DA-Sync: A Doppler-Assisted Time-Synchronization Scheme for Mobile Underwater Sensor Networks

Jun Liu, *Student Member*, *IEEE*, Zhaohui Wang, *Student Member*, *IEEE*, Michael Zuba, *Student Member*, *IEEE*, Zheng Peng, *Student Member*, *IEEE*, Jun-Hong Cui, *Member*, *IEEE*, and Shengli Zhou, *Member*, *IEEE*

Abstract—Time synchronization plays a critical role in distributed network systems. In this paper, we investigate the time synchronization problem in the context of underwater sensor networks (UWSNs). Although many time-synchronization protocols have been proposed for terrestrial wireless sensor networks, none of them can be directly applied to UWSNs. This is because most of these protocols do not consider long propagation delays and sensor node mobility, which are important attributes in UWSNs. In addition, UWSNs usually have high requirements in energy efficiency. To solve these new challenges, innovative time synchronization solutions are demanded. In this paper, we propose a pairwise, cross-layer, time-synchronization scheme for mobile underwater sensor networks, called *DA-Sync*. The scheme proposes a framework to estimate the doppler shift caused by mobility, more precisely through accounting the impact of the skew. To refine the relative velocity estimation, and consequently to enhance the synchronization accuracy, the Kalman filter is employed. Further, the clock skew and offset are calibrated by two runs of linear regression. Simulation results show that DA-Sync outperforms the existing synchronization schemes in both accuracy and energy efficiency.

Index Terms—UWSNs, synchronization, mobility, sensor node, propagation delay

1 Introduction

In past several years, underwater sensor networks (UWSNs) have drawn considerable attention from both academia and industry [2], [11], [17], [33], [34]. UWSNs facilitate or enable a wide range of aquatic applications, including coastal surveillance, environmental monitoring, undersea exploration, disaster prevention, and mine reconnaissance. However, due to the high attenuation of radio waves in water, UWSNs have to rely on acoustic communications [1], [39]. The unique characteristics of underwater acoustic communications and networking, such as low available bandwidth, long propagation delays, high error probability, and sensor node (passive or proactive) mobility (in mobile networks) pose grand challenges to almost every layer of network protocol stack and applications [2], [9], [11], [17], [25], [32], [34], [35], [36], [37], [38].

In this paper, we tackle the time synchronization problem, which is critical to many UWSN design issues. In fact, most UWSNs applications either benefit from or require time synchronization services. For instance, data collected by underwater sensor nodes need global time stamps to make data fusion and processing meaningful. TDMA, one of the commonly used medium access control (MAC) protocols, often requires good synchronization among sensor nodes. Furthermore, time synchronization is indispensable in most of the localization algorithms, in both underwater sensor networks [5], [8], [20], [26], [44] and terrestrial sensor networks [6], [15], [18], [19], [24], [43].

In the literature, numerous time synchronization protocols have been proposed for terrestrial wireless sensor networks [12], [13], [14], [30], [40]. Most of them claim to be able to achieve high accuracy with reasonable energy expenditure. However, these algorithms cannot be directly applied to UWSNs. This is because most of these approaches assume negligible propagation delays among sensor nodes, which is not true in UWSNs. UWSNs often feature long propagation delays due to the low transmission speed of sound in water (about 1500 m/s). In addition, for mobile UWSNs, propagation delays between nodes are time-varying because of sensor node mobility. Furthermore, acoustic transmissions are power demanding, which requires high energy efficiency. All of these features in UWSNs bring new challenges for time synchronization algorithms.

Recently, some time synchronization algorithms, such as TSHL [41], MU-Sync [10], Mobi-Sync [28], and D-Sync [29] have been proposed for UWSNs. In these algorithms, the issue of long propagation delays is often well addressed. However, they all ignore one issue or another. For example, TSHL assumes that nodes are fixed, which makes it not suitable for mobile networks. While MU-Sync is designed for mobile underwater networks, it is not energy efficient, and Mobi-Sync is for dense network. In this paper, we propose a novel time-synchronization scheme, called *DA-Sync*, which is a fundamental cross-layer-designed time-synchronization

- *J. Liu is with the Institute of Computing Technology, Chinese Academy of Science. E-mail: liuj@nibcc.com.*
- *Z. Wang, M. Zuba, Z. Peng, J.-H. Cui, and S. Zhou are with the Underwater Sensor Network Lab, University of Connecticut, Storrs, CT 06269.*
  E-mail: {zhwang, michael.zuba, zhp05001, jcui, shengli}@engr.uconn.edu.

Manuscript received 2 Feb. 2012; revised 6 Aug. 2012; accepted 2 Jan. 2013; published online 14 Jan. 2013.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2012-02-0054. Digital Object Identifier no. 10.1109/TMC.2013.13.

protocol specific for mobile UWSNs, with high accuracy and high energy efficiency as its major design goals.

For time synchronization, the basic task is to get two clocks synchronized, no matter what kind of network topology those clocks sit in. DA-Sync provides a fundamental method to synchronize two sensor nodes, i.e., an ordinary node and a reference node, for various scenarios, for example, one hop network/multiple hop network, peer-to-peer/master-slave, proactive/reactive, and distributed/centralized. Any time synchronization protocol specific for a certain scenario can be built based on DA-Sync. Additionally, DA-Sync is the first physical-MAC cross-layer-designed time-synchronization scheme. In the physical layer, a well-designed preamble and a Doppler scaling factor estimation algorithm are adopted to measure the relative velocities of one sensor node to the others. These velocities are further refined using the Kalman filter. By incorporating relative moving velocities between the transmitter and receiver, the accuracy of the propagation-delay estimation is greatly improved compared with traditional methods, which usually utilize half of the round trip time to estimate one way propagation delay. In addition, by differentiating the propagation delays of "Sync-Req"(synchronization request message) and "Sync-Res"(synchronization response message), two observation sample points can be obtained in one message exchange process, which makes synchronization messages efficient in terms of "sampling" rate. Furthermore, with the help of the MAC layer time stamping, we can further reduce the nondeterministic errors that are commonly encountered by time synchronization algorithms which rely on message exchanges. Using simulations, we show that DA-Sync outperforms the existing schemes in both accuracy and energy efficiency [27].

The remainder of this paper is organized as follows: We first review design challenges and some related work in Section 2. Then we introduce how to utilize Doppler shifts to estimate relative velocities in Section 3 and describe DA-Sync in detail in Section 4. Following that, we conduct error analysis for the delay estimation in Section 5 and present simulation results in Section 6. Finally, we offer our conclusion and future work in Section 7. Table 1 shows all the notations in this work.

TABLE 1
Notation Summary

θ, β	clock skew, clock offset
$\dot{\theta}$	initial or draft skew
η	sound propagation speed
t_r	response time(time interval t_3 - t_2)
v_0	ordinary node's average relative velocity from T_1 to t_2
v_1	ordinary node's average relative velocity from t_3 to T_4
v_0^m	measured v_0 by exploring Doppler effects
v_1^m	measured v_1 by exploring Doppler effects
v	real velocity
$v^e(k)$	estimated $v(k)$ with Kalman Filter
v_0^e	estimated v_0 with Kalman Filter
α	real acceleration
α^e	estimated acceleration
τ_1	propagation delay of Sync-Req
τ_2	propagation delay of Sync-Res
$\Delta\tau$	τ_2 - τ_1
Δd	$\Delta\tau\eta$, relative moving distance during t_2 to T_4
T_1	sending time of Sync-Req
t_3	sending time of Sync-Res
t_2	receiving time of Sync-Req
T_4	receiving time of Sync-Res

2 Challenges and Related Work

2.1 Design Challenges

To achieve time synchronization in UWSNs, two critical challenges have to be addressed. First, the underwater acoustic channel features long and dynamic propagation delays, which makes the conventional two-way delay measurements inefficient and inaccurate. Second, constrained energy due to limited power supply restricts the time synchronization overhead.

2.1.1 Long and Dynamic Propagation Delays

Any software solution to time synchronization via message exchanges has to face several uncertainties which could affect accuracy. Those uncertainties include sending time, accessing time, transmission time, propagation time, reception time, interrupt handling, encoding time, decoding time, and byte alignment time [21], [22]. In UWSNs, among these uncertainties, the propagation time is dominant due to the low propagation speed of acoustic signals. And also, sensor nodes in an underwater environment often have passive mobilities caused by water currents or proactive mobilities with use of mobile platforms. Those features complicate time synchronization by causing long and variational propagation delays. In such a situation, it is hard, if not impossible, to estimate real-time propagation delays between two sensor nodes. However, most of the existing time-synchronization schemes use half of the round trip time to calculate one way propagation delay. Due to the node mobility, propagation delays on the way forth and back are not necessarily identical, especially when nodes move at high speed. This issue severely decreases the accuracy of most time synchronization approaches.

2.1.2 Energy Constraints

Underwater sensor nodes are usually powered by battery, for which it is difficult to replenish. Therefore, the lifetime of sensor nodes is restricted by the limited power supply. Computation and communication are the two major energy costs associated with synchronization. Computation consists of calculations performed on the sensor node platform chip to achieve synchronization, while communication relates to the energy consumed during the transmission and reception of synchronization reference packets. The amplifier and transducer of an acoustic modem typically

need tens of watts to reliably transmit signals to another sensor node within a reasonable distance. However, the power consumption of most processing chips is on the order of a single watt or less. This means that the energy consumed by acoustic communication in UWSNs is much higher than computation. Thus, to some extent, it is worthy to design an algorithm which requires a relatively high computational complexity, but helps reduce the number of exchanged synchronization reference packets. Theoretically, to achieve a low-synchronization overhead, both clock skew and offset need to be estimated and compensated to avoid frequent resynchronization, as the skew leads to increasing time difference. Additionally, the "sampling" rate, which represents the percentage of the exchanged messages that can be used as sample points for synchronization, should be as high as possible.

2.2 Related Work

Although there are significantly growing interests in UWSNs over the past several years, the research on time synchronization for UWSNs is still relatively limited.

TSHL [41] is designed for high latency networks, which can manage long propagation delays and remain energy efficient simultaneously. TSHL combines one-way and two-way MAC-layer message delivery. One-way communication is used to estimate the clock skew and two-way is used for clock offset. TSHL works well for static underwater sensor networks, but it cannot handle mobile scenarios as it assumes constant propagation delays among sensor nodes. Particularly, when nodes move fast, TSHL performs even worse than no time synchronization as described in [10].

