Simulation and Performance Analysis of the IEEE1588 PTP with Kalman Filtering in Multi-hop Wireless Sensor Networks

Baoqiang Lv¹, Yiwen Huang¹, Taihua Li¹*, Xuewu Dai¹, Muxi He¹, Wuxiong Zhang², and Yang Yang²

1. School of Electronic and Information Engineering, Southwest University, Chongqing, 400715, China
2. Shanghai Research Center for Wireless Communications, ShanghaiTech University, Shanghai, 201210, China

*Corresponding author, Email: {jietelv8, piaoyao, catalyst, daixuewu, hemuxi}@swu.edu.cn, {wuxiong.zhang, yang.yang}@wico.sh

Abstract—As the wireless sensor networks (WSNs) find wider and wider applications, time synchronization has emerged as a key technology to improve network performance and enable WSNs in time-sensitive applications. The recently proposed IEEE 1588 Precision Time Synchronization Protocol (PTP) has shown its capability for the wired Ethernet, but its performance in multi-hop WSNs is still an open question. This is because of the non-negligible asymmetric delays caused by the wireless media access control protocol, low-cost crystal oscillators and time-stamping uncertainties due to limited resources in WSNs nodes. This paper studies the performance of the IEEE 1588 in a multi-hop WSNs by state-space modeling and realistic simulation. Furthermore, the Kalman filter is introduced to improve the offset and skew estimation. The realistic simulator was developed on the OMNeT++ discrete event simulation platform and the simulation results show that the proposed Kalman filtering improves the performance of time synchronization in multi-hop WSNs in terms of offset and skew estimation.

Index Terms—Time Synchronization; IEEE 1588 PTP; WSNs; Kalman Filter; OMNeT++

I. INTRODUCTION

Many applications of Wireless Sensor Networks (WSNs) need to maintain time synchronization, such as, event detection, localization, data fusion, Time Division Multiple Access (TDMA) scheduling, remote health monitoring. Generally, the node clocks of WSNs are implemented through crystal oscillator and interrupt mechanism. Furthermore, the precision of the WSN’s clock is greatly affected by manufacturing imperfections, temperature variation and the interrupt latency [1]. Moreover, the wireless sensors whose resources are limited is cheap, and are significantly constrained in computing resources, and computational capacity [2]. Therefore, one of the main challenges in WSNs time synchronization is how to achieve high-precision time synchronization while requiring less overhead and consuming less communication bandwidth and CPU sources.

Time synchronization has emerged as a hot research field in distributed systems. The Network Time Protocol (NTP) [3] has been widely used in the Internet; however, in the ideal condition, the accuracy of NTP is within few milliseconds in WSNs, which is not enough for the application requires high time accuracy, such as TDMA scheduling and some industrial wireless networks. In recent years, more and more researchers pay attention to time synchronization protocols and lots of time synchronization protocols have been proposed, such as Flooding Time Synchronization Protocol (FTSP) [4], Timing-sync protocol for sensor networks (TPSN) [5], Reference Broadcast Synchronization (RBS) [6], Delay Measurement Time Synchronization (DMTS) [7], however, these protocols are used in wireless networks with nearly zero propagation delay between nodes, and failed to achieve optimization on energy efficiency and accuracy[8]. Therefore, the new design of time synchronization protocol in WSNs is needed. As a recently proposed time synchronization standard for Ethernet in industrial data acquisition and control applications, the IEEE 1588 Precision Time Synchronization Protocol (PTP) has shown it capability to achieve high synchronization accuracy. The PTP standard and associated techniques are thoroughly examined by J. C. Eidson [9]. The PTP Time synchronization indeed is a process that estimates the local drifting clock’s offset and skews by exchanging timing information so that the local clock tracks the reference time as accurately as possible. The PTP time synchronization accuracy is claimed to be in microsecond range, which is much better than the time synchronization accuracy of NTP and FTSP, etc. However, although the PTP has shown it capability for wired Ethernet, its performance in multi-hop WSNs is still an open question due to the non-negligible asymmetric delays and time-stamping uncertainties caused by the wireless media access control protocol and limited resources in WSNs nodes. This motivates us study the IEEE 1588’s performance in multi-hop WSNs.

However, in real networks, in order to save the energy on radio transmission, multi-hop communications are widely used in wireless sensor networks. In multi-hop WSNs, a node may work as combination of both sensor node and relay node. Furthermore, large populations of
sensor nodes will collaborate in order to complete the measuring data at the same time, data gathering, data fusion and localization. In the wireless sensor network with large scale of energy limited nodes, multi hop time synchronization is necessarily applied. Therefore, more and more researchers pay attention to time synchronization in multi-hop WSNs and lots of multi-hop time synchronization algorithms have been proposed over the years. K. Y. Cheng et al proposed Pairwise Broadcast Synchronization (PBS) in multi-hop WSNs [10]. V. Kaseva et al proposed a delay-based time synchronization protocol that is based on calculating delays as packets traverse from node to node for application packets in multi-hop WSNs [11]. A multi-hop time synchronization scheme is proposed for underwater acoustic networks [12]. Haitao Xiao et al proposed a multi-hop low cost time synchronization algorithm for wireless sensor network in bridge health diagnosis system with the purpose of reducing energy consumption and lengthening whole WSN‘ life [13]. Zhehan Ding et al proposed an improvement of energy efficient multi-hop time synchronization algorithm in wireless sensor network, which consumes less energy than RBS and can decrease overhead [14]. Compare to most existing time synchronization protocols, one of promising features of the PTP is its less demanding on communication bandwidth and computation costs, while keeping the similar level of time accuracy [15]. With small communication bandwidth, the PTP occupies relatively fewer network resources. The PTP has simple operation, small computational and memory requirement, lower hardware requirements, and is easy to maintain. As a result, a well optimization on energy efficiency and accuracy can be achieved by using the PTP in multi-hop WSNs.

In this paper, the IEEE 1588 PTP is adopted in multi-hop WSNs, and the state-space modeling is used to describe the process of time synchronization. The clock offset between master clock and slave clock is estimated and adjusted by IEEE 1588 PTP, but the accuracy of estimated value for skew and offset is not high enough. So an open source simulator for IEEE1588 time synchronization over 802.11 networks has been proposed to evaluate PTP’s performance by realistic simulation [16]. Ling Ye et al proposed the Kalman filter for precision time synchronization in wireless sensor networks to improve PTP’s performance [17]. Moon et al proposed a linear programming-based algorithm to estimate the clock skew [18]. Xu Bao proposed time synchronization algorithm for WSNs based on cluster [19]. Since the Kalman filter can effectively eliminate noise to avoid influence of the measurement values, it has been intensively studied in time synchronization [20]. In this paper, we study the application of Kalman filter to the PTP’s offset and skew estimation in multi-hop WSNs.

A simulator has been developed to evaluate the performance of the proposed IEEE 1588 PTP with Kalman filter for multi-hop WSNs. Due to its capability of simulating the real wireless channel, the OMNeT++ discrete event simulator (DES) is selected as the simulation platform. The OMNeT++ is an open source object-oriented modular discrete event network simulator. Its model consists of hierarchically nested modules; modules communicate with each other by exchanging timing messages, which can contain arbitrarily complex data structures. Our simulator is a further development of x-simulator [21] and new relay nodes are developed, which combine both the IEEE 1588 PTP master node and slave node into one relay node.

The rest of this paper is organized as follows: the clock model is described in section II. The PTP model is presented in Section III. Section IV describes the simulator development for multi-hop time synchronization. We introduced noises in IEEE 1588 offset calculation in section V. Kalman filtering for multi-hop time synchronization is proposed by section VI and section VII presents the performance evaluation.

II. CLOCK MODEL

In practical applications, it is impossible to obtain the crystal oscillator with exactly the same properties of clocks, which were generated by the different rate of counter. In order to understand the clock bias caused by instantaneous phase variations and frequency deviation of the crystal oscillator from its normal value, clock needs to be modeled.

A. Master Node’s Ideal Clock

Let t and M(t) represent the reference time (or termed as global time) and the clock of the master node at time t, respectively, it is common to assume the clock of the master node is accurate. That is:

\[ M(t) = t \] (1)

Therefore, synchronizing the slave clock to the master clock is equal to synchronizing to the reference time t.

B. Slave Node’s Drifting Clock

Let \( \theta(t) \) represent the slave node’s offset at reference time t, the offset \( \theta(t) \) is defined as the time difference between the master clock t and the slave clock C(t).

\[ \theta(t) = C(t) - t \] (2)

Skew, represented by \( \gamma(t) \), is the deviation of the local clock frequency from the reference clock frequency and can be expressed as:

\[ \gamma(t) = \frac{dC(t)}{dt} - 1 \] (3)

Skew is always expressed by the dimensionless unit of Parts Per Million (PPM), and \( dC(t)/dt \) is the clock’s change rate used to describe the local clock frequency. In our simulator, the slave clock is modeled by a “simple skew model” (SKM) as shown in reference [22]. In this model, the variations of both the offset and skew are regarded as random processes, and the dynamics of both offset and skew can be written as follows.

\[ \theta(k + 1) = \theta(k) + \gamma(k) \tau + \omega(k) \] (4)
\( \gamma(k+1) = \gamma(k) + \alpha_p(k) \) \hspace{1cm} (5)

where \( k \) represents the \( k \)-th physical clock update, \( \tau \) is a small time interval (referred to as clock update interval) determining how frequently the clock model is updated. \( \alpha_p(k) \) and \( \alpha_v(k) \) are two independent random processes representing the phase noise (referred to as offset noise) and frequency drifting (referred to as skew noise), respectively. Similar to the treatment in [23], \( \alpha_p(k) \) and \( \alpha_v(k) \) are considered as two uncorrelated white Gaussian random processes with variances \( \delta_p^2 \) and \( \delta_v^2 \) respectively.

III. MATHEMATICAL MODE OF PTP

In this section, the PTP is described as a state-space modeling through two measurement equations so that Kalman filtering is used to improve the performance of PTP time synchronization between the master node and slave node.

A. Clock Offset Estimation by PTP

A typical process of the PTP time synchronization between the master node and its slave nodes is based on a delay request-response mechanism, as demonstrated in Fig. 1. Four types of time-stamped packets are defined in the IEEE 1588 standards [15]. They are Sync packet, Follow_Up packet, Delay_Req packet and Delay_Resp packet. The Follow_UP packet is used to transport the timestamp of the preceding Sync packet if there are uncertainties on the master clock. In our simulator, we assume the master node has enough resources and is able to process the time-stamped packet quickly and there is no need to use the Follow_Up packet.

The time synchronization is carried out in a series of packet exchange procedures which is repeated periodically and each procedure of packet exchange involves a sequence of packet transmission. Let \( n \) denote the \( n \)-th procedure of packet exchange, at the beginning of the \( n \)-th time synchronization procedure, a Sync packet is sent from the master node to the slave node at time \( t_1 \). When the Sync packet is received by the slave, the receiving time \( t_2 \) provided by the slave's drifting clock is recorded. Let \( d_{ms} \) denote the propagation delay of a Sync packet traveling from the master node to the slave node, and let \( \theta \) denote the clock offset between the master node and the slave node at the \( n \)-th synchronization procedure. The following relationship holds:

\[ t_1(n) + \theta(n) + d_{ms}(n) = t_2(n) \] \hspace{1cm} (6)

Since equation (6) has two unknown variables, offset and master-slave propagation delay, one more equation is needed to measure the propagation delay so that the offset can be solved. Propagation delay measurement is initiated by the slave clock. Firstly, a Delay_Req packet from the slave node is sent to the master node at time \( t_3 \), and the master node receives the packet and records its receiving time \( t_4 \). A Delay_Resp packet embedding time \( t_3 \) from master node is then sent to the slave node as soon as possible. Let \( d_{sm} \) denote propagation delay of packet traveling from the slave node and master node. The following relationship holds:

\[ t_4(n) + \theta(n) + d_{sm}(n) = t_3(n) \] \hspace{1cm} (7)

The clock offset \( \theta \) between the master node and slave node can be regarded as an unknown quantity. Consequently, the offset can be calculated by equations (6) and (7):

\[ \theta(n) = \frac{(t_4(n) - t_1(n)) - (t_3(n) - t_2(n)) + d_{ms}(n) - d_{sm}(n)}{2} \] \hspace{1cm} (8)

where equation (8) is called clock offset measurement equation, which describes the relationship among the clock offset and the time stamps during the process of IEEE 1588 packet exchange. Under the condition of propagation delay symmetry, i.e., \( d_{ms}=d_{sm} \), the offset can be calculated as:

\[ \theta(n) = \frac{(t_2(n) - t_1(n)) - (t_3(n) - t_4(n))}{2} \] \hspace{1cm} (9)

For example, if the Master node starts a round of time synchronization by sending the Sync packet at \( t_1=0.1 \) second (measured at master accurate clock) and the Sync packet arrives at Slave1 at \( t_2=0.100015 \) second (measured at Slave1’s drifting clock), Slave1 node sends back a Delay_Req at \( t_3=0.100019 \) second (measured at Slave1’s drifting clock) and the Master node receives Delay_Resp at \( t_4=0.100029 \) second (measured at Master's accurate clock). Once the Delay_Resp arrives at Slave1, the offset between Slave1’s clock and Master’s clock can be calculated by equation (9). Once the first hop Master-Slave1 synchronization is done, Slave1 will work as a master in the second hop (Slave1-Slave2). A similar process is carried out between Slave1 and Slave2.

B. Clock Skew Estimation by PTP

Although the IEEE 1588 standard does not define the skew estimation, for a better synchronization between the slave clock and the master clock, it is good to estimate both the clock offset and skew. The measured value of clock offset is acquired by equation (8). Then, the value of the clock skew \( \gamma(n) \) can be acquired by sending consecutive Sync message. In our simulator, the clock offset \( \theta \) is assumed as constant in the process of PTP synchronization packet exchange. A simple measured
value of the clock skew $\gamma(n)$ can be obtained by the offset measured values:

$$\gamma(n) = \frac{\theta(n) - \theta(n-1)}{\Delta T}$$  \hspace{1cm} (10)

IV. SIMULATOR DEVELOPMENT FOR MULTI-HOP TIME SYNCHRONIZATION

In order to study the performance of IEEE 1588 PTP time synchronization in multi-hop WSNs, we developed a realistic simulator on the OMNeT++ platform. The proposed simulator is a further development of the x-simulator [21] and it supports three types of nodes, namely Master, SIM2 and Slave2. Fig. 2 illustrates a simple two-hop network topology for the purpose of demonstration. It is worth noting that node SIM2 works as a slave node in its first hop to Master on the left and as a master node in its second hop to Slave2 on the right.

As shown in Fig. 3, the main blocks of Master node are (1) an ideal clock representing the reference time $t$ and (2) a PtpM1 block implementing the functions of PTP master specified by the IEEE 1588 standard. Similarly, the main blocks of Slave2 node are: (1) a Drifting Clock that will be synchronized to the master clock $M(t)$; (2) a PtpS2 block implementing the functions of PTP slave specified by the IEEE 1588 standard. These two types of nodes are the standard master/slave nodes defined by the IEEE 1588 standard. They are implemented in a similar manner as the master/slave node in the x-simulator [21].

The unique feature of the proposed simulator is the SIM2 node which enables us to simulate multi-hop time synchronization. The SIM2 node integrates functions of both the PTP master and PTP slave into one node. As shown in Figure 3, the SIM2 node has three main blocks: (1) a Drifting Clock; (2) PtpS1 block is similar to the PtpS2 block in Slave2 node but works as a PTP slave in the first hop to the Master’s PtpM1 block; (3) PtpM2 block is the additional block working as a PTP master in the second hop to the Slave2’s PtpS2 block.

The process of time-stamped PTP packet exchange among these nodes and the time-stamping procedure within the SIM2 and Slave2 nodes are described next. In the following description, unless otherwise specified, packet represents the time-stamped PTP packet between two nodes and message represents the information exchange within a node.

A. PTP Packet Exchange in the First Hop

The process of time-stamped PTP packet exchange among the Master node and SIM2 node in the first hop and the time-stamping procedure within the SIM2 node are described as follows:

Firstly, the Master’s PtpM1 sends a Sync packet periodically at an interval $\Delta T$ to the PtpS1 in the SIM2 node of the first hop. The Sync contains the time $t$, which is the Sync’s sending time measured by the master clock.

The Sync first arrives at the input gate of the SIM2, and then received by the PtpS1. The PtpS1 records $t_1$ and instantly sends a Sync_Time_Req message back to its Drifting Clock for the time stamp of $t_2$. Once receiving the time request message Sync_Time_Req, the Drifting Clock retrieves the time of drifting clock as $t_2$ and then sends a Time_Res message containing $t_2$ back to the PtpS1. As a result, the PtpS1 obtains the receiving time stamp $t_2$ of the PTP packet Sync.

Once the PtpS1 has obtained and recorded the time stamp $t_2$, it waits for a short period of time delay (to simulate the delays caused by clock interrupt processing and CPU scheduling). Then, PtpS1 sends a Dreq_Time_Req message again to the Drifting Clock to obtain the time stamp of $t_3$. Once receiving the Dreq_Time_Req, the Drifting Clock retrieves its local clock time $t_3$ and replies with a Time_Res message containing $t_3$.

When PtpS1 receives the Time_Res containing $t_3$, it records $t_3$ and immediately sends a Delay_Req packet to the Master.

When Master receives the Delay_Req packet it records its arrival time $t_4$ and encapsulates $t_4$ into a Delay_Res packet containing $t_4$. The Delay_Res packet is then sent back to the SIM2.

When the PtpS1 of SIM2 node receives the Delay_Res, it retrieves the time-stamp $t_4$.

Once the above packet and message exchange procedures are completed, PtpS1 of SIM2 node acquire all the four time stamps $(t_1, t_2, t_3, t_4)$. Then the PtpS1 is able to calculate the offset between the Master’s Ideal Clock and the SIM2’s Drifting Clock by the offset estimation equation (8).

B. PTP Packet Exchange in the Second Hop

The process of time-stamped PTP packet exchange among the SIM2 node and Slave2 node in the second hop and the time-stamping procedure within the Slave2 node are described as follows.

The PtpM2 sends a Sync2_time_req message to the SIM2’s Drifting Clock periodically at an interval $\Delta T$. The Drifting Clock receives the Sync2_time_req, and obtains its arrival time $t_5$, then, sends a Time_Res message containing $t_5$ to the PtpM2.

Once receiving the Time_Res message from the SIM2’s Drifting Clock, time $t_5$ is encapsulated into a Sync packet and PtpM2 sends the Sync packet immediately to
the PtpS2 in the Slave2 of the second hop. The Sync contains the time \( t_1 \), which is the Sync’s sending time.

When the Sync first arrives at the input gate of Slave2, it is processed by Slave2’s PtpS2 block. From now on, the reset of PTP packet exchange in the second hop is the same as that in the first hop, as shown in step (2)–(6) in previous subsection. The only difference from the first hop is that the PTP packet exchange in the second hop is between PtpM2 and PtpS2.

Remark: Compared to the time-stamping of Sync in Master node, the time-stamping of Sync in Slave2 node is provided by the Drifting Clock, rather than the reference clock. It worth pointing that, before sending out the Sync packet to Slave2, Slave2’s PtpM2 first acquires the time of the drifting clock by sending a Sync2_time_req message to the Slave2’s Drifting Clock and then, encapsulate the time-stamping into the Sync packet.

V. NOISES IN IEEE 1588 OFFSET CALCULATION

In the IEEE1588 PTP standard, we generally assume that propagation delay is equivalent, namely \( d_{i_m}=d_{i_a} \); however, in the real networks, due to wireless media sharing, conflict avoidance and other factors, the propagation delay is assumed to be asymmetry between the master node and slave node, that is \( d_{i_m}\neq d_{i_a} \). In this paper, the propagation delay asymmetry is described as delay jitter, so the propagation delay \( d_{i_m} \) and \( d_{i_a} \) are considered as Gaussian random process \( N(d,\Delta d) \) with mean \( d \) and variance \( \Delta d^2 \). Let \( \Delta d=(d_{i_m}-d_{i_a})/2 \), equation (8) can be expressed as:

\[
\theta_{\Delta d}(n) = \frac{[(t_2(n)-t_1(n))-(t_3(n)-t_2(n))]}{2} + \Delta d
\]

where \( \Delta d \) is considered as Gaussian random variable with zero-mean and variance \( 1/2*\Delta d^2 \). In embedded systems, timestamp uncertainty is significantly affected by several different factors such as interrupt latency, timing jitter and scheduling, so the measurement value \( t_i (i=1, 2, 3, 4) \) of both the master clock and slave clock have also time stamp uncertainty \( \Delta t_i \). Both \( \Delta t_i \) and \( \Delta t_p \) correspond to the timestamp uncertainty of the master clock, both \( \Delta t_i \) and \( \Delta t_p \) correspond to the timestamp uncertainty of the slave clock. Therefore, the measurement equation of clock offset \( \theta_{\Delta d} \) can be written in the following form:

\[
\theta_{\Delta d}(n) = \frac{[(t_2(n)-t_1(n))-(t_3(n)-t_2(n))]}{2} + v_{\Delta d}(n)
\]

where corresponds to measurement noise of the clock offset \( \theta_{\Delta d} \) which is defined as follows.

\[
v_{\Delta d}(n) = \frac{\Delta t_1 + \Delta t_2}{2} - \frac{\Delta t_1 + \Delta t_2}{2} + \Delta d
\]

The variance is the sum of variances of each independent random variables. It is defined as follows.

\[
\delta_{\Delta d}^2 = \frac{1}{2} (\delta_{\Delta d_{MTS}}^2 + \delta_{\Delta d_{STS}}^2 + \delta_{\Delta d_p}^2)
\]  

(14)

We assume that timestamp uncertainty for the master node can be disregarded, so \( \delta_{\Delta d_{MTS}}^2 \) is equal to zero; likewise, \( \delta_{\Delta d_p}^2 \) can be initially ignored, unless the uncertainty on propagation delay is specifically assumed to be symmetry in the study. Thus, the skew uncertainty is obtained from

\[
\delta_{\gamma_d}^2 = \frac{2}{\Delta T^2} \delta_{\Delta d_{STS}}^2
\]

(15)

It is worth pointing out that the skew \( \gamma_d(n) \) is obtained by the measurement value of the offset \( \theta_{\Delta d} \), and the measurement noise and are correlated, so their covariance is.

\[
\text{Cov} (v_{\Delta d}, v_{\gamma_d}) = \text{Cov} (v_{\theta_{\Delta d}}, v_{\gamma_d}) = \sqrt{2} \frac{\delta_{\Delta d}^2}{\Delta T}
\]

(16)

VI. KALMAN FILTERING FOR MULTI-HOP TIME SYNCHRONIZATION

In order to adopt the Kalman filter to improve the performance of offset and skew estimation as described by the IEEE 1588 standard, the clock’s state equations (4) and (5) have to be written slightly to fit the state space model. Generally, time synchronization occurs periodically at a fixed time synchronization interval \( \Delta T \). In addition, considering the facts that the time-stamped PTP packet exchange can be completed in a short time and the variation of a physical clock in such a short period is slow, it is reasonable to assume the clock offset and skew are constant between two time synchronization interval \( \Delta T \). As a result, the clock’s state equations (4) and (5) are written as:

\[
\theta(n+1) = \theta(n) + \gamma(n) \Delta T + \omega_\theta(n)
\]

\[
\gamma(n+1) = \gamma(n) + \omega_\gamma(n)
\]

(17)

(18)

where \( n \) represents the \( n-th \) time synchronization procedure, \( \theta(n) \) and \( \gamma(n) \) represent the offset and the skew between the master node and slave node at the \( n-th \) time synchronization procedure, respectively. Both the \( \omega_\theta(n) \) and \( \omega_\gamma(n) \) correspond to the phase noise on \( \theta(n) \) and the frequency noise on \( \gamma(n) \), respectively.

In order to implement time synchronization between the slave clock and the master clock, the inputs corrections \( \mu_d(n) \) and \( \mu_f(n) \) are calculated at the \( n-th \) synchronization procedure and applied to adjust the offset and skew, respectively. Therefore, the current value of both offset and skew calculated in equations (17) and (18) at the \( n-th \) synchronization procedure can be written as recursive relationship:

\[
\theta(n+1) = \theta(n) + \mu_d(n) + [\gamma(n) - \mu_f(n)] \Delta T + \omega_\theta(n)
\]  

(19)
\[ \gamma(n+1) = \gamma(n) - \mu_x(n) + \omega_x(n) \]  
\[ \mu_x(n) = \theta_x(n) \]  
\[ \gamma_x(n) = \gamma_x(n) \]  
where \( \theta(n) \) represents the current value of offset at the \( n \)-th synchronization procedure, \( \gamma(n) \) represents the current value of skew at the \( n \)-th synchronization procedure.

Ideally, if the timestamp information is sufficiently accurate, measurement value of both the clock offset and skew can be directly applied as corrections to the slave clock, that is:

\[ \mu_x(n) = \theta_x(n) \]  
\[ \gamma_x(n) = \gamma_x(n) \]  

In real networks, timestamp information is occasionally inaccurate and it is necessary to be preprocessed by some filters. We can know by analyzing the previous sections, it becomes almost natural to consider Kalman filtering can be used to achieve recursive estimator, consequently the clock state equations (19) and (20) can be expressed as matrix form:

\[ \bar{x}(n) = Ax(n) + B\mu(n-1) + \alpha(n-1) \]  

where \( \mu(n)=[\mu_x(n), \mu_y(n)]^T \) and \( \bar{x}(n)=[\theta_x(n), \gamma_x(n)]^T \) are input control vector and the state vector at the \( n \)-th synchronization, respectively. \( \alpha(n)=[\omega_x(n), \omega_y(n)]^T \) is the process noise subject to Gaussian random process \( N(0, \mathbf{Q}) \), both \( A = [1 \Delta T; 0 \quad 1] \) and \( B = [-1 \Delta T; 0 \quad -1] \) represent state transition matrix and input control matrix, respectively.

Likewise, both equations (8) and (10) correspond to the measurement equation of offset \( \theta_x(n) \) and skew \( \gamma_x(n) \), respectively, can be expressed as the vector matrix measurement equation by defining the clock measurement vector \( y(n)=[\theta_x(n), \gamma_x(n)]^T \)

\[ y(n) = H\bar{x}(n) + \nu(n) \]  

where \( H = [1 \quad 0; 0 \quad 1] \) is an identity matrix of measurement systems and \( \nu(n) \) represents measurement noise of a Gaussian random process \( N(0, \mathbf{R}) \).

The Kalman filter for recursive equations are written in the following:

\[ \bar{x}(n|n-1) = Ax(n-1) \]  
\[ P(n|n-1) = AP(n-1)A^T + Q \]  

where the matrix \( \mathbf{Q} \) corresponds to the process noise covariance matrix. Since the random processes \( \theta(n) \) and \( \gamma(n) \) are independent, \( \mathbf{Q} \) is a 2x2 diagonal matrix which is reported:

\[ Q = \begin{bmatrix} \delta_\theta^2 & 0 \\ 0 & \delta_\gamma^2 \end{bmatrix} \]  

If these parameters are known, initialization of the matrix \( \mathbf{P}(n|n-1) \) is \( \mathbf{P}(1|0) = \mathbf{Q} \).

The general expression for the Kalman gain \( \mathbf{K}(n) \) is expressed as:

\[ \mathbf{K}(n) = \mathbf{P}(n|n-1)H^T[HP(n|n-1)H^T + \mathbf{R}]^{-1} \]  

where \( \mathbf{R} \) corresponds to the measurement noise covariance matrix, which must be set with the measurement uncertainties calculated in equations (14) and (15).

\[ R = \delta_{\omega}^2 \begin{bmatrix} 1 & \frac{\sqrt{2}}{\Delta T} \\ \frac{\sqrt{2}}{\Delta T} & \frac{2}{(\Delta T)^2} \end{bmatrix} \]  

It worth pointing out that \( \mathbf{H} \) is an identity, as a consequence, the correction equations is given by the following expression.

\[ \bar{x}(n) = \bar{x}(n|n-1) + \mathbf{K}(n)[y(n) - \bar{x}(n|n-1)] \]  
\[ P(n) = [1 - \mathbf{K}(n)]P(n|n-1) \]  

The state estimate is obtained by equation (29) which is designed to give a better and accurate estimate of the local clock’s offset and skew. Once the offset and skew is estimated, a clock correction is carried out to modify the drifting clock’s offset and skew accordingly and the time synchronization between the master node and slave node can be achieved.

VII. PERFORMANCE EVALUATION

In this section, we introduce the development of our multi-hop simulator and describe the implementation of PTP packet exchange in multi-hop WSNs. Then, the performance of the proposed Kalman filtering for PTP was analyzed through several simulations. The simulation and performance analysis are done at various scenarios by changing the standard deviation of offset and skew of the slave’s local clock.

A. Development of the Multi-Hop Simulator

In order to simulate the clock model based on Kalman filter, a multi-hop network is constructed on OMNeT++ network simulator. In our simulator, three types of PTP time packets are defined according to the PTP standards. Namely, they are Sync, Delay_req and Delay_Resp. The topology of the multi-hop network is shown in Fig.4, where the Master1 is the master node for the first hop time synchronization and all the slave nodes are to keep their clocks synchronized with Master1’s perfect clock. SIM2 node works as the PTP slave in the first hop and as the master node for the second hop. Slave2 is a slave node working as the PTP slave in the second hop.

![Figure 4. The topology of the two-hop network](image-url)
simple module, namely PtpS1, Clock, PtpM2, Bufferrx, Buffertx, and Manager, among which the PtpS1 is the main module of SIM2, and it is employed to achieve the function of the PTP slave node, receives and sends cyclic and acyclic messages; the PtpM2 is used to achieve the function of the PTP master node specified by the IEEE 1588 standard; the Bufferrx and Buffertx are realizations of the buffer block, which is used to simulate the behaviour of a first in first-out (FIFO) transmission queue; the manager is responsible for generating cyclic or acyclic traffic row. In addition, it worth pointing out that the Slave2 has a similar structure with the SIM2, but the Slave2 has not the PtpM2 which is used to implement the functions of the PTP master node.

Figure 5. Implementation of the S1M2 node

In our simulator, the time synchronization interval $\Delta T$ for the first hop and second hop are the same. When the first hop time synchronization starts, the second hop time synchronization also starts. The implementation of the PTP packets exchange in multi-hop time synchronization is described as follows.

A time synchronization timer in Master1 is triggered at an interval $\Delta T$. When the timer fires, the first hop time synchronization is initiated and Master1 sends a Sync message to SIM2. From now on, the packet exchange in the first hop is the same as the PTP packet exchange in the first hop, as shown in step (1)-(6) in subsection A of section IV. Since Master1 is the master node in the first hop, SIM2 is to keep time synchronized with Master1.

Another time synchronization timer in PtpM2 is also triggered at a synchronization interval $\Delta T$. When the timer fires, PtpM2 sends a Sync2_time_req message to its Clock module to acquire the time value of the drifting local clock. From now on, the packet exchange in the second hop is the same as the PTP packet exchange as shown in step (1)-(3) in subsection B of section IV. Please note that SIM2 is the master node in the second hop time synchronization and Slave2 is to keep time synchronized with SIM2.

B. Simulation Results

In the first simulation experiment, the initial values of the offset and skew of the drifting clocks of S1M2 and Slave2 are 0s and 10ppm, respectively. The update interval of the slave clocks is 0.1ms, which is much smaller than the synchronization interval $\Delta T$ ($\Delta T=0.1s$).

Please note that $\Delta T$ is also the interval of recursive Kalman filtering algorithm. Different levels of time-stamping uncertainties are simulated by using different measurement noises. This is done by setting the standard deviation ($\delta_{STS}$) to a value in the range of $[10^{-2}~10^{-8}]$. A small $\delta_{STS}$ corresponds to the accurate hardware-assisted time stamping and a larger $\delta_{STS}$ represents the fluctuating software time stamping.

![Figure 6. Offset standard deviation against timestamp uncertainty ($\delta_{STS}$)](image6)

The simulation results are shown in Fig. 6 and 7, where two different time synchronization approaches are compared. One uses the standard IEEE 1588 PTP alone without Kalman filtering and another one is the proposed PTP scheme with Kalman filtering.

Fig. 6 shows that, in both the first and second hop, the standard deviation of clock offset given by the proposed PTP with Kalman filtering are significantly smaller than the case without Kalman filtering. This verifies the performance improvement of the proposed algorithm. Furthermore, it can be seen that the offset standard deviation of the second hop is larger than the offset standard deviation in the first hop. This applies to both the standard PTP without Kalman filtering and the improved PTP with Kalman filtering. This means that the accumulation of offset and skew errors appears in multi-hop time synchronization network.
Third, it can be seen that, in the standard PTP without Kalman filtering, when \( \delta_{STS} > 10^{-5} \) with the timestamps uncertainty increasing, the offset standard deviation have a steady increment. This suggests that, the timestamp uncertainty is the main factor which affects the clock stability. Meanwhile, in the standard PTP without Kalman filtering, the offset standard deviation of the second hop is significantly larger than that of the first hop. On the other hand, in the proposed PTP with Kalman filtering, the offset standard deviation of the second hop is almost the same as the value of the first hop and the offset standard deviation has almost no change and is basically stable at a fixed value.

Fig. 7 shows that the skew standard deviation without Kalman filtering increases as the timestamp uncertainty is increasing. In the standard PTP without Kalman filtering, when \( \delta_{STS} > 10^{-5} \), with the timestamps uncertainty increasing, the offset standard deviation will have a gradual increment. However, in the proposed PTP with Kalman filtering, the skew standard deviation is not only significantly smaller than the case without Kalman filtering, but also maintains a comparatively stable trend. Meanwhile, in the standard PTP without Kalman filtering, the skew standard deviation of the second hop is significantly larger than that of the first hop. In the proposed PTP with Kalman filtering, the skew standard deviation of the second hop is almost the same as the first hop.

Based on the analysis of Fig 6 and 7, we can make a conclusion that the accumulation of synchronization error in the proposed PTP with Kalman filtering is smaller compared with the standard PTP without Kalman filtering. Furthermore, the PTP with Kalman filtering can achieve better time synchronization performance and stability compared with the standard PTP without Kalman filter in multi-hop WSNs. This mean that Kalman filter can eliminate noise and reduce the accumulation of synchronization error and results in the better adaptability in multi-hop time synchronization networks.

Third, it can be seen that, in the standard PTP without Kalman filtering, when \( \delta_{STS} > 10^{-5} \) with the timestamps uncertainty increasing, the offset standard deviation have a steady increment. This suggests that, the timestamp uncertainty is the main factor which affects the clock stability. Meanwhile, in the standard PTP without Kalman filtering, the offset standard deviation of the second hop is significantly larger than that of the first hop. On the other hand, in the proposed PTP with Kalman filtering, the offset standard deviation of the second hop is almost the same as the value of the first hop and the offset standard deviation has almost no change and is basically stable at a fixed value.

Fig. 7 shows that the skew standard deviation without Kalman filtering increases as the timestamp uncertainty is increasing. In the standard PTP without Kalman filtering, when \( \delta_{STS} > 10^{-5} \), with the timestamps uncertainty increasing, the offset standard deviation will have a gradual increment. However, in the proposed PTP with Kalman filtering, the skew standard deviation is not only significantly smaller than the case without Kalman filtering, but also maintains a comparatively stable trend. Meanwhile, in the standard PTP without Kalman filtering, the skew standard deviation of the second hop is significantly larger than that of the first hop. In the proposed PTP with Kalman filtering, the skew standard deviation of the second hop is almost the same as the first hop.

Based on the analysis of Fig 6 and 7, we can make a conclusion that the accumulation of synchronization error in the proposed PTP with Kalman filtering is smaller compared with the standard PTP without Kalman filtering. Furthermore, the PTP with Kalman filtering can achieve better time synchronization performance and stability compared with the standard PTP without Kalman filter in multi-hop WSNs. This mean that Kalman filter can eliminate noise and reduce the accumulation of synchronization error and results in the better adaptability in multi-hop time synchronization networks.

In the second simulation experiment, we compared the performance of the proposed multi-hop time synchronization at two different clocks. Clock A is a relatively stable clock with \( \delta_{STS} = 10^{-16} \) and \( \sigma_{\theta} = 10^{-14} \) [24]. Another clock is a relatively unstable clock B with \( \delta_{STS} = 10^{-16} \) and \( \sigma_{\theta} = 10^{-12} \) [25]. The results of the standard PTP without Kalman filtering and the proposed one with Kalman filtering for different clocks are compared in Fig. 8 and Fig. 9. Both figures show the results of the slave clock in the second hop.

Fig. 8 shows that, for both Clock A and B, in the all range of timestamp uncertainty, the offset standard deviation with Kalman filtering is smaller than the case without Kalman filtering. In the standard PTP without Kalman filtering, the offset standard deviation of clock A and clock B show a gradual increment as the timestamps uncertainty increasing. When the time stamping uncertainties are large (\( \delta_{STS} > 10^{-5} \)), it dominates the synchronization errors and the offset standard deviation of Clock A is almost equal to the offset standard deviation of Clock B. While, in the proposed PTP with Kalman filtering, the offset standard deviation of clock A is significant smaller than that of clock B, even the time stamping uncertainties are large. Meanwhile, we can find that the offset standard deviation of clock B with Kalman filtering is smaller than that of clock A without Kalman filtering.

Similarly, Fig. 9 shows the propose PTP with Kalman filtering performs better than the standard PTP without Kalman filtering in terms of skew estimation. In the standard PTP, the skew standard deviation shows a significant increasing trend as a larger measurement noise occurs. However, with Kalman filter, the skew standard deviation maintain relatively stable and is significantly smaller than that of the PTP without Kalman filtering. This applies to both clock A and clock B. Furthermore, the skew standard deviation of clock A is significant smaller than that of clock B.

From Fig. 8 and 9, we can conclude that the multi-hop time synchronization performance can be improved significantly by adapting the Kalman filtering techniques. In particular, for inaccurate clock (like Clock B in the simulation), the performance improvement is more
obvious, which is a good benefit for low-cost WSNs, as the clocks of the WSN nodes are mostly inaccurate because of the low price and varying working conditions.

VIII. CONCLUSION

In this paper, we proposed a Kalman filtering to improve the IEEE 1588 PTP time synchronization performance in multi-hop wireless sensor networks. The performance of the proposed algorithm is validated by our developed multi-hop WSN simulator on the OMNe++ platform. The result shows that, compared with the PTP without Kalman filtering, the proposed PTP with Kalman filtering is able to achieve higher precision and stability in multi-hop wireless time synchronization.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (NSFC) under grant 61101135, the Fundamental Research Funds for the Central Universities (XDJK2012C065, XDJK2014C167) and Science and Technology Commission of Shanghai Municipality under grant 13XD1403400.

REFERENCES