. Jeske, and J. Li, Least squares-based estimation of relative clock offset and frequency in sensor networks with high latency, *IEEE Transactions on Communications*, *58*(12), 2010, 3613–3620.

Biographies

Fanrong Shi received B.E. in Communication Engineering and M.E. in Communication and Information System from the Southwest University of Science and Technology (SWUST), Mianyang, China. He is currently pursuing the Ph.D. degree in Control Science and Engineering at SWUST. His research interests include Time Synchronization, Distributed Sensing and Location in Wireless Sensor Networks, and Wireless Communication System. He is a member of the Robot Technology Used for Special Environment Key Laboratory of Sichuan Province, Southwest University of Science and Technology. Currently, he is a lecturer in the Department of School of Information Engineering, SWUST.

Xianguo Tuo received the B.E., M.E., and Ph.D. degrees in Nuclear Geophysics, Environmental Radiation Protection, Applied Nuclear Technology in Geophysics from the Chengdu University of Technology in Chengdu, China, in 1988, 1993, 2001, respectively. From 2006 to 2007, he worked as visiting scholar at the School of Bioscience at University of Nottingham, UK. Since 2002, he is Professor with College of Information and Engineering, Chengdu University of Technology, China. Since 2012, he becomes Professor with the School of National Defense Science and Technology, Southwest University of Science and Technology in Mianyang, Sichuan, China. Currently, he is the director of the Artificial Intelligence Key Laboratory of Zigong, Sichuan. He received The Excellent Youth Scientists Award by Ministry of Science & Technology in 2006. Currently, his research interests are in detection of radiation, seismic exploration, and specialized robots.

Jing Lu is now a Ph.D. student from major of Control Science and Engineering, in Southwest University of Science and Technology. In 2014, he finished his graduate project of human–robot interaction subject in LIRMM (Montpellier Laboratory of Informatics, Robotics and Microelectronics), Montpellier, and worked in Aldebaran, Paris, as an intern student, working at a humanoid robot project. In the same year, he received his M.S degree in Electrical Engineering from Montpellier University Graduate Engineering School. Currently, he is working on physical robots' controls. In particular, he focuses on the development of the exploration robot in special environment, especially the USV and sphere robot.

Zhenwen Ren received M.E. in Communication and Information System from the Southwest University of Science and Technology (SWUST). He is currently pursuing the Ph.D. degree in control science and engineering at Nanjing University of Science and Technology, Nanjing, China. His research interests include Machine Learning, Pattern Recognition, Computer Vision, and Compressed Perception. Currently, he is Lecturer with the Department of School of National Defence Science and Technology, SWUST.

Lili Ran received B.E. in Electric Engineering and M.E. in Rail Transit and Electrical Automation from the Southwest Jiaotong University. His research interests include Numerical Simulation Distributed Sensing and System Modelling. She is a member of the Robot Technology Used for Special Environment Key Laboratory of Sichuan Province, Southwest University of Science and Technology. Currently, she is a lecturer in the Department of School of Information Engineering, Southwest University of Science and Technology.