MU-Sync[10] is proposed to synchronize nodes in a cluster based UWSNs. In MU-Sync, the cluster head is responsible for launching the time synchronization process, determining the number of reference messages, calculating clock skew and offset for every ordinary node and broadcasting the calculated results to all nodes within the cluster. MU-Sync performs two runs of linear regression. The first run allows the cluster head to estimate the draft skew by totally ignoring the variance of propagation delays. The second run is used to correct the estimated skew and calculate the offset. Although MU-Sync aims to solve sensor node mobility issue in UWSNs, it requires relatively high message overhead. Furthermore, MU-Sync uses half of the round trip time to calculate one-way propagation delay, which causes extra errors, especially for fast moving situations or when ordinary nodes respond to the cluster head after experiencing a long time duration.

Mobi-Sync applies spatial correlation of nodes' velocities to estimate the time varying propagation delay. It is an assumed hierarchal structure in which surface buoys are equipped with GPS to obtain global time reference and super nodes can communicate directly with them to maintain synchronization. While being effective in estimating the time varying delay, this protocol needs to know the exact correlation model between nodes, which is very hard to obtain. In addition, for Mobi-Sync, the network has to be densely deployed to ensure that each ordinary node maintains connectivity to at least three or more super nodes to have a good estimate of velocity.

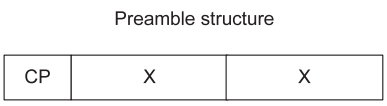


Fig. 1. Preamble structure, consisting of two identical waveforms preceded with a cyclic prefix.

The most recent work, D-Sync [29], is the first work that leverages the Doppler shift in underwater environments to do time synchronization. In D-Sync, by estimating and compensating for the Doppler shift, the calculation accuracy of propagation delay and synchronization are improved. However, D-Sync does not consider the effect of the skew during the process of estimating the Doppler scaling factor, which reduces the accuracy of Doppler shift estimation and affects the accuracy of time synchronization. Furthermore, D-Sync completely trusts the measured speed with Doppler shift, which also leads to synchronization errors.

3 Doppler Scaling Factor Estimation

In this section, we show how to utilize Doppler scale estimation in UWSNs to gather sensor nodes' relative moving velocities. We start with a general approach for the Doppler scaling factor estimation, and then talk about the resolvability between the sensor mobility and the clock skew in underwater sensor networks.

3.1 Estimation of Doppler Scaling Factor

The Doppler scale estimation has been an interesting topic ever since the appearance of wireless communications, and in the underwater acoustic communication, it plays a more critical role. In this paper, we consider a novel Doppler estimation strategy using a bank of autocorrelators with a well-designed preamble [31].

The preamble structure for Doppler estimation is shown in Fig. 1, in which a cyclic prefix (CP) is added in front of two identical waveforms. Defining $x(t)$ as the baseband signal of the waveform, the repetition pattern can be expressed as

$$x(t) = x(t + T_0), \quad -T_c \le t \le T_0, \tag{1}$$

where T_0 and T_c denote the time duration of the waveform and the cyclic prefix, respectively. At the receiver side, after certain manipulation, it can be shown that the received baseband signal is expressed as

$$y(t) = e^{-j2\pi\frac{a}{1+a}f_c T_0} y\left(t + \frac{T_0}{1+a}\right), \quad -\frac{T_c - \tau_{\max}}{1+a} \le t \le \frac{T_0}{1+a}, \tag{2}$$

where f_c is the center frequency, a is the Doppler scaling factor we will estimate, and $\tau_{\max}$ denotes the channel multipath delay spread. The discrete time expression of the $y(t)$ at a sampling rate of λB is

$$y[n] = y(t) \mid_{t=n/(\lambda B)}, \tag{3}$$

where B is the signal frequency bandwidth, and λ is an integer representing the time-domain oversampling factor.

To estimate the Doppler scaling factor, we exploit the periodic property of the received preamble via a bank of

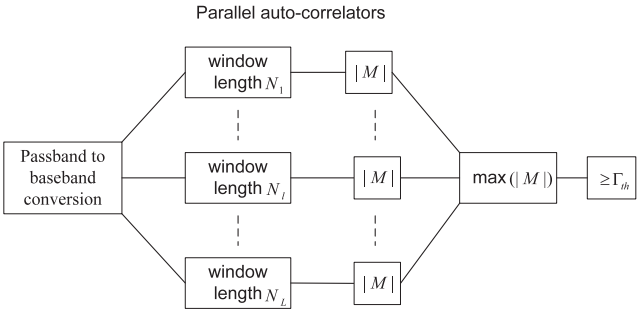


Fig. 2. Structure of parallel autocorrelators.

autocorrelators as shown in Fig. 2. Defining N_l as the autocorrelation window length of the lth branch, the correlation metric of the lth branch can be calculated as

$$M(N_l, d) = \frac{\sum_{i=d}^{d+N_l-1} y[i]y[i+N_l]}{\sqrt{\sum_{i=d}^{d+N_l-1} |y[i]|^2 \cdot \sum_{i=d}^{d+N_l-1} |y[i+N_l]|^2}}, \quad (4)$$

where d is the index of autocorrelation outputs. Note that N_l should be chosen close to $T_0/(\lambda B)$, which is the number of baseband samples of $x(t)$.

Once the maximum value of the correlation metrics at any branch is larger than a certain threshold Γ_{th}, the receiver declares the detection of a useful signal. The Doppler scaling factor can be estimated based on the window length of the branch which has the maximum autocorrelation output

$$\hat{a} = \frac{T_0/(\lambda B) - \hat{N}}{\hat{N}} \qquad \text{where } \hat{N} = \arg\max_{\{N_l\}} |M(N_l, d)|. \quad (5)$$

Obviously, one can increase the time-domain resampling factor λ to increase the Doppler estimation accuracy. However, the maximum values of λ cannot be larger than the ratio of the passband sampling rate to the signal bandwidth B.

A linear interpolation can be adopted at the autocorrelator output to improve the estimation accuracy. Define l as the index of the branch which has the largest correlation output. Let $|X_l|$ denote the maximum amplitude of the metric at the lth branch, and $|X_{l-1}|$ and $|X_{l+1}|$ as the neighbors from the $(l-1)$th and $(l+1)$th branch, respectively. Define Δa as the spacing of the Doppler searching grids. The offset of the Doppler scaling factor estimation can be interpolated as

$$\delta = \frac{|X_{l+1}| - |X_{l-1}|}{4|X_l| - 2|X_{l-1}| - 2|X_{l+1}|} \Delta a, \quad (6)$$

which leads to $\tilde{\hat{a}} = \hat{a} + \delta$. Thus, the estimation of the velocity is $\hat{v} = \tilde{\hat{a}}c$, where c is the sound speed in water.

Fig. 3 demonstrates simulation results of the root-mean-square error for the speed estimates in both high and low SNR scenarios [31]. An orthogonal frequency multiplexing modulated (OFDM) symbol with 512 subcarriers is used to generate the preamble. The oversampling factor is chosen to be $\lambda = 8$. Considering the computational complexity, two stages of velocity estimation are adopted. A coarse estimate will be obtained in the first stage with large grid spacing $\Delta a = \Delta v/c$, where $\Delta v = 1.46$ m/s. The estimate will be refined in the second stage with fine grid spacing $\Delta a = \Delta v/c$, where $\Delta v = 0.366$ m/s. One can find that with the help of the interpolating operation, high accuracy estimation of velocity can be achieved.

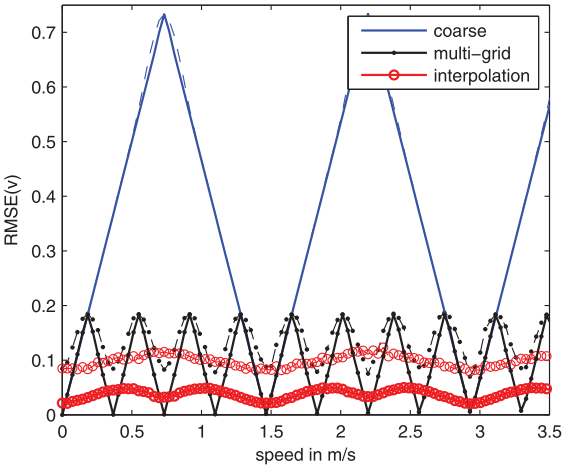


Fig. 3. Average velocity estimation error using varying amounts of correlators and a simple interpolation between the measured points; thin, dashed lines are $\gamma = 0$ dB and thick full lines are $\gamma = 10$ dB.

3.2 Resolvability between Sensor Mobility and Clock Skew

In the Doppler estimation method discussed above, an underlying assumption is that the clocks of the transmitter and receiver are synchronized. However, in underwater sensor networks before time synchronization, estimation of the Doppler scaling factor at the receiver is actually a combination of the Doppler effect caused by sensor mobility and the clock skew between the transmitter and the receiver. In the sequel, we develop a model to establish the relationship between the joint Doppler effect and the individual Doppler effect caused by the sensor mobility and clock skew, and the resolvability of the sensor mobility and clock skew based on the estimated joint Doppler scaling factor.

We start with a scenario in which a message in the waveform $x_{AB}(t)$ is sent from sensor node A to sensor node B, with node A as the reference node. Denote a as the combined Doppler scaling factor of the waveform received at node B, and define v and θ as the speed and the clock skew of node B relative to node A, respectively. The Doppler scaling factor induced by the sensor mobility is thus $a_m \triangleq v/\eta$, where η is the sound speed in water.

Following the derivation in [31], the received waveform at node B can be formulated as the summation of signals arrived along multiple physical paths,

$$y_{AB}(t) = \sum_{p=1}^{N_p} A_p x_{AB}\big((1+a_m)t - \tau_p\big), \quad (7)$$

where N_p denotes the number of paths, and A_p and τ_p denote the amplitude and delay of the pth path, respectively.

