Mobi-Sync: Efficient Time Synchronization for Mobile Underwater Sensor Networks

Jun Liu, *Student Member*, *IEEE Computer Society*, Zhong Zhou, *Member*, *IEEE Computer Society*, Zheng Peng, *Member*, *IEEE Computer Society*, Jun-Hong Cui, *Member*, *IEEE Computer Society*, Michael Zuba, *Student Member*, *IEEE Computer Society*, and Lance Fiondella, *Member*, *IEEE Computer Society*

Abstract—Time synchronization is an important requirement for many services provided by distributed networks. A lot of time synchronization protocols have been proposed for terrestrial Wireless Sensor Networks (WSNs). However, none of them can be directly applied to Underwater Sensor Networks (UWSNs). A synchronization algorithm for UWSNs must consider additional factors such as long propagation delays from the use of acoustic communication and sensor node mobility. These unique challenges make the accuracy of synchronization procedures for UWSNs even more critical. Time synchronization solutions specifically designed for UWSNs are needed to satisfy these new requirements. This paper proposes Mobi-Sync, a novel time synchronization scheme for mobile underwater sensor networks. Mobi-Sync distinguishes itself from previous approaches for terrestrial WSN by considering spatial correlation among the mobility patterns of neighboring UWSNs nodes. This enables Mobi-Sync to accurately estimate the long dynamic propagation delays. Simulation results show that Mobi-Sync outperforms existing schemes in both accuracy and energy efficiency.

Index Terms—UWSNs, synchronization, sensor node

1 Introduction

In recent years, Underwater Sensor Networks (UWSNs) have gained significant attention from academic and industrial researchers due to the potential benefits and unique challenges posed by the water environment [1], [2], [3], [4]. UWSNs have allowed a host of applications to become both feasible and effective, including coastal surveillance, environmental monitoring, undersea exploration, disaster prevention, and mine reconnaissance. However, due to the high attenuation of radio waves in water, acoustic communication is emerging as the most suitable media. Several characteristics specific to underwater acoustic communications and networking introduce additional design complexity into almost every layer of the network protocol stack [1], [2], [3], [4], [5]. For example, low communication bandwidth, long propagation delays, higher error probability, and sensor node mobility are concerns that must be confronted.

This paper addresses the time synchronization problem, a critical service in any sensor network. Nearly all UWSN applications depend on time synchronization service. For example, data mining requires global time information, TDMA, one of the most commonly used Medium Access Control (MAC) protocols, often requires nodes to be synchronized. Furthermore, most of the localization algorithms for underwater [6], [7], [8], [9] and terrestrial sensor networks [10], [11], [12], [13], [14] assume the availability of time synchronization service.

• *The authors are with the Computer Science and Engineering Department, University of Connecticut, 371 Fairfield Way, Unit 2155, Storrs, CT 06269-2155. E-mail: {jul08003, zhongzhou, zhengpeng, jcui, michael.zuba, lfiondella}@engr.uconn.edu.*

Manuscript received 11 Sept. 2011; revised 25 Jan. 2012; accepted 4 Feb. 2012; published online 13 Feb. 2012.
Recommended for acceptance by X. Cheng.
For information on obtaining reprints of this article, please send e-mail to: tpds@computer.org, and reference IEEECS Log Number TPDS-2011-09-0616.
Digital Object Identifier no. 10.1109/TPDS.2012.71.

Numerous time synchronization protocols for terrestrial Wireless Sensor Networks (WSNs) have been proposed in the literature [15], [16], [17], [18], [19], [20]. Their synchronization accuracy and energy efficiency for land-based applications is cogent. However, most of these approaches assume that the propagation delay among sensors is negligible. This is not the case in UWSNs, which suffer from the low propagation speeds of acoustic signals (roughly 1,500 m/s in water). Sensor node mobility also contributes to long and variable propagation delay in UWSNs. These additional complicating factors render previous approaches less suitable for adaptation to UWSNs. Furthermore, the batteries of underwater sensor nodes are difficult to recharge and it is often impractical to replace due to their relative inaccessibility. This lack of serviceability imposes even more stringent requirements. The UWSN will need to be energy efficient. This set of distinguishing characteristics introduce new challenges into the design of time synchronization schemes for UWSNs.

There are various time synchronization algorithms already proposed for UWSNs, including TSHL [21], MU-Sync [22], and D-Sync [23]. These algorithms effectively address the long propagation delays. However, they all exhibit particular shortcomings. For example, TSHL is designed for static networks. Therefore, it does not consider sensor node mobility. MU-Sync confronts the mobility issue, but it is not energy efficient. D-Sync overlooks the effect of the skew when estimating the Doppler shift. To overcome the limitations of existing approaches, this paper proposes Mobi-Sync, a high energy efficient time synchronization scheme specifically designed for mobile UWSNs.