We assume that both nodes operate at an identical sampling rate f_s, such that the transmitted waveform $x(t)$ is discretized as

$$x_{AB}[n] = x_{AB}(t) \mid_{t=n/f_s}. \quad (8)$$

However, given the clock skew θ, the sampling rate at node B is actually θf_s relative to node A, leading to discretized samples

$$y_{AB}[n] = y_{AB}(t) \mid_{t=n/\theta f_s} = \sum_{p=1}^{N_p} A_p x_{AB} \left[\underbrace{\frac{(1+a_m)}{\theta}}_{=1+a_{AB}} n - \tau_p \right]. \quad (9)$$

Based on (9), we find that the joint Doppler scaling in received samples is the multiplication of the scaling effects caused by sensor mobility and the clock skew individually. However, for the message $x_{BA}(t)$ sent from node B to node A, the received signal at node A is discretized as

$$y_{BA}[n] = y_{AB}(t) \mid_{t=n\theta/f_s} = \sum_{p=1}^{N_p} A_p x_{BA} \left[\underbrace{\theta(1+a_m)}_{=1+a_{BA}} n - \tau_p \right], \quad (10)$$

which shows that the joint Doppler scaling in received samples is the division of the above two individual effects. Given the different relationship between the joint Doppler scaling effect and the two individual Doppler effects of message transmissions in two different directions, parameters of sensor mobility a_m and the clock skew θ are therefore resolvable.

Based on the Doppler scaling factor a_m, the relative speed of the two sensor nodes can be directly obtained as

$$v = a_m \eta = \begin{cases} ((1+a_{AB})\theta - 1)\eta, & A \longrightarrow B \\ ((1+a_{BA})/\theta - 1)\eta, & B \longrightarrow A \end{cases} \quad (11)$$

which will be used in the DA-Sync protocol for speed estimation refinement in Section 4.

Note that due to the sensor mobility, a_m varies from transmission to transmission, and θ is unknown. Estimation of a_m and θ therefore cannot be obtained directly based on a single round-transmission in two directions. Since a_{AB} and a_{BA} can be gathered, in DA-Sync, at this time, θ is assigned with an initial value "1", named as initial skew denoted by $\dot{\theta}$. In doing so, relative speed v can be estimated with (11) and input as a known value for speed estimation refinement. Any errors introduced by this assumption can be corrected in the calibration phase.

4 Description of DA-Sync

4.1 Overview of DA-Sync

For time synchronization between pairs of clocks, most of the algorithms rely on estimating the clock offset and skew, which present the relation between the time measured by two different clocks. DA-Sync also yields to this pairwise synchronization approach. In DA-Sync, an ordinary node's clock aims to become synchronized with the reference clock of the reference node. Hence we have:

$$T = \theta * t + \beta, \quad (12)$$

where T stands for the measured time of the ordinary node; t is the reference time; θ and β are the relative clock skew and offset, respectively.

Overall, the time synchronization procedure of DA-Sync consists of five phases, namely, *data collection*, *velocity estimation refinement*, *propagation delay estimation*, *linear regression*, and *calibration*. In Phase I, with two-way synchronization reference message exchanges, the ordinary node collects time stamps for receiving and sending reference messages as well as its moving velocity relative to the reference node. For simplicity, we assume that the Doppler effect caused by node mobility can be successfully mitigated via a resampling operation during the physical-layer data decoding [23], so that all the synchronization messages can be correctly deciphered. In Phase II, by leveraging the Kalman Filter, the velocities obtained in the physical layer through Doppler shifts are refined. During Phase III, the ordinary node calculates the propagation delays based on the time and speed information collected in Phase I and Phase II. In Phase IV, the ordinary node organizes and performs the first linear regression with ordinary least-squares estimation (OLSE) over a set of time stamps as well as corresponding propagation delays to estimate the draft clock skew and offset. In Phase V, to further improve the synchronization accuracy, the ordinary node updates the initial skew with the calculated value in Phase III and Phase IV, and calculates the weight for each exchanged message. After that, it reperforms delay estimation and reruns linear regression with weighted least-squares estimation (WLSE) to obtain the final clock skew and offset.

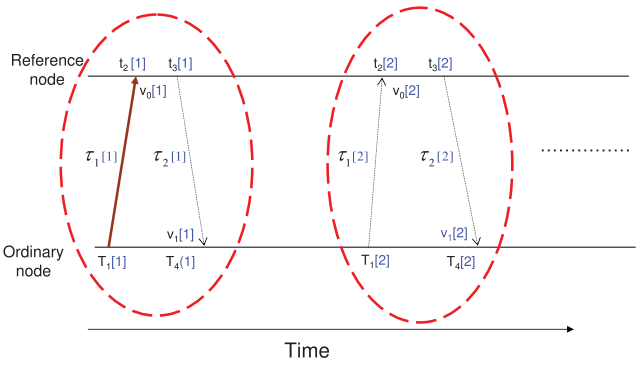


Fig. 4. Message exchange process.

4.2 Phase I: Data Collection

In the beginning, an ordinary node initiates message exchanges by sending a "Sync-Req" message to a neighboring reference node. The ordinary node records the sending time stamp T_1, obtained at the MAC layer, right before the message leaves. Upon receiving the Sync-Req message, the reference node estimates and records the ordinary node's relative moving velocity v_0 with Doppler shifts as specified in Section 3. Meanwhile, it marks its local time as t_2. Then, after a time interval t_r (waiting for the hardware sending-receiving transition and avoiding collisions), the reference node sends back a "Sync-Res" message which contains t_2, v_0 and its sending time t_3. When receiving the Sync-Res message, the ordinary node records its receiving time T_4 and its relative moving velocity to the reference node, v_1. Fig. 4 shows the message exchange process between the ordinary node and the reference node.

Depending on the accuracy requirement from the application, the above message exchange process can run multiple times. After a couple of rounds of message exchange, the ordinary node collects a set of time stamps consisting of T_1, t_2, t_3, and T_4 and relative velocities consisting of v_0 and v_1.

4.3 Phase II: Velocity Estimation Refinement

Considering that multiple message exchanges occur in the synchronization process, estimation of the relative speed between the reference node and the ordinary node can be improved by incorporating the estimates obtained during the previous message exchanges.

Assuming the relative motion between the reference node and the ordinary node follows the first-order kinematic model, we have the dynamic equation

$$\mathbf{x}(k+1) = \mathbf{F}(k)\mathbf{x}(k) + \mathbf{\Gamma}(k)w(k), \tag{13}$$

where $w(k)$ denotes the discrete-time process noise, which is supposed to follow a Gaussian distribution $w(k) \sim \mathcal{N}(0, R_w)$,[1] and

$$\mathbf{x}(k+1) = [\, v(k+1) \quad \alpha(k+1)\,]^T$$
$$\mathbf{F}(k) = \begin{pmatrix} 1 & \Delta T(k) \\ 0 & 1 \end{pmatrix}, \quad \mathbf{\Gamma}(k) = \begin{pmatrix} \Delta T^2(k)/2 \\ \Delta T(k) \end{pmatrix},$$

in which $v(k+1)$ and $\alpha(k+1)$ denote velocity and acceleration, respectively. $\Delta T(k)$ can be determined as follows:

$$\Delta T(k) = \begin{cases} \dfrac{\tau_1[i] + \tau_2[i]}{2} + t_3[i] - t_2[i], & (k = 2i-1) \\ \dfrac{\tau_2[i-1] + \tau_1[i]}{2} + t_2[i] - t_3[i-1], & (k = 2i), \end{cases}$$

where the estimates $v^e(2i-1)$ and $v^e(2i)$ are corresponding to $v_0[i]$ and $v_1[i]$ at the ith message exchange, respectively.

The measurement equation can be formulated as

$$z(k) = \mathbf{H}\mathbf{x}(k) + n(k), \tag{14}$$

where, $H = [\,1 \quad 0\,]$, $n(k)$ denotes the measurement noise $n(k) \sim \mathcal{N}(0, R_n)$, and $z(k)$ is the velocity measurement corresponding to the estimated velocity obtained in Section 3,

$$z(k) = \begin{cases} v_0^m[i], & (k = 2i-1) \\ v_1^m[i], & (k = 2i) \end{cases}.$$

The estimation of the state $x(k+1)$ based on the measurement $z(k)$ can be obtained with the Kalman filter. We adopt a two-point differencing procedure for initialization

$$v^e(1) = v_0^m[1],$$
$$\alpha^e(1) = \frac{v_1^m[1] - v_0^m[1]}{\Delta T(1)},$$

and the initial covariance matrix is

$$\mathbf{P}(1|1) = \begin{pmatrix} R_n & R_n/\Delta T(1) \\ R_n/\Delta T(1) & 2R_n/(\Delta T(1))^2 \end{pmatrix}. \tag{15}$$

The covariance of process noise can be calculated as

$$\mathbf{Q}(k) = \mathrm{E}[\mathbf{\Gamma}(k)w(k)w(k)^T\mathbf{\Gamma}(k)^T] = \mathbf{\Gamma}(k)R_w\mathbf{\Gamma}(k)^T. \tag{16}$$

Fig. 5 presents how to estimate the state $\hat{\mathbf{x}}(k+1 \mid k+1)$ following the procedure of the Kalman filter, which is an optimal MMSE state estimator under the Gaussian assumption of both the process noise and measurement noise. In the Kalman filter, filter gain is decided by two parameters, i.e., R_w and R_n, which somehow reflect the "weights" on measured velocity and predicted velocity. Our simulation results (see Section 6) show that with the help of the Kalman filter, the synchronization accuracy can be greatly improved.

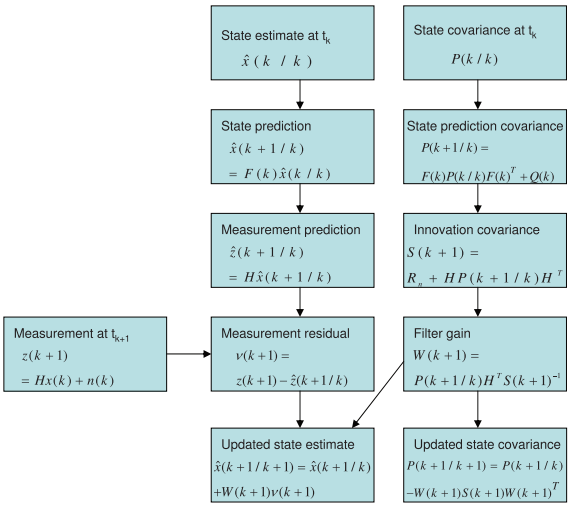


Fig. 5. Estimation with Kalman filter.

Remark 1. Note that the model above assumes a first-order kinematic model with a constant acceleration between consecutive sampling times. In fact, the relative movement between reference node and ordinary node can be more complicated. To improve the estimation accuracy, a higher-order kinematic model can be used by incorporating the acceleration rate into the state vector.

$$\mathbf{x}(k+1) = \left[v(k+1)\ \alpha(k+1)\ \dot{\alpha}(k+1)\right]^T \tag{17}$$

With matrices in (13) and (14) modified as

$$\mathbf{F}(k) = \begin{pmatrix} 1 & \Delta T(k) & T^2(k)/2 \\ 0 & 1 & \Delta T(k) \\ 0 & 0 & 1 \end{pmatrix},$$
$$\mathbf{\Gamma}(k) = \begin{pmatrix} \Delta T^2(k)/2 \\ \Delta T(k) \\ 1 \end{pmatrix} \quad \mathbf{H} = (\,1 \quad 0 \quad 0\,), \tag{18}$$

where $\dot{\alpha}(k+1)$ denotes the acceleration rate. Following the same procedure as introduced above, the relative speed can be estimated.

4.4 Phase III: Propagation-Delay Estimation

Phase III aims to estimate the long and dynamic propagation delays. In DA-Sync, two propagation delays need to be estimated, i.e., τ_1 and τ_2. In terms of delay estimation, many other time synchronization protocols take half of the two-way round trip time as one-way propagation delay, which means that τ_1 is equivalent to τ_2. In DA-Sync, by leveraging relative moving velocities obtained from the physical layer, we can differentiate τ_1 and τ_2, which is required in mobile scenarios.

Regarding the velocity, in DA-Sync, only one dimension is considered. That is because, to estimate the propagation

1. Note that currently there is no specific research on the sensor node mobility variations. According to the law of large numbers (LLN), in this work we simply assume that both the process noise and the measurement noise follow Gaussian distributions. For a system with non-Gaussian noise, more advanced filtering algorithms, such as the bootstrap filter [16] and the particle filter [3], can be used.

delay, only the relative velocity is demanded. The individual moving pattern of sensor nodes does not affect the propagation delay between them. Thus, in DA-Sync, for the ordinary node, it essentially either moves toward to the reference node or moves far away from the reference node. The amazing thing is that the velocity estimated by utilizing the Doppler scaling factor is exactly what is needed, i.e., the relative velocity. Therefore, the velocities obtained from the physical layer can be directly applied in DA-Sync. The following steps show how propagation delays τ_1 and τ_2 are estimated in each message exchange process.

Considering the round trip of the message exchange process and incorporating (12), we obtain:

$$\begin{cases} \tau_1 + \tau_2 = (T_4 - T_1)/\theta + t_2 - t_3 \\ \tau_1 + \Delta\tau = \tau_2, \end{cases} \tag{19}$$

where Δd stands for the relative moving distance during t_2 to T_4. In DA-Sync, we assume that the relative velocity between the ordinary node and the reference node changes linearly, and the propagation speed of acoustic signal η keeps constant as $1,500$ m/s. Based on these assumptions, we estimate Δd as

$$\Delta d = v_0^e(t_3 + \tau_2 - t_2) + \frac{1}{2}\alpha^e(t_3 + \tau_2 - t_2)^2. \tag{20}$$

To make (20) solvable, at this stage, we first ignore the relative motion between the two nodes during the propagation time τ_2 in (20), which turns (20) to (21):

$$\Delta d = v_0^e(t_3 - t_2) + \frac{1}{2}\alpha^e(t_3 - t_2)^2. \tag{21}$$

Later in Phase V, the error introduced by this inappropriate value assignment can be calibrated.

We define:

$$\begin{cases} t_3 - t_2 &= t_r \\ T_4 - T_1 &= \nu \\ \frac{1}{2}\alpha^e {t_r}^2 &= \mu \\ t_r + \tau_2 &= \Delta t. \end{cases} \tag{22}$$

Being aware of $\Delta\tau$ and combining (19), (21), and (22), we obtain the propagation delays of τ_1 and τ_2 as follows:

$$\begin{cases} \tau_1 = \dfrac{\nu\eta - \theta t_r(\eta + v_0^e) - \mu}{2\theta\eta} \\ \tau_2 = \dfrac{\nu\eta + \theta t_r(v_0^e - \eta) + \mu}{2\theta\eta}. \end{cases} \tag{23}$$

Note that in (23), to calculate τ_1 and τ_2, besides the measured time stamps and velocities, a skew θ is also needed. Nevertheless, the skew is the final goal of time synchronization which is an unknown parameter. In DA-Sync, the skew is assigned with an initial value "1", named as initial skew denoted by $\dot{\theta}$. By doing so, τ_1 and τ_2 can be computed and taken as constants to assist the linear regression. And the errors introduced by $\dot{\theta}$ can be calibrated in Phase V.

4.5 Phase IV: Linear Regression

During Phase IV, the ordinary node performs linear regression over the data points in (24) to estimate the draft clock skew θ and offset β, where $[i]$ determines the index of the message exchange process.

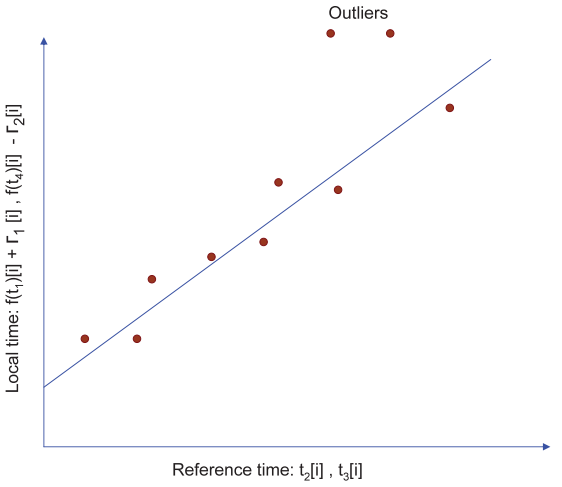


Fig. 6. Linear Regression.

$$(t_2[i], T_1[i] + \tau_1[i]) \text{ or } (t_3[i], T_4[i] - \tau_2[i]) \tag{24}$$

With linear regression, the skew θ and the offset β can be computed as shown in Fig. 6. Generally speaking, there are several ways to perform linear regression. ordinary least-squares estimation (OLSE) is one of the most popular approaches. Essentially, it makes each sample point have the same opportunity to affect the estimating θ and β. However, DA-Sync is based on an assumption that the relative velocity changes linearly, which is not always the case in the real underwater environment. Therefore, in Fig. 6, the variance of residual error should not be constant for all values of the independents, and it is possible that in some extremely bad cases, sample points are far away from what are expected. To reduce the impacts from those outliers, besides OLSE, WLSE is also adopted.

WLSE is an approach for correcting the problem of heteroskedasticity by log-likelihood estimation of a weight that adjusts errors of prediction. It performs better than OLSE because it introduces an extra functional coefficient "weight" to module the power of each sample data. The following steps show how WLSE works.

Above all, the "sum of squared deviations" is:

$$\sigma(\beta, \theta) = \sum_{i=1}^{m} \omega_i \big(T_i - \beta - \theta t_i\big)^2, \tag{25}$$

where T_i stands for $T_1[i] + \tau_1[i]$ or $T_4[i] - \tau_2[i]$ and t_i represents $t_2[i]$ or $t_3[i]$. For linear regression, WLSE will find the right values of $\hat{\theta}$ and $\hat{\beta}$ minimizing $\sigma(\beta, \theta)$:

$$\begin{aligned} \sigma(\hat{\beta}, \hat{\theta}) &= \sum_{i=1}^{m} \omega_i \big(T_i - \hat{\beta} - \hat{\theta} t_i\big)^2 \\ &= \arg\min \sum_{i=1}^{m} \omega_i \big(T_i - \beta - \theta t_i\big)^2 \end{aligned}, \tag{26}$$

where $\sigma(\hat{\beta}, \hat{\theta})$ stands for the "errors" to be minimized. This is achieved by using standard techniques from calculus, namely the property that a quadratic formula reaches its minimum value when its derivatives vanish. Taking the

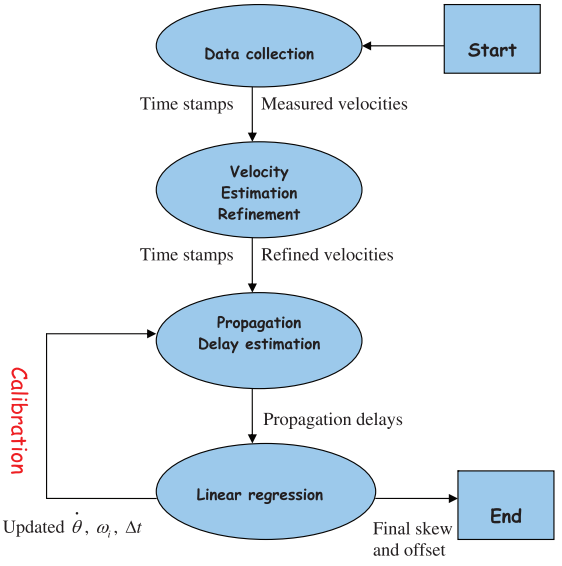


Fig. 7. Calibration procedure.

derivative of $\sigma(\hat{\beta},\hat{\theta})$ with respect to θ and β and setting them to zero gives the following set of equations:

$$\begin{cases} \dfrac{\partial \sigma}{\partial \beta}\mid_{\beta=\hat{\beta}} = -2\sum_{i=1}^{m}\omega_i\left(T_i-\hat{\beta}-\hat{\theta}t_i\right)=0 \\ \dfrac{\partial \sigma}{\partial \theta}\mid_{\theta=\hat{\theta}} = -2\sum_{i=1}^{m}\omega_i\left(T_i-\hat{\beta}-\hat{\theta}t_i\right)t_i=0 \end{cases} \tag{27}$$

Solving (27) gives the least-squares estimates of θ and β as

$$\begin{cases} \hat{\beta}_\omega &= \overline{T_\omega}-\hat{\theta}_\omega\overline{t_\omega} \\ \hat{\theta}_\omega &= \dfrac{\sum_{i=1}^{m}\omega_i(t_i-\overline{t_\omega})(T_i-\overline{T_\omega})}{\sum_{i=1}^{m}\omega_i(t_i-\overline{t_\omega})^2}, \end{cases} \tag{28}$$

where $\overline{T_\omega}$ denotes the weighted means of T_i, and $\overline{t_\omega}$ stands for weighted means of t_i:

$$\begin{cases} \overline{T_\omega}=\dfrac{1}{\sum\omega_i}\sum_{i=1}^{m}\omega_i T_i \\ \overline{t_\omega}=\dfrac{1}{\sum\omega_i}\sum_{i=1}^{m}\omega_i t_i. \end{cases} \tag{29}$$

In general, the weight ω_i assigned to the ith observation is a function of the variance of this observation, denoted as ${\sigma_i}^2$. A straightforward way to define ω_i is as follows:

$$\omega_i = 1/\sigma_i, \tag{30}$$

σ_i, however, is an unknown parameter which is decided by θ and β. Under this situation, in DA-Sync, for the first linear regression, each ω_i is assigned a same value "1", which turns WLSE to OLSE. After the first linear regression, draft θ and β will be obtained. σ_i can then also be computed. Next in the calibration phase, WLSE can be used to give each observation proper power to affect the estimated final skew and offset.

4.6 Phase V: Calibration

In Phase V, the ordinary node carries out the calibration process to correct some parameters and recalculate the clock skew and offset, as shown in Fig. 7. The calibration process will run iteratively until it reaches the stopping criteria, which is either when the number of runs reaches the maximum loop number 10 or the difference between estimated skew in this run and last run is less than 100 ppm.

There are three parameters to be adjusted, initial skew $\dot{\theta}$, weight ω_i, and the time period Δt. When inputting the measured relative speed and calculating the propagation delays in Phase III, the skew is assigned as an initial value "1", which is not true. In the calibration phase, the initial skew $\dot{\theta}$ is updated with the estimated skew, and the measured speed is updated and refined by reperforming the Kalman filter. In terms of Δt, since the first estimated τ_2 has been obtained, in (21), t_r can be replaced with Δt, which changes (21) back to (20). Then with updated skew and (21), the propagation delays τ_1 and τ_2 are recalculated for each message exchange process. According to ω_i, with the estimated skew and offset, the ordinary node computes the "weight" for each synchronization message. Finally, with all of the updated propagation delays and unchanged time stamps, linear regression is reperformed with WLSE to compute the final clock skew and offset.

Simulation results show that via a calibration process, without any extra message exchanges, the skew can be significantly corrected [10].

4.7 Discussions

Technically, any time-synchronization scheme should be accurate and energy efficient. DA-Sync takes sensor node's mobility into consideration throughout the synchronization procedure. Instead of estimating the propagation delay with $RTT/2$, DA-Sync effectively utilizes physical layer characteristics, i.e., a joint Doppler scaling effect considering both sensor node mobility and clock skew, to gather the sensor node's relative moving speed. By leveraging the speed, DA-Sync can accurately differentiate the propagation delay τ_1 and τ_2. Consequently, the accuracy of propagation-delay estimation as well as the time synchronization will be significantly enhanced.

Second, in terms of energy efficiency, it is directly affected by exchange messages overhead and computational cost. However, in reality, the energy cost in computation can be ignored compared to sending and receiving of message in UWSNs. (For instance, in our equipment, the power consumption of the modem which is for sending and receiving acoustic signals is around 20 Watt. However, the power consumption of the microprocessor, which is running the program, is only about 1 Watt.) For message overhead, it is directly affected by synchronization accuracy. Since time-synchronization accuracy determines the frequency of resynchronization, when the tolerated clock error is fixed, a high accuracy protocol requires less resynchronization, which means less message overhead. Additionally, by differentiating τ_1 and τ_2, in each message exchange process, two sample points can be collected for linear regression, which means DA-Sync can achieve same accuracy with less message overhead. This also greatly helps to increase the energy efficiency of DA-Sync.

Remark 2. Although DA-Sync is a pairwise synchronization scheme, it can be easily extended to the multi-hop

network, and provides a network-wide synchronization service. Basically, only two phases are needed. In first phase, one sensor node needs to be selected as a network-wide reference node, and a hierarchical structure should be established with the reference node as the root. In the second phase, a pairwised DA-Sync is applied for synchronization along edges of this structure to establish a global timescale throughout the network. Finally, all the sensor nodes will get synchronized with the reference node and the network-wide synchronization is achieved. Note that, although a detailed investigation on the multihop synchronization is beyond this work, according to [14], most likely, the synchronization error will not increase linearly with the number of hops. Instead, it is expected to converge after certain number of hops as the synchronization error between any two sensor nodes will probabilistically take different values from the normal distribution. The randomness in the sign as well as the magnitude of the synchronization error and drift prevents the error from blowing up.

5 Error Analysis of Delay Estimation

In this section, we analyze the influence of possible errors on the propagation-delay estimation. To do this, it is crucial to understand that all observations of physical quantities are subject to uncertainties. In the context of time synchronization, among all uncertainties including sending time, accessing time, transmission time, propagation time, reception time, interrupt handling, encoding time, decoding time, and byte alignment time [21], [22], the long and variable propagation delay is dominant due to the low propagation speed of acoustic signals [41]. Therefore, the propagation-delay estimation is the main source of the synchronization error.

In DA-Sync, there are four sources of errors on the propagation-delay estimation, i.e., errors from inappropriate initial or the first estimated draft skew $\dot{\theta}$, inappropriate Δt or errors on estimated velocity and acceleration. Among these sources, errors on acceleration are based on the error of velocity. As we assume a sensor node's moving velocity changes linearly, the error of acceleration estimation is skipped. As for the errors on inappropriate Δt and errors on initial skew $\dot{\theta}$, both can be corrected in the calibration process. Using the Kalman filter, the accuracy on the velocity estimation can also be improved.

During the process of velocity refinement, in the state covariance $\mathbf{P}(k+1 \mid k+1)$, $\mathbf{p}(k+1 \mid k+1)_{00}$ actually represents the errors on the velocity v_0^e, and $\mathbf{P}(k+1 \mid k+1)_{11}$ stands for the error on the acceleration α^e. According to the definition of Δt, the error on Δt is exactly the error on τ_2. If we use $\Delta\theta$ to represent the error on initial or draft skew $\dot{\theta}$, we have

$$\begin{cases} v_0 = v_0^e + \mathbf{p}(k+1 \mid k+1)_{00} \\ \theta = \dot{\theta} + \Delta\theta \\ \Delta t = t_r + \tau_2 + \Delta\tau_2. \end{cases} \tag{31}$$

Considering $\tau_1 = f_1(\dot{\theta}, v_0^e, \Delta t)$, $\tau_2 = f_2(\dot{\theta}, v_0^e, \Delta t)$, and taking advantage of error propagation rules [7], [42], we can obtain:

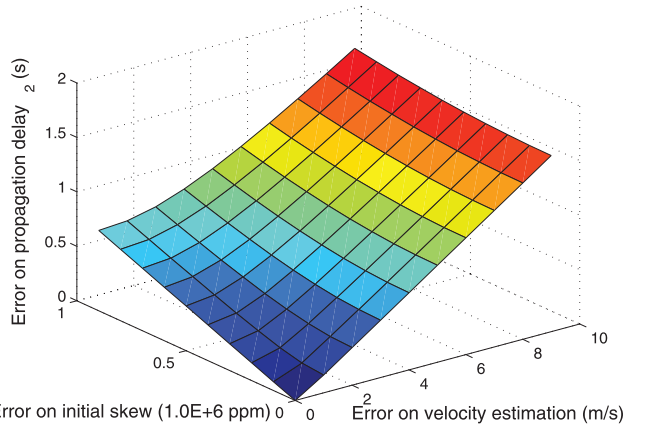


Fig. 8. The effects of initial skew errors and velocity estimation errors in the propagation-delay estimation.

$$\begin{aligned} \Delta\tau_1 &\approx \sqrt{\left(\frac{\partial f_1}{\partial\theta}\Delta\theta\right)^2 + \left(\frac{\partial f_1}{\partial v_0^e}\Delta v_0^e\right)^2 + \left(\frac{\partial f_1}{\partial\Delta t}\Delta\tau_2\right)^2}, \\ \Delta\tau_2 &\approx \sqrt{\left(\frac{\partial f_2}{\partial\theta}\Delta\theta\right)^2 + \left(\frac{\partial f_2}{\partial v_0^e}\Delta v_0^e\right)^2 + \left(\frac{\partial f_2}{\partial\Delta t}\Delta\tau_2\right)^2}. \end{aligned} \tag{32}$$

Taking τ_2 as an instance and giving fixed sample values for parameters v_0^e, α^e, t_r, $\dot{\theta}$ and T_1, t_2, t_3, T_4, the relationship among the various errors are shown in Fig. 8. τ_1 will have a similar situation.

6 Performance Evaluation

DA-Sync is evaluated in a self-designed simulator developed with C++ and runs in Linux system. In the simulator, all actions are managed by an event scheduler, and each node is an object of the sensor node class, maintains its own local time and position, and moves following a predefined mobility pattern. By assuming that each reference packet has the same size, the transmission delay is fixed. The propagation delay is calculated based on the distance between the reference node and ordinary node, with an assumption of a constant propagation speed. During synchronization, each sensor node records its local time before sending message or upon receiving message. Since DA-Sync aims to provide a pairwise synchronization method, we randomly deploy 30 underwater sensor nodes in a $1{,}000\ \mathrm{m} \times 1{,}000\ \mathrm{m} \times 1{,}000\ \mathrm{m}$ region, and make any of the two sensor nodes communicate with each other, and neglect the collision during synchronization. We randomly pick one sample node as the ordinary node which aims to become synchronized and select another node as the reference node.

6.1 Simulation Settings

Without loss of generality, the inherent skew of the ordinary node is predefined as $50\ \mathrm{ppm}$, and it remains unchanged during the time synchronization procedure. The clock offset of the ordinary node is initialized as $0.0008\ \mathrm{s}$. In addition, the response time t_r is fixed as $1\ \mathrm{s}$, and the propagation speed is $1500\ \mathrm{m/s}$. Regarding the mobility behavior of every node, we consider a linear kinematic model as follows:

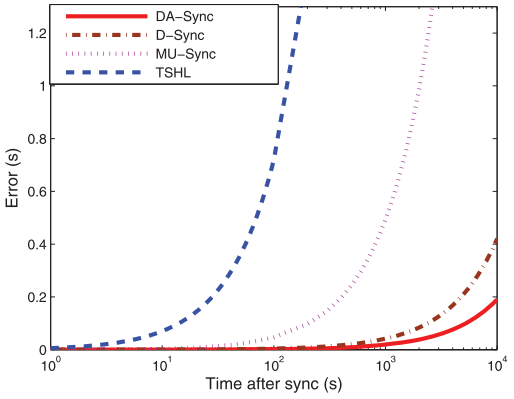


Fig. 9. Error versus time after sync.

$$\begin{cases} V_x &= k_1 t + k_4 \\ V_y &= k_2 t + k_5, \end{cases} \tag{33}$$

where V_x determines the speed at the X axis, and V_y stands for the speed at the Y axis. k_1 and k_2 are parameters which are closely related to environmental factors such as tides and bathymetry. k_4 and k_5 are used to simulate some random factors. In our simulations, we choose k_1 and k_2 from random variables following a normal distribution with π as the mean value and 0.1π as the variance. k_4 and k_5 are chosen from random variables following a normal distribution with 0.1 as the mean value and 0.01 as the variance. In our simulations, whenever the speeds reach maximal value which is 5 m/s, k_1 and k_2 will change to become negative. On the contrary, when the speeds reach zero, k_1 and k_2 will change to become positive.

6.2 Results and Analysis

In this section, we evaluate the performance of DA-Sync. Note that, different algorithms have different sync message (including request and response messages) packet sizes since they need to carry different amounts of information. In this work, sync message packet size for DA-Sync and D-sync is 40 bytes, and MU-Sync and TSHL is 32 bytes.

6.2.1 Performance on Accuracy

Fig. 9 illustrates how error grows after time synchronization completes with different schemes, namely, DA-Sync, D-Sync, MU-Sync, and TSHL. As discussed earlier, when time goes by, the skew causes increasing errors. Therefore, this comparison demonstrates the accuracies on the skew that various schemes can achieve. From the figure, we can observe that DA-Sync performs much better than both MU-Sync and TSHL. The reason for the poor performance of TSHL is because TSHL is mainly designed for static networks, and it does not consider the mobility of sensor nodes. Although it applies linear regression to estimate and compensate both skew and offset, it treats propagation delay as a constant during the time synchronization period. This clearly causes significant errors on the propagation-delay estimation. For MU-Sync, although it claims to be able to handle sensor node mobility by running two trials of linear regression, it uses half of the round trip time to compute one-way propagation delay, which introduces certain errors, especially when nodes move fast or have long response time to the request message.

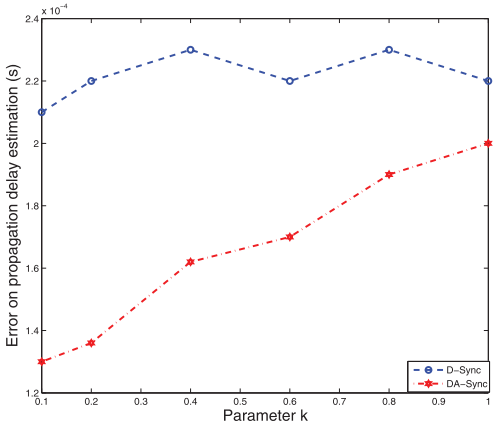


Fig. 10. Error on propagation-delay estimation.

Compared to DA-Sync, D-Sync ignores the effect of the clock skew when utilizing the Doppler effect, which reduces the accuracy of the Doppler shift estimation and affects the accuracy of propagation-delay estimation. Furthermore, D-Sync completely trusts the measured speed with Doppler shift, which also leads error on propagation-delay estimation. Fig. 10 demonstrates the propagation-delay estimation errors of DA-Sync and D-Sync. Those errors will be propagated to the linear regression process and affect the synchronization accuracy. DA-Sync exploits a joint Doppler effect caused by both the sensor mobility and clock skew to differentiate τ_1 and τ_2. The speed is significantly refined with Kalman filtering. Moreover, calibration process is adopted to further calibrate estimated clock skew and offset. These techniques make DA-Sync more accurate.

6.2.2 Effect of Number of Messages

Fig. 11 shows how message overhead affects the accuracy of DA-Sync. Theoretically, from the linear regression perspective, the more exchanged messages are involved, the more data points can be collected, and the more precise the estimated skew and offset are. In this set of simulations, we run DA-Sync with different number of messages, ranging from 8 to 40. As expected, the simulation results reveal that when more sample messages are involved in the time synchronization process, DA-Sync can achieve better accuracy. However, more messages also means more energy

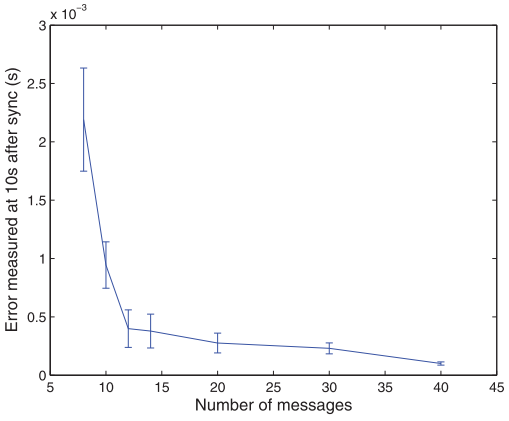


Fig. 11. Error after 10 s versus number of messages.

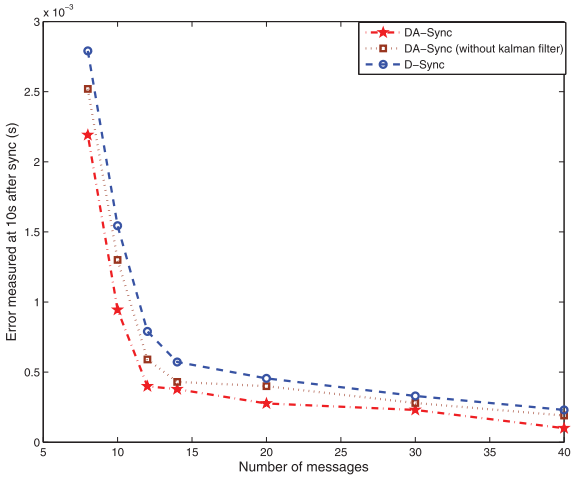


Fig. 12. Error after 10 s versus number of messages.

consumption. Due to the energy constraints of UWSNs, the message overhead for time synchronization should be strictly controlled. In other words, there is a tradeoff between time synchronization accuracy and energy efficiency.

To compare with D-Sync fairly, a new Fig. 12 is provided. It shows that to achieve the same accuracy in one run of time synchronization, the D-Sync always needs more exchanged messages than DA-Sync, which means more energy consumption. And to further compare them, we implement the DA-Sync without Kalman filter, that means exactly the same energy consumption with D-Sync, and it turns out that it is still better than D-Sync since it accounts the skew when estimate the doppler shift.

6.2.3 Performance on Energy Efficiency

Fig. 13 compares various synchronization schemes in terms of energy efficiency. Assuming that the computational cost is negligible, the energy efficiency is defined as follows:

$$\rho = \frac{\vartheta}{\kappa \varrho \, \gamma} \tag{34}$$

where κ stands for the number of resynchronizations which are needed in the period of ϑ seconds to keep the clock error below certain value. ϱ represents the number of messages for each synchronization process and γ is packet size. In the simulation, we fix ϱ as 24 messages, and ϑ as 10^6 seconds for

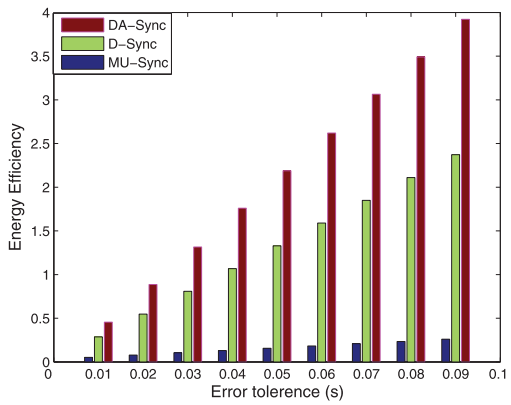


Fig. 13. Energy efficiency with varying error tolerance.

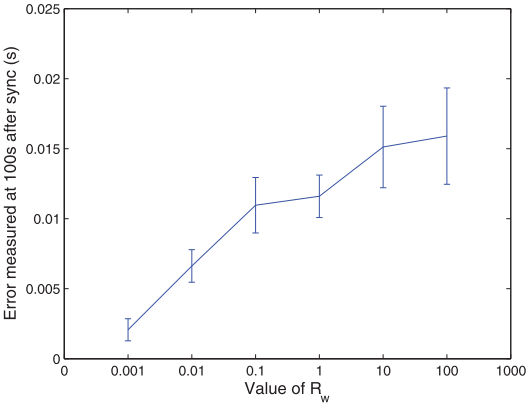


Fig. 14. Error after 100s versus noise.

each algorithm. γ is set to 40 Bytes for DA-Sync and D-Sync and 32 Bytes for MU-Sync. By doing so, given certain tolerated clock error, ρ can be estimated by calculating κ. Note that, since clock skew causes increasing error with time progresses, the frequency of resynchronization is determined by skew estimation accuracy, which is also part of time synchronization accuracy. Such that a higher accuracy time-synchronization scheme will have a smaller κ. The result shows that, for all the schemes, as tolerated error is relaxed, ρ is increasing, because of the decreasing of κ. DA-Sync is the best among all schemes.

6.2.4 Effect of Noise

To gain insights on the effect of Doppler speed estimation accuracy on synchronization performance, in this test we fix the variance of the measurement noise in (14) and increase the variance of the process noise in (13). Fig. 14 shows that the synchronization error increases as the process noise increases. This result agrees well with intuition.

6.2.5 Effect of Calibration

Fig. 15 demonstrates the effect of the calibration procedure, and is represented as the clock skew accuracy improved by calibration, which is the percentage of

$$\frac{\theta_{wc} - \theta}{\theta_{nc} - \theta}, \tag{35}$$

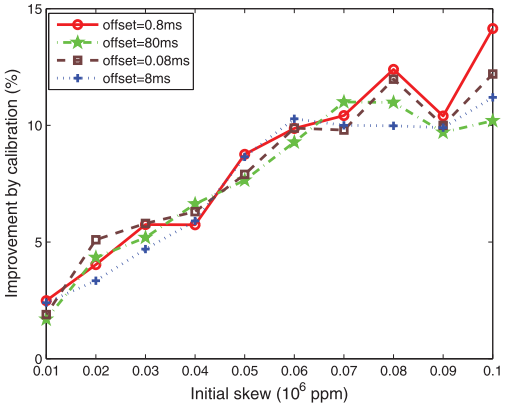
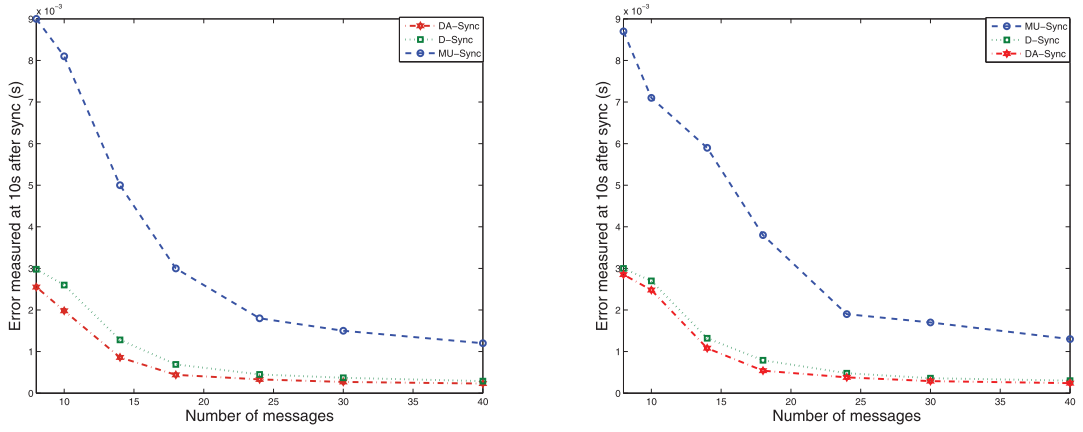


Fig. 15. Performance improvement with calibration.



(a) Kinematic model (b) Random walk

Fig. 16. Effect of mobility pattern.

where θ_{wc} and θ_{nc} stand for the estimated skew with and without calibration, respectively, and θ is the real skew. The calibration procedure of DA-Sync aims to make up the impact from the initial skew $\dot{\theta}$, the weight ω_i, and the time period Δt. The results show that the offset has no impact on the calibration process, and with the initial skew further from the real skew, the calibration process becomes more efficient.

6.2.6 Effect of Mobility Patterns

Figs. 16a and 16b compare DA-Sync with the existing protocols under two different mobility patterns. Fig. 16b corresponds to a random walk model with the speed region from -2 m/s to 2 m/s. Fig. 16a corresponds to a second-order kinematic model in (17) and (18), and kinematic models are often used to characterize the node mobility behavior in underwater environments [4]. The node velocities used in our simulations are

$$\begin{cases} V_x = k_1 \lambda v \sin(kk_2 x)\cos(kk_3 y) + k_1 \lambda \cos(2kk_1 t) + k_4 \\ V_y = -\lambda v \cos(kk_2 x)\sin(kk_3 y) + k_5, \end{cases} \quad (36)$$

where V_x and V_y denote the velocity along the X- and Y-axes, respectively. $k_1, k_2, k_3, \lambda,$ and v are the variables, which are closely related to environment factors such as tides and bathymetry. These parameters change with different environments. Random variables k_4 and k_5 simulate additional random physical factors. Coefficient k controls the turning speed. In our simulations, we assume they are all random variables which are subject to normal distribution. Table 2 shows the numerical values for the mean and variance of these variables. Those values are set according to empirical observations, which suggest that underwater objects may move at speeds of two to three knots (i.e., three to six kilometers per hour) in a typical underwater setting [11].

TABLE 2
Parameter Setting for Kinematic Model

Setting	Value	Setting	Value
k_1	$\mathcal{N}(\pi,0.1\pi)$	k_2	$\mathcal{N}(\pi,\pi)$
k_3	$\mathcal{N}(2\pi,0.2\pi)$	k_4	$\mathcal{N}(1,0.1)$
k_5	$\mathcal{N}(1,0.1)$	λ	$\mathcal{N}(0.3,0.03)$
v	$\mathcal{N}(1,0.1)$m/s	k	0.1

The results clearly show that with different mobility patterns, DA-Sync still outperforms D-Sync and MU-Sync in terms of synchronization accuracy. That is because for DA-Sync, individual moving patterns of sensor nodes will only affect the synchronization accuracy when it affects the 1D relative movement. Even if the relative movement between reference node and ordinary node is nonlinear, the higher-order kinematic model in (17) and (18) can be used to better track the 1D relative speed. Comparing Figs. 16a and 16b, one can see that relative to the random-walk mobility pattern, higher synchronization accuracy can be achieved under the kinematic mobility pattern. This result is understandable as the kinematic mobility pattern can be easily tracked by the Kalman filter, hence leading to a lower speed estimation error.

7 Conclusions and Future Work

In this paper, we presented DA-Sync, a novel time-synchronization scheme for mobile UWSNs. DA-Sync is a fundamental cross-layer time-synchronization scheme, which leverages Doppler effects and employs the Kalman filter to improve the accuracy of propagation-delay estimation in time synchronization. Our simulation results showed that this new approach can achieve high accuracy with low message overhead.

In the future, we plan to implement DA-Sync in underwater testbeds and explore its performance in real underwater environments.

Acknowledgments

This work was supported in part by the US National Science Foundation under CAREER grant nos. 0644190, 0709005, 0721834, 0821597, and 1018422 and the US Office of Navy Research under YIP grant no. N000140810864.

REFERENCES

[1] I.F. Akyildiz, D. Pompili, and T. Melodia, "Challenges for Efficient Communication in Underwater Acoustic Sensor Networks," *ACM SIGBED Rev.*, vol. 1, no. 1, pp. 3-8, July 2004.

[2] I.F. Akyildiz, D. Pompili, and T. Melodia, "Underwater Acoustic Sensor Networks: Research Challenges," *Ad Hoc Networks*, vol. 3, no. 3, pp. 257-279, Mar. 2005.

[3] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," vol. 50, no. 2, pp. 174-188, Feb. 2002.

[4] A.C. Bagtzoglou and N. A., "Chaotic Behavior and Pollution Dispersion Characteristics in Engineered Tidal Embayments: A Numerical Investigation," *J. Am. Water Resources Assoc.*, vol. 43, pp. 207-219, 2007.

[5] P. Bahl and V.N. Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System," *Proc. IEEE INFOCOM*, pp. 775-784, Mar. 2000.

[6] N. Bodhi, H. Balakrishnan, E. Demaine, and S. Teller, "Mobile-Assisted Localization in Wireless Sensor Networks," *Proc. IEEE INFOCOM*, pp. 172-183, Mar. 2005.

[7] P. Bork, H. Grote, D. Notz, and M. Regler, *Data Analysis Techniques in High Energy Physics Experiments.* Cambridge Univ., 1993.

[8] V. Chandrasekhar, W.K. Seah, Y.S. Chaoo, and H.V. Ee, "Localization in Underwater Sensor Networks - Survey and Challenges," *Proc. ACM Int'l Conf. UnderWater Networks and Systems (WUWNet '06)*, pp. 33-40, Sept. 2006.

[9] N. Chirdchoo, W.S. Soh, and K.C. Chua, "Aloha-Based MAC Protocols with Collision Avoidance for Underwater Acoustic Networks," *Proc. IEEE INFOCOM*, May 2007.

[10] N. Chirdchoo, W.-S. Soh, and K.C. Chua, "MU-Sync: A Time Synchronization Protocol for Underwater Mobile Networks," *Proc. Third ACM Int'l Workshop Underwater Networks (WuWNet '08)*, Sept. 2008.

[11] J.-H. Cui, J. Kong, M. Gerla, and S. Zhou, "Challenges: Building Scalable Mobile Underwater Wireless Sensor Networks for Aquatic Applications," *IEEE Network*, vol. 20, no. 3, Special Issue on Wireless Sensor Networking, pp. 12-18, May/June 2006.

[12] J. Elson, L. Girod, and D. Estrin, "Fine-Grained Network Time Synchronization Using Reference Broadcasts," *Proc. Fifth Symp. Operating Systems Design and Implementation (OSDI '02)*, pp. 147-163, Dec. 2002.

[13] S. Ganeriwal, R. Kumar, S. Adlakha, and M. Srivastava, "Network-Wide Time Synchronization in Sensor Networks," technical report, Univ. of California, Los Angeles, Apr. 2002.

[14] S. Ganeriwal, R. Kumar, and M.B. Srivastava, "Timing-Sync Protocol for Sensor Networks," *Proc. First Int'l ACM Conf. Embedded Networked Sensor Systems (SenSys)*, pp. 138-149, 2003.

[15] D.K. Goldenberg, A. Krishnamurthy, W.C. Maness, Y.R. Yang, A. Young, A.S. Morse, A. Savvides, and B.D.O. Anderson, "Network Localization in Partially Localizable Networks," *Proc. IEEE INFOCOM*, pp. 313-326, Mar. 2005.

[16] N. Gordon, D. Salmond, and A. Smith, "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation," *Proc. IEE Radar and Signal Processing*, vol. 140, no. 2, pp. 107-113, Apr. 1993.

[17] J. Heidemann, Y. Li, A. Syed, J. Wills, and W. Ye, "Research Challenges and Applications for Underwater Sensor Networking," *Proc. IEEE Wireless Comm. and Networking Conf. (WCNC)*, 2006.

[18] L. Hu and D. Evans, "Localization for Mobile Sensor Networks," *Proc. ACM MobiCom*, pp. 45-57, Sept. 2004.

[19] X. Ji and H. Zha, "Sensor Positioning in Wireless Ad-Hoc Sensor Networks Using Multidimensional Scaling," *Proc. IEEE INFOCOM*, pp. 2652-2661, Mar. 2004.

[20] K.D. Frampton, "Acoustic Self-Localization in a Distributed Sensor Network," *IEEE Sensors J.*, vol. 6, no. 1, pp. 166-172, Feb. 2006.

[21] H. Kopetz and W. Schwabl, "Clock Synchronization in Distributed Real-Time Systems," *IEEE Trans. Computers*, vol. 36, no. 8, pp. 933-939, Aug. 1987.

[22] H. Kopetz and W. Schwabl, "Global Time in Distributed Real-Time System," Technical Report 15/89, Technische Univ. Wien, 1989.

[23] B. Li, S. Zhou, M. Stojanovic, L. Freitag, and P. Willett, "Multicarrier Communication over Underwater Acoustic Channels with Nonuniform Doppler Shifts," *IEEE J. Oceanic Eng.*, vol. 33, no. 2, pp. 198-209, Apr. 2008.

[24] H. Lim and J.C. Hou, "Localization for Anisotropic Sensor Networks," *Proc. IEEE INFOCOM*, pp. 138-149, Mar. 2005.

[25] J. Liu, X. Han, M. Al-Bzoor, M. Zuba, J.-H. Cui, R.A. Ammar, and S. Rajasekaran, "PADP: Prediction Assisted Dynamic Surface Gateway Placement for Mobile Underwater Networks," *Proc. 17th IEEE Symp. Computers and Comm. (ISCC '12)*, July 2012.

[26] J. Liu, Z. Wang, Z. Peng, J.-H. Cui, and S. Zhou, "JSL: A Joint Solution for Localization and Time Synchronization Underwater Sensor Networks," *Proc. IEEE Comm. Soc. Conf. Sensor, Mesh, and Ad Hoc Comm. and Networks (SECON)*, June 2012.

[27] J. Liu, Z. Wang, Z. Peng, M. Zuba, J.-H. Cui, and S. Zhou, "TSMU: A Time Synchronization Scheme for Mobile Underwater Sensor Networks," *Proc. IEEE Global Telecomm. Conf. (GlobeCom '11)*, pp. 1-6, Dec. 2011.

[28] J. Liu, Z. Zhou, Z. Peng, J.-H. Cui, M. Zuba, and L. Fiondella, "Mobi-Sync: Efficient Time Synchronization for Mobile Underwater Sensor Networks," *IEEE Trans. Parallel and Distributed Systems (TPDS)*, Feb. 2012.

[29] F. Lu, D. Mirza, and C. Schurgers, "D-Sync: Doppler-Based Time Synchronization for Mobile Underwater Sensor Networks," *Proc. Fifth ACM Int'l Workshop UnderWater Networks (WUWNet '10)*, 2010.

[30] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, "The Flooding Time Synchronization Protocol," *Proc. Second Int'l ACM Conf. Embedded Networked Sensor Systems (SenSys)*, pp. 39-49, 2004.

[31] S. Mason, C.R. Berger, S. Zhou, and P. Willett, "Detection, Synchronization, and Doppler Scale Estimation with Multicarrier Waveforms in Underwater Acoustic Communication," *IEEE J. Selected Areas in Comm.*, Special Issue on Underwater Wireless Communications and Networks, vol. 26, no. 9, pp. 1638-1649, Dec. 2008.

[32] H. Mo, Z. Zhou, M. Zuba, Z. Peng, J.-H. Cui, and Y. Shu, "Practical Coding-Based Multi-Hop Reliable Data Transfer for Underwater Acoustic Networks," *Proc. IEEE Global Comm. Conf. (GlobeCom)*, Dec. 2012.

[33] A. Novikov and A.C. Bagtzoglou, "Hydrodynamic Model of the Lower Hudson River Estuarine System and Its Application for Water Quality Management," *Water Resource Management*, vol. 20, pp. 257-276, 2006.

[34] J. Partan, J. Kurose, and B.N. Levine, "A Survey of Practical Issues in Underwater Networks," *Proc. ACM Int'l Workshop UnderWater Networks (WUWNet)*, pp. 17-24, Sept. 2006.

[35] J. Partan, J. Kurose, and B.N. Levine, "A Survey of Practical Issues in Underwater Networks," *Proc. ACM Int'l Conf. UnderWater Networks and Systems (WUWNet '06)*, pp. 17-24, Sept. 2006.

[36] D. Pompili and T. Melodia, "Three-Dimensional Routing in Underwater Acoustic Sensor Networks," *Proc. Second ACM Int'l Workshop Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks*, pp. 214-221, 2005.

[37] L. Pu, Y. Luo, Y. Zhu, Z. Peng, S. Khare, J.-H. Cui, B. Liu, and L. Wang, "Impact of Real Modem Characteristics on Practical Underwater MAC Design," *Proc. IEEE OCEANS*, May 2012.

[38] S. Roy, P. Arabshahi, D. Rouseff, and W. Fox, "Wide Area Ocean Networks: Architecture and System Design Considerations," *Proc. ACM Int'l Conf. UnderWater Networks and Systems (WUWNet '06)*, pp. 25-32, Sept. 2006.

[39] S. Roy, T.M. Duman, V. McDonald, and J.G. Proakis, "High Rate Communication for Underwater Acoustic Channels Using Multiple Transmitters and Space-Time Coding: Receiver Structures and Experimental Results," *IEEE J. Oceanic Eng.*, vol. 32, no. 3, pp. 663-688, July 2007.

[40] F. Sivrikay and B. Yener, "Time Synchronization in Sensor Networks: A Survey," *IEEE Network*, vol. 18, no. 4, pp. 45-50, July/Aug., 2004.

[41] A. Syed and J. Heidemann, "Time Synchronization for High Latency Acoustic Networks," *Proc. IEEE INFOCOM*, 2006.

[42] Taylor and John, *An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurements.* Univ. Science Books, 1982.

[43] S. Tilak, V. Kolar, N.B. Abu-Ghazaleh, and K.-D. Kang, "Dynamic Localization Protocols for Mobile Sensor Networks," *Proc. IEEE Int'l Workshop Strategies for Energy Efficiency in Ad-Hoc and Sensor Networks*, Apr. 2005.

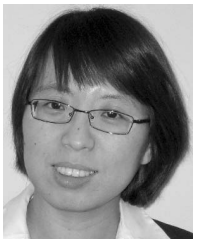
[44] Z. Zhou, J.-H. Cui, and S. Zhou, "Localization for Large-Scale Underwater Sensor Networks," *Proc. Sixth Int'l IFIP-TC6 Conf. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, May 2007.

Jun Liu received the BEng degree in computer science from the Wuhan University, China, in 2002. He is currently working toward the PhD degree and is a research assistant at the Underwater Sensor Network Lab, University of Connecticut, Storrs. His major research interests include time synchronization, localization, deployment for underwater acoustic networks, and also interested in operating system, cross layer design. He is a student member of the IEEE.

Zhaohui Wang received the BS degree in electrical engineering from the Beijing University of Chemical Technology in 2006 and the MSc degree in electrical engineering from the Institute of Acoustics, Chinese Academy of Sciences, Beijing, in 2009. She is currently working toward the PhD degree in the Department of Electrical and Computer Engineering at the University of Connecticut, Storrs. Her research interests include communications, signal processing, and detection, with a recent focus on multicarrier modulation algorithms and signal processing for underwater acoustic communications. She is a student member of the IEEE.

Michael Zuba received the BS degree in computer science and engineering from the University of Connecticut, Storrs, in 2010. He is currently working toward the PhD degree and is working as a research assistant in the Underwater Sensor Network Lab, University of Connecticut, Storrs. His main research interests include underwater acoustic networks, autonomous underwater vehicle networks, and security. He is a student member of the IEEE.

Zheng Peng received BS degrees in both computer science and control science from Zhejiang University, China, in 2002, and the MS degree in computer science from the University of Electrical Science and Technology of China in 2005. He is currently working toward the PhD degree and is working as a research assistant in the Underwater Sensor Network Lab, University of Connecticut, Storrs. His main research interests include underwater acoustic networks, including protocol design, operating systems, underwater sensor nodes, and testbeds. He is a student member of the IEEE.



Jun-Hong Cui received the BS degree in computer science from Jilin University, China, in 1995, the MS degree in computer engineering from the Chinese Academy of Sciences in 1998, and the PhD degree in computer science from the University of California, Los Angeles, in 2003. Currently, she is on the faculty of the Computer Science and Engineering Department at the University of Connecticut, Storrs. Her research interests include the design, modeling, and performance evaluation of networks and distributed systems. Recently, her research mainly focuses on exploiting the spatial properties in the modeling of network topology, network mobility, and group membership, scalable and efficient communication support in overlay and peer-to-peer networks, and algorithm and protocol design in underwater sensor networks. She is actively involved in the community as an organizer, a TPC member, and a reviewer for many conferences and journals. She is a guest editor for ACM *Mobile Computing and Communications Review* and *Elsevier Ad Hoc Networks*. She cofounded the first ACM International Workshop on UnderWater Networks (WUWNet 2006) and now serves as the WUWNet steering committee chair. She is a member of the IEEE, ACM, ACM SIGCOMM, ACM SIGMOBILE, IEEE Computer Society, and IEEE Communications Society.

Shengli Zhou received the BS and MSc degrees in electrical engineering and information science from the University of Science and Technology of China, Hefei, in 1995 and 1998, respectively. He received the PhD degree in electrical engineering from the University of Minnesota, Minneapolis, in 2002. He is currently the Charles H. Knapp associate professor in electrical engineering in the Department of Electrical and Computer Engineering, University of Connecticut, Storrs. His general research interests include wireless communications and signal processing. His recent focus has been on underwater acoustic communications and networking. He served as an associate editor for the *IEEE Transactions Wireless Communications* from 2005 to 2007 and the *IEEE Transactions Signal Processing* from 2008 to 2010. He is now an associate editor for the *IEEE Journal of Oceanic Engineering*. He received the 2007 ONR Young Investigator award and the 2007 Presidential Early Career Award for Scientists and Engineers. He held a United Technologies Corporation Associate Professorship in Engineering Innovation from 2008 to 2011 and has been a member of the Connecticut Academy of Science and Engineering since 2010. He is a member of the IEEE.