The distinguishing attribute of Mobi-Sync is how it utilizes information about the spatial correlation of mobile sensor nodes to estimate the long dynamic propagation delays among nodes. The time synchronization procedure consists of three phases: delay estimation, linear regression,

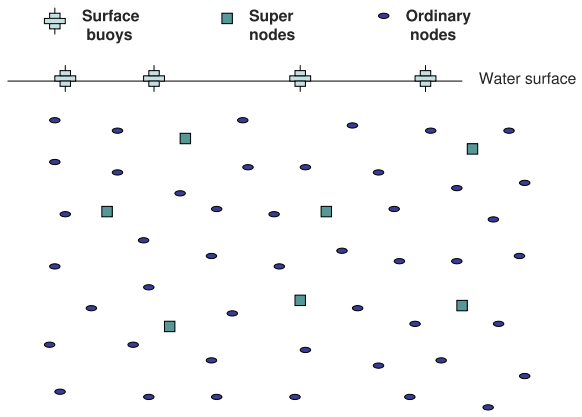


Fig. 1. Underwater sensor network architecture.

and calibration. Phase I acquires information about the spatial correlations of the mobile sensor nodes to accurately estimate the propagation delays. In Phase II, sensor nodes perform linear regression based on MAC layer time stamps and corresponding propagation delays to produce initial estimates of the clock skews and offsets. These initial results serve as inputs to Phase III, which calibrates the estimates, further improving the synchronization accuracy. During calibration, the final clock skew and offset estimates are obtained by updating certain parameters and repeating the delay calculations and linear regressions. Extensive simulations demonstrate the effectiveness of the proposed approach for time synchronization, confirming that it does not suffer from mobility. The results indicate that Mobi-Sync outperforms existing schemes with respect to both accuracy and energy efficiency.

The paper is organized as follows: background information and related work are reviewed in Section 2. Section 3 presents Mobi-Sync. Error analysis for the delay estimation is conducted in Section 4, while Section 5 presents simulation results. Conclusions and future work are offered in Section 6.

2 Background and Related Work

This section aims to introduce some background knowledge including network Architecture, pairwise synchronization, spatial correlation and some related works.

2.1 Network Architecture

In this paper, we consider a hierarchical underwater sensor network architecture [7], as shown in Fig. 1 The network consists of three types of nodes.

- *Surface buoys*. Surface buoys are equipped with GPS to obtain global time references and perform localization. They serve as the "satellite" nodes in underwater environment.
- *Super nodes*. Super nodes are powerful sensor nodes, working as reference clocks, as they always maintain synchronization with surface buoys. Moreover, super nodes can perform moving speed estimation as they can directly communicate with the surface buoys to obtain real time location and global time information.
- *Ordinary nodes*. Ordinary nodes are the sensor nodes aiming to become synchronized. They are inexpensive and have low complexity, cannot make direct contact with surface buoys and can only communicate with their neighboring ordinary nodes or super nodes. The lifetime of ordinary node is restricted by its limited battery supply.

2.2 Pairwise Synchronization

In order to perform time synchronization for pairs of clocks, most existing algorithms rely on estimating the clock offset and skew, which present the relation between the time measured by two different clocks. In most cases, the time difference is captured by exchanging time-stamped packets, or "pings," between nodes. Mobi-Sync also yields to this pairwise synchronization approach. In terms of time synchronization, an ordinary node is the clock aiming to get synchronized and a super node plays the role of the reference clock. We then get

$$T = a \cdot t + b, \tag{1}$$

where T stands for the measured time of ordinary node, t is reference time, a is the relative clock skew, and b is the offset.

2.3 Spatial Correlation

Research in hydrodynamics shows that in certain underwater environments the movement of underwater objects exerts specific characteristics related to the surrounding environmental factors such as water current, pressure and water temperature [24], [25]. Generally speaking, there is no unified mobility model that can be applied for all underwater environments. However, some mobility models have been devised show that the movement of underwater objects is not in a totally random fashion. Temporal and spatial correlation is always inherent in such movement. Fortunately, this characteristic has a positive impact on Mobi-Sync as it can be used for an ordinary node to calculate its own moving velocity.

In the network, each underwater node's mobility behavior during a specific time period can be represented with a speed vector [7], $V = [v(1), v(2), \ldots, v(i), \ldots, v(k)]$. Where $v(i)$ is the average speed corresponding to a certain short time period.

For each ordinary node, it cannot communicate with surface buoys, unlike super nodes. However, by taking advantage of sensor nodes spatial correlation characteristic, its moving velocity can be estimated as follows.

Assuming an ordinary node j aims to compute its velocity $[v_x(j), v_y(j)]$, where $v_x(j)/v_y(j)$ denotes the current speed of node j in the x/y axis. If node j can obtain the velocities of its neighboring super nodes, its own velocity can be calculated as follows [24], [25][1]

$$\begin{cases} v_x(j) = \sum_{i=1}^{m} \zeta_{ij} v_x(i), \\ v_y(j) = \sum_{i=1}^{m} \zeta_{ij} v_y(i), \end{cases} \tag{2}$$

where m is the number of neighbors, and the interpolation coefficient ζ_{ij} is calculated as

1. This paper assumes that objects maintain constant in depth (z-axis) and that the mobility pattern of objects takes place in the (x, y) axis. Therefore, objects with the same (x, y) but different z move with the same mobility pattern. This is the prevalent assumption found in hydrodynamics research [24], [25] .

$$\zeta_{ij} = \frac{\frac{1}{r_{ij}}}{\sum_{i=1}^{m} \frac{1}{r_{ij}}}, \quad (3)$$

where r_{ij} denotes the euclidean distance between node i and node j. Consequently, in order to apply the existing theory, it is necessary to know the distance between the ordinary nodes and the neighboring super nodes.

2.4 Related Work

In the literature, there are various time synchronization protocols for distributed systems like terrestrial radio sensor networks, in which ordering of events is crucial. A landmark paper in computer clock synchronization is Lamport's work [26] that elucidates the importance of virtual clocks in systems where causality is more important than absolute time. It has emerged as an important influence in sensor works, in which many applications only require relative time instead of absolute time.

The Network Time Protocol (NTP) [27] is a widely used hierarchical protocol implemented to synchronize clocks in large networks like the Internet. NTP provides accuracy in the order of milliseconds by typically using GPS to achieve synchronization to external sources that are organized in levels called stratums. However, in underwater sensor networks, GPS may not be available for all the scenarios. Additionally, one-way delay estimated as one half of the round trip transit time brings significant errors due to the long propagation delay in UWSNs.

Reference Broadcast Synchronization (RBS) [15] is a well-known receiver-receiver synchronization algorithm. It completely kills errors that derive from the sender side, and it adopts the concept of postfacto synchronization, allowing the time synchronization process to happen after data collection rather than ahead of time. However, RBS requires extra message exchange to communicate the local time-stamps between any two nodes which intend to become synchronized. The major idea of the RBS algorithm greatly depends on immediate reception of reference messages.

Timing-sync Protocol for Sensor Networks (TPSNs) [16] is a sender-receiver time synchronization which employs a two-way message exchange for synchronization. Although TPSN takes care of propagation delays, it does not take the clock skew into account during synchronization period. Instead, it only computes offset, which severely limits its accuracy.

Flooding Time Synchronization Protocol (FTSP) [17] is designed for sniper localization. Therefore, it is required to achieve considerably high precision. As FTSP applies a flooding technique, it is robust in regards to network topology changes. The FTSP is not applicable to underwater sensor network mainly because it also assumes instant reception. Additionally, it requires hardware calibration, which is not a completely software solution.

Although UWSNs have drawn considerable attention in the past several years, there are still only a few works on time synchronization in UWSNs.

TSHL [21] is time synchronization scheme designed for high-latency networks, and addresses long propagation delays and energy consumption issues. In TSHL, both one-way and two-ways MAC-layer message exchange are employed, where one-way is to estimate the clock skew and two-ways is to calculate the clock offset. TSHL assumes underwater sensor networks are static and therefore suffers from sensor nodes mobility, especially when nodes move fast [22].

MU-Sync [22] is a time synchronization approach proposed for a cluster-based UWSNs. In MU-Sync, the cluster head manages the process of time synchronization including launching 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 this cluster. In MU-Sync, linear regression runs twice. The first run allows the cluster head to estimate the draft skew by assuming constant propagation delays. The second run is used to calibrate the estimated skew and calculate the offset. Although MU-Sync claims to be able to solve the mobility issue, it has relatively high message overhead. Moreover, MU-Sync applies half of the round trip time to compute one way propagation delay, which involves significant errors when sensor nodes move fast or regular nodes experience a long response time.

D-Sync [23] considers the the Doppler shift in underwater environments caused by node mobility, which is one of the major impairments to underwater communication. By estimating and compensating the Doppler shift, the accuracy of propagation delay calculation can be improved and therefore, so will the synchronization accuracy. However, D-Sync ignores the effect of the skew during the process of calculating the Doppler scaling factor, which reduces the accuracy of synchronization. Additionally, estimating the Doppler shift needs special hardware designed for underwater acoustic communication which is not a complete software solution.

3 Description of Mobi-Sync

This section introduces Mobi-Sync in detail. An overview is presented prior to the main body of Mobi-Sync which is described with three phases. A discussion of the protocol is then conducted.

3.1 Overview of Mobi-Sync

The propagation delay estimation performed in Phase I consists of two steps, message exchange and delay calculation. In the message exchange step, an ordinary node launches time synchronization by broadcasting a request message. Upon receiving the request message, each neighboring super node schedules two response messages. These response messages contain the recorded velocity vector of the super node and its MAC layer time stamp. In the delay calculation step, by making a reasonable assumption, the ordinary node self-computes its velocity vector by utilizing spatial correlations contained in the velocity vectors of the neighboring super nodes. The ordinary node continues to broadcast request messages until it obtains enough data points to perform linear regression.

In Phase II, the ordinary node executes the first round of linear regression with a set of time stamps received from the super nodes and the corresponding propagation delays. This provides an estimate of the draft clock skew and offset. This regression employs an advanced Weighted Least

TABLE 1
Parameter Description

a, b	clock skew, clock offset
r	initial distance between an ordinary node and a super node
V_p	propagation speed
t_r	response time = t_{r1}+t_{r2}
t_{r1}, t_{r2}	the first, second response time
t_i	time period for an ordinary node to record a new sub-speed vector
V_x,V_y	relative speed in x, y axis
$v_x(j)$,$v_y(j)$	current speed of node j in the x, y axis
T_1,T_3,T_5	sending time of SR, RS_1, RS_2
T_2,T_4,T_6	receiving time of SR, RS_1, RS_2
d_1,d_2,d_3	propagation distances of SR, RS_1, RS_2
τ_1,τ_2,τ_3	propagation delays of SR, RS_1, RS_2
h_1	round trip distance of $T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_4$
h_2	round trip distance of $T_1 \rightarrow T_2 \rightarrow T_5 \rightarrow T_6$
L_1,L_2	straight-line distance super node "A" moves relatively to the ordinary node during t_{r1}, t_r

Squares Estimation (WLSE) procedure to reduce the impact of the assumption in Phase I.

In Phase III, to further improve the synchronization accuracy, the ordinary node updates certain initial parameters, such as initial skew and initial distance, and recalculates the delay and reperforms the linear regression to obtain the final clock skew and offset. All notations are listed in Table 1.

To further improve the synchronization accuracy, the ordinary nodes update certain initial parameters in Phase III, specifically initial skew and initial distance. Delay calculation is performed again and a second round of linear regression obtains the final clock skew and offset estimates. Table 1 summarizes all notations.

3.2 Phase I: Propagation Delay Estimation

3.2.1 Message Exchange

Fig. 2a illustrates message exchange among sensor nodes for the case where there are three super nodes available to assist the ordinary node perform time synchronization. Fig. 2b shows a single run of the message exchanged between the ordinary node and each super node. The synchronization procedure starts when an ordinary node initializes the synchronization process by broadcasting the synchronization request message SR to its neighboring super nodes. SR contains the sending time-stamp T_1 obtained at the MAC layer, immediately before it departs from the ordinary node. Upon receiving SR, super nodes mark their local time as T_2. From that moment, they start to record their moving velocity with the frequency of $1/t_i$. After suspending for a fixed time interval t_{r1}, each super node sends back the first response message RS_1 with a MAC layer sending time-stamp T_3.[2] The ordinary node learns T_3 from RS_1 and infers T_2 as $T_2 = T_3 - t_{r1}$. After a second fixed time period t_{r2}, super

2. Note that the distance from each super node to the ordinary node is different, Mobi-Sync assumes that response messages from the super nodes do not collide.

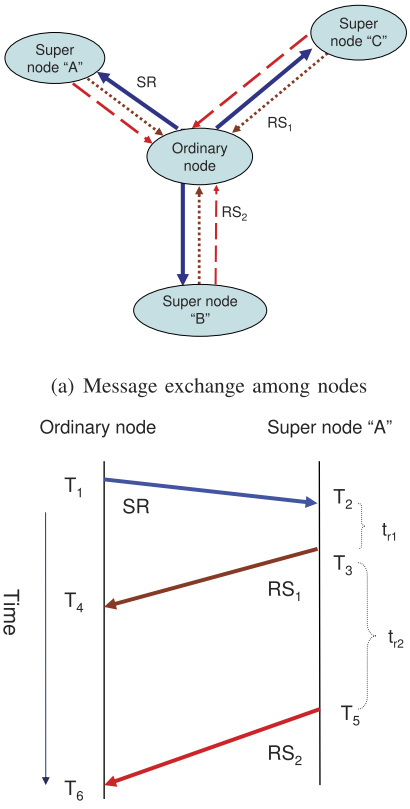


Fig. 2. Message exchange.

nodes send back the second response message RS_2. Upon launching the synchronization process, the ordinary node listens for RS_1 and RS_2, and records the corresponding receiving time T_4 and T_6 for each super node.

3.2.2 Delay Calculation

Considering the two round trips and combining with (1), we gather the round trip distance h_1 and h_2

$$\begin{cases} h_1 = d_1 + d_2 = \left(T_2 - T_3 + \dfrac{T_4 - T_1}{a}\right) \times V_P, \\ h_2 = d_1 + d_3 = \left(T_2 - T_5 + \dfrac{T_6 - T_1}{a}\right) \times V_P. \end{cases} \quad (4)$$

Where d_1 and d_2 are propagation distances of SR and RS_1, respectively. Regarding (4), to calculate h_1 and h_2, in addition to the measured time stamps, the skew "a" is also needed. However, this skew is unknown and the goal is to estimate it accurately. In Mobi-Sync, at this time, the skew is assigned with an initial value "1," named as initial skew. In doing so, h_1 and h_2 may be calculated and treated as constants to assist the subsequent computations. Any errors introduced by this assumption can be corrected in the calibration phase.

Upon receiving an RS_1 message from a super node, the ordinary node calculates the initial distance r. At this moment, the initial distance r is represented by half the round trip distance h_1. Since the first response time t_{r1} is very short, the error will be small. Later, the calibration phase will reduce the negative impact of these errors. Therefore, the initial distance r is

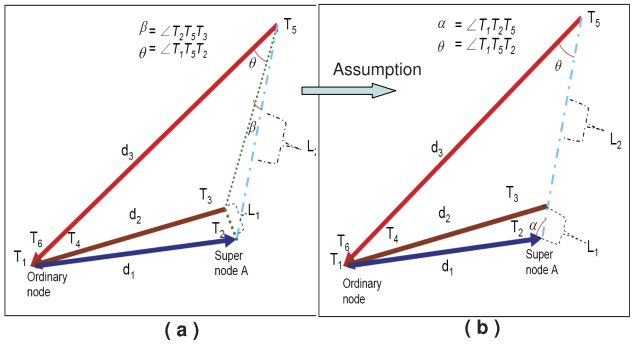


Fig. 3. Relative motion.

$$r = h_1/2 = \frac{d_1 + d_2}{2} = \frac{\left(T_2 - T_3 + \frac{T_4 - T_1}{a}\right) \times V_P}{2}. \quad (5)$$

From the initial distance r and super node velocity vectors, the ordinary node can estimate its own velocity vector by repeatedly computing the distances and velocity vectors according to (2). After calculating its velocity relative to the super nodes, the ordinary node infers the relative motion of each neighboring super node. The relative motion between each super node and the ordinary node is characterized by two triangles, namely, $\triangle T_1T_2T_3$ and $\triangle T_1T_2T_5$ as shown in Figs. 3a and 3b.[3]

Since t_{r1} is much shorter than t_{r2}, note from Fig. 3a, the straight-line distance super node "A" moves relative to the ordinary node during t_{r1}, named as L_1, will be much shorter than L_2, which is the straight-line distance super node "A" moves relative to the ordinary node during t_r. Hence, β will typically be very narrow. This suggests that Fig. 3a can be approximated as Fig. 3b. Equation (6) calculates L_1 and L_2 based on Fig. 3b.

$$\begin{cases} L_{1x} = \sum_{j=1}^{(T_3-T_2)/t_i} v_x(j)t_i, \; L_{2x} = \sum_{j=1}^{(T_5-T_2)/t_i} v_x(j)t_i, \\ L_{1y} = \sum_{j=1}^{(T_3-T_2)/t_i} v_y(j)t_i, \; L_{2y} = \sum_{j=1}^{(T_5-T_2)/t_i} v_y(j)t_i, \\ L_1 = \sqrt{L_{1x}^2 + L_{1y}^2}, \; L_2 = \sqrt{L_{2x}^2 + L_{2y}^2}. \end{cases} \quad (6)$$

In Fig. 3b, combining (4) and applying the Cosine theorem for the angle α common to both $\triangle T_1T_2T_3$ and $\triangle T_1T_2T_5$, the propagation delays for SR, RS_1, RS_2 are

$$\begin{cases} \tau_1 = f_d(L_1, L_2, h_1, h_2) = \dfrac{L_1\left(h_2{}^2 - L_2{}^2\right) + L_2(L_1{}^2 - h_1{}^2)}{2(L_1h_2 - L_2h_1)V_p}, \\ \tau_2 = h_1/V_p - f_d(L_1, L_2, h_1, h_2), \\ \tau_3 = h_2/V_p - f_d(L_1, L_2, h_1, h_2). \end{cases} \quad (7)$$

3.2.3 Multiple Requests

Based on its accuracy requirement, the ordinary node requests multiple runs of this synchronization process. Each request follows the sequence of steps described in the previous section such that propagation delays can be estimated. For each neighboring super node, the ordinary node collects a set of time stamps consisting of T_3, T_4, T_5, and T_6 as well as the propagation delays of the exchanged messages.

3. Figs. 3a and 3b are picked as a general instance to depict the relative motion between an ordinary node and any super node "A."

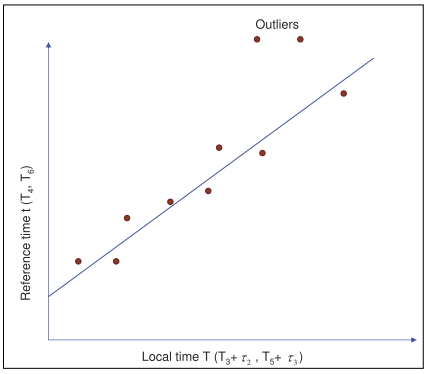


Fig. 4. Linear regression.

3.3 Phase II: Linear Regression

In Phase II, the ordinary node performs the first run of linear regression, estimating the draft clock skew and offset shown in Fig. 4 Each round of message exchange described by Fig. 2b obtains two sample points.

$$(T_{3,i} + \tau_{2,i}, T_{4,i}) \; and \; (T_{5,i} + \tau_{3,i}, T_{6,i}), \quad (8)$$

where i is the index of the message exchange round. Linear regression is performed on these sample points. Mobi-Sync assumes that L_2 is much longer than L_1, which may not hold for all mobility patterns. Consequently, some sample points will inevitably contain significant error. A Weighted Least Squares Estimation procedure reduces the impact of outliers. The sum of squared deviations is

$$\sigma(b, a) = \sum_{i=1}^{m} \omega_i \left(T_i - b - a f_p(t_i)\right)^2, \quad (9)$$

where T_i denotes T_4 or T_6, and $f_p(t_i)$, represents $T_3 + \tau_2$ or $T_5 + \tau_3$ respectively. WLSE produces estimates for $\hat{a}$ and $\hat{b}$ that minimizes $\sigma(b, a)$ of (9)

$$\begin{aligned} \sigma(\hat{b}, \hat{a}) &= \sum_{i=1}^{m} \omega_i \left(T_i - \hat{b} - \hat{a} f_p(t_i)\right)^2 \\ &= min \sum_{i=1}^{m} \omega_i \left(T_i - b - a f_p(t_i)\right)^2. \end{aligned} \quad (10)$$

To weigh the importance of each data point with the coefficient ω_i, Mobi-Sync uses the following observations. A potential source of error in Mobi-Sync comes from the assumption $\beta = 0$, which holds only when L_2 is much longer than L_1. The maximal error induced by this assumption comes from the data point with the largest value of beta. Assuming L_1 is the radius of the circle in Fig. 5, the super node could be at any point on the circle at time T_3. When $\overrightarrow{T_2T_3}$ is perpendicular to $\overrightarrow{T_5T_3}$ ($\gamma = \pi/2$), beta is maximized. $F(\beta) = \sin(\beta)$ is monotonically increasing within $[0, \pi/2]$ and upper bounded by L_1/L_2. Thus, $F(\beta)$ indicates the error. Therefore, weight is defined as

$$\omega_i = 1/F(\beta[i]) = L_2[i]/L_1[i], \quad (11)$$

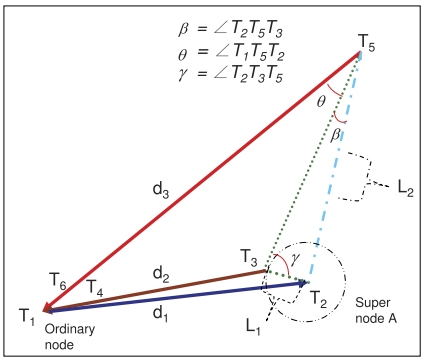


Fig. 5. Weight definition.

where i is the index of the run. WLSE increases the utility of the data samples and reduces the impact of outliers, improving the accuracy of the estimates for skew and offset.

3.4 Phase III: Calibration

In Phase III, the ordinary node performs the calibration process to improve accuracy. Due to node mobility, the distance d_1 will be different from the distance d_2, so the initial distance $r = d_1 \neq \frac{d_1+d_2}{2}$. Since both d_1 and d_2 have been estimated, it is now possible to update the initial distance "r" with d_1 and recalculate the velocity vectors. The initial skew we set to "1" when calculating h_1 and h_2 during Phase I is also revised . The first estimate of skew is available, making the recalculation of the round trip time h_1 and h_2 possible. This is accomplished with no additional message overhead by redoing the calculations with the original time stamps and updated parameters, further calibrating the skew and offset.

3.5 Discussions

This section provides a qualitative discussion of Mobi-Sync's key parameters, including implementation tradeoffs that must be considered.

The first response time t_{r1} should be as short as possible. A short interval minimizes the distance a super node moves between T_1 and T_4. This ensures that half the round trip time from T_1 to T_4 is closer to the one way trip time from T_1 to T_2, making the distance and velocity vector estimates more accurate. However, the length of t_{r1} is restricted by hardware constraints, including the sending-receiving mode transition delay and MAC layer issues like collisions. Therefore, some delays cannot be avoided.

The parameter t_i controls the frequency at which super nodes record their speed. This directly influences the accuracy of the estimated velocity vector. When t_i is shorter, the super node records its motion more often, enabling the ordinary node to estimate its velocity with greater precision. However, the size of t_i also determines the size of the velocity vector. Since t_r is constant, a small value of t_i increases the size of the velocity vector.

Now consider the second response time parameter t_{r2}. It is expected that a super node will move away from its original position during this period. This makes the assumption ($\beta = 0$) more reasonable, rendering the algorithm more practical. It is relevant to ask if increasing t_{r2} guarantees L_2 will also be longer. The answer is in fact, no, because sensor nodes may move according to a sophisticated mobility pattern, changing its heading on several occasions. The value of t_{r2} that produces a long L_2 is closely related to the mobility pattern. Furthermore, if t_{r2} is too long, the magnitude of the velocity vector must be very large. This will make the size of the RS_2 unmanageably large. Mobi-Sync must avoid sending or receiving large packets because this will interfere with synchronization. Larger packets also increase energy consumption and the probability of packet collision. Therefore, t_{r2} must guarantee L_2 is long, but remain within a reasonable range of values. Under a certain mobility pattern and fixed t_{r1} and t_i the simulations shown in Fig. 7a identified 0.6 ms as the optimal value of t_{r2}, which controls packet size as well as the length of t_{r2}.

The parameter t_r consists of t_{r1} and t_{r2}. It collaborates with t_i to manage the size of the velocity vectors and the size of the RS_2 message. Therefore, given t_i, the length of t_r must be chosen such that the size of the speed vector is acceptable.

4 Error Analysis of Delay Estimation

This section analyzes the influence of errors in the estimation of propagation delay. To do this, it is crucial to understand that all measurements of physical quantities are subject to uncertainty. Each of the quantities in the expression $d_1 = f_d(L_1, L_2, h_1, h_2)$, contain some error. Define ΔL_1, ΔL_2, Δh_1, and Δh_2 as the average error in the measurements of L_1, L_2, h_1, and h_2. It is necessary to study how these introduce error into the estimate of propagation delay d_{p1}.

The values ΔL_1 and ΔL_2 arise from the error in the relative velocity, defined as ΔV_x and ΔV_y, because of inaccuracies in the measurement of the distance and the recorded velocity vectors of the super nodes. Error in h_1 and h_2, are a result of the approximation used to estimate initial skew.

First, consider how ΔV_x and ΔV_y affect ΔL_1 and ΔL_2. From (2) and the theory of error propagation [28], [29]:

$$\begin{cases} \Delta L_{1x} = \sum_{j=1}^{m} |t_i \Delta V_x[j]|, \\ \Delta L_{1y} = \sum_{j=1}^{m} |t_i \Delta V_y[j]|. \end{cases} \tag{12}$$

In order to simplify the analysis of error, it is assumed that $\Delta V_x[1] = \cdots = \Delta V_x[j] \cdots = \Delta V_x[m] = \Delta V_x$ and $\Delta V_y[1] = \cdots = \Delta V_y[j] \cdots = \Delta V_y[m] = \Delta V_y$, so that

$$\begin{cases} \Delta L_{1x} = \sqrt{m {t_i}^2 {\Delta V_x}^2} = \sqrt{m} t_i \Delta V_x, \\ \Delta L_{1y} = \sqrt{m {t_i}^2 {\Delta V_y}^2} = \sqrt{m} t_i \Delta V_{y,} \end{cases} \tag{13}$$

where m and n are

$$\begin{cases} m = \frac{T_3 - T_2}{t_i}, \\ n = \frac{T_5 - T_2}{t_i}. \end{cases} \tag{14}$$

Similarly, for ΔL_2:

$$\begin{cases} \Delta L_{2x} = \sqrt{n {t_i}^2 {\Delta V_x}^2} = \sqrt{n} t_i \Delta V_x, \\ \Delta L_{2y} = \sqrt{n {t_i}^2 {\Delta V_y}^2} = \sqrt{n} t_i \Delta V_y. \end{cases} \tag{15}$$

Considering the length of vector $\vec{L_1}$ and $\vec{L_2}$:

$$\begin{cases} L_1 = \sqrt{L_{1x}^2 + L_{1y}^2}, \\ L_2 = \sqrt{L_{2x}^2 + L_{2y}^2}. \end{cases} \tag{16}$$

Applying error propagation rules [28], [29], produces

$$\begin{cases} \Delta L_1 = \sqrt{\left(\dfrac{\partial L_1}{\partial L_{1x}}\Delta L_{1x}\right)^2 + \left(\dfrac{\partial L_1}{\partial L_{1y}}\Delta L_{1y}\right)^2}, \\ \Delta L_2 = \sqrt{\left(\dfrac{\partial L_2}{\partial L_{2x}}\Delta L_{2x}\right)^2 + \left(\dfrac{\partial L_2}{\partial L_{2y}}\Delta L_{2y}\right)^2}. \end{cases} \tag{17}$$

To simplify these expressions, it is assumed that these errors exert the same impact on the speed estimation in X and Y axis, $\Delta V_x = \Delta V_y = \Delta V$. By doing so, we gather

$$\begin{cases} \Delta L_1 = \sqrt{m} t_i \Delta V, \\ \Delta L_2 = \sqrt{n} t_i \Delta V. \end{cases} \tag{18}$$

As a result, Δh_1 and Δh_2 arise from imprecise initial skew Δa. It follows from (4) that

$$\begin{cases} \Delta h_1 = \dfrac{(T_1 - T_4)\Delta a}{a^2}, \\ \Delta h_2 = \dfrac{(T_1 - T_6)\Delta a}{a^2}. \end{cases} \tag{19}$$

Considering all errors including ΔL_1, ΔL_2, Δh_1, Δh_2, and adhering to the measured data analysis approach [28], Δd_1 can be expressed as a function of these errors

$$\Delta d_1 \approx \left|\frac{\partial f_d}{\partial L_1}\Delta L_1\right| + \left|\frac{\partial f_d}{\partial L_2}\Delta L_2\right| + \left|\frac{\partial f_d}{\partial h_1}\Delta h_1\right| + \left|\frac{\partial f_d}{\partial h_2}\Delta h_2\right|. \tag{20}$$

To simplify, the following notations are introduced:

$$\begin{cases} \dfrac{\partial f_d}{\partial L_1} = \varphi, \dfrac{\partial f_d}{\partial L_2} = \chi, \\ \dfrac{\partial f_d}{\partial h_1} = \psi, \dfrac{\partial f_d}{\partial h_2} = \omega, \\ T_1 - T_2 = \eta, \\ T_1 - T_6 = \phi. \end{cases} \tag{21}$$

Based on [28], [29], the error in the propagation delay Δd_1 caused by ΔL_1, ΔL_2, Δh_1, and Δh_2 is

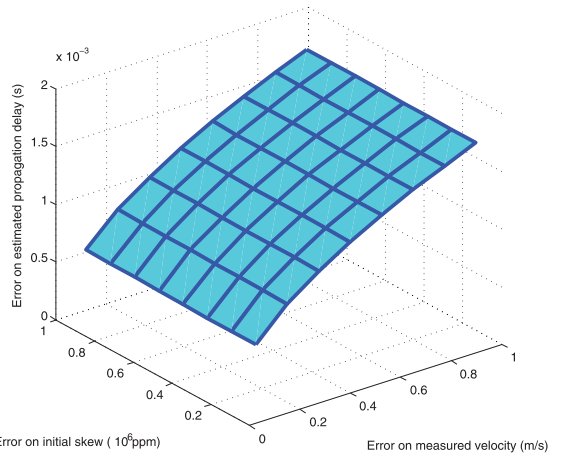


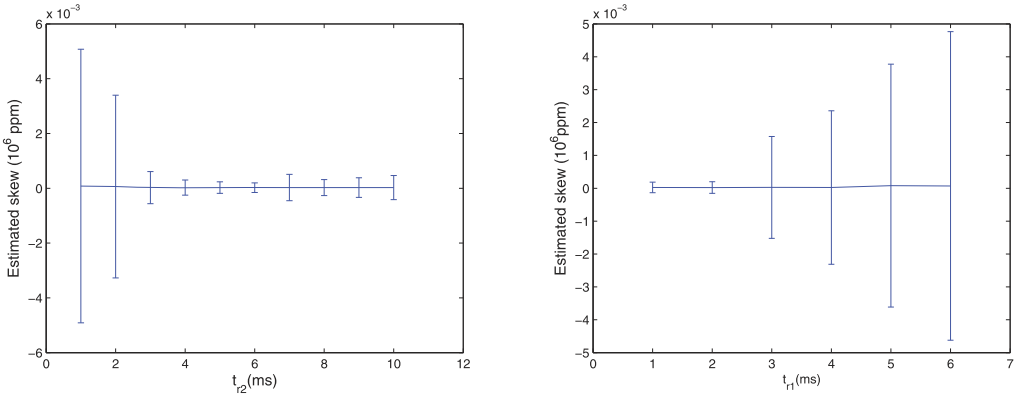
Fig. 6. Error delivered to propagation delay.

$$\Delta d_{p_1} \approx \sqrt{(\varphi\Delta L_1)^2 + (\chi\Delta L_2)^2 + (\psi\Delta h_1)^2 + (\omega\Delta h_2)^2}/V_p. \tag{22}$$

Combining all the equations above, the error in the propagation delay Δd_{p_1} as a function of ΔV and Δa is

$$\Delta d_{p_1} \approx \sqrt{m(t_i\varphi\Delta V)^2 + n(t_i\chi\Delta V)^2 + \left[(\psi\eta + \omega\phi)\frac{\Delta a}{a^2}\right]^2}\Big/V_p. \tag{23}$$

Equation (23) characterizes the error in the estimate of propagation delay due to measurement errors in ΔV and Δa. While the analysis is performed in the context of d_{p_1}, it also pertains to d_{p_2} and d_{p_3}. Fig. 6 illustrates this relationship more generally. The configuration parameter for this scenario is the same with the simulation. From Fig. 6, it is obvious that error in the propagation delay is always below 2 ms, even when the error in the measured velocity is 1 m/s and the error in the initial skew is 10^6 ppm. Such an error in the skew is extremely large and rare. The accuracy will almost always be much better. Novikov and Bagtzoglou [24] showed that error in velocity estimates are less than 1 ft/s (0.305 m/s), suggesting that the error in the propagation will be on the order of microseconds.



(a) Effect of t_{r2} (b) Effect of t_{r1}

Fig. 7. Effect of t_{r1} and t_{r2}.

TABLE 2
Parameter Setting

Setting	Value	Setting	Value
inherent skew	$50ppm$	clock offset	$0.000080s$
t_{r1}	$2ms$	propagation speed	$1500m/s$
t_{r2}	$6ms$	t_i	$1ms$
k_1	N(π,0.1π)	k_2	N(π,0.1π)
k_3	N(2π,0.2π)	k_4	N(1,0.1)
k_5	N(1,0.1)	λ	N(3,0.3)
v	N(1,0.1)m/s		

5 Performance Evaluation

In this section, we evaluate the performance of Mobi-Sync with simulations.

5.1 Simulation Settings

In the simulation, 10 super nodes and 30 regular sensor nodes are randomly deployed in a $100\ \mathrm{m} \times 100\ \mathrm{m} \times 100\ \mathrm{m}$ region. One ordinary node is randomly picked as the sample node wishes to synchronize with its neighboring super nodes. Without loss of generality, the inherent skew of this ordinary node is defined as $50\ \mathrm{ppm}$, and it maintains unchanged during the time synchronization procedure. The clock offset of this ordinary node is initialized as 0.8×10^{-4} s. Additionally, the predefined parameter t_{r1} is set to $2\ \mathrm{ms}$, and t_i is $1\ \mathrm{ms}$. The propagation speed in the simulated environment remains constant at $1{,}500\ \mathrm{m/s}$. The kinematic model of [25] characterizes the mobility behavior. The node velocities are

$$\begin{cases} V_x = k_1 \lambda v \sin(k_2 x)\cos(k_3 y) + k_1 \lambda \cos(2k_1 t) + k_4, \\ V_y = -\lambda v \cos(k_2 x)\sin(k_3 y) + k_5, \end{cases} \quad (24)$$

where V_x and V_y determine the velocity along the X and Y axis respectively. $k_1, k_2, k_3, \lambda, v$ are variables, which are closely related to environment factors such as tides and bathymetry. These parameters change with different environments. Random variables k_4, k_5 simulate additional random physical factors. In our simulations, we assume k_1, k_2 to be random variables which are subject to normal distribution with π as mean values and the standard derivations to be 0.1π. k_3 is subject to normal distribution with 2π as mean value and the standard derivation to be 0.2π. λ is subject to normal distribution with 3 as the mean value and 0.3 as the standard derivation. v is subject to normal distribution with 1 as the mean value and 0.1 as the standard derivation. k_4, k_5 are random variables which are subject to normal distribution with 1 as mean values and 0.1 as standard derivations. Table 2 shows the numerical values for the mean and variance of these variables.

5.2 Results and Analysis

Unless otherwise specified, the values reported here were obtained from the average of 1,000 simulation runs.

Figs. 7a and 7b investigate how parameter t_{r1} and t_{r2} affect the accuracy of the synchronization with fixed mobility pattern and t_i, and within the range of an acceptable size of speed vectors (within possible value range of t_{r2}). As discussed in Section 3, t_{r1} is restricted by hardware constraints, we first fix it as 2 ms to evaluate the effect of t_{r2}. Fig. 7a shows that when t_{r2} increases, the estimated skew does not necessarily get closer to the inherent skew. This may be explained by observing that sensor nodes change their direction frequently under the kinematic mobility pattern. After T_3, the relative motion between the super nodes and the ordinary node are correlated due to the mobility pattern, so L_2 is not necessarily longer when t_{r2} increases. From the large number of experiments, the value $t_{r2} = 6\ \mathrm{ms}$ minimized the difference between the estimated and real skew. Subsequent simulation experiments reported here use this optimal value of 6ms for t_{r2}. Fig. 7b demonstrates how t_{r1} affects the accuracy of the synchronization with fixed t_{r2}, which is $6\ \mathrm{ms}$. It clearly shows that the error goes up as t_{r1} increases. This is because when t_{r1} increases, L_1 is no longer much shorter than L_2, which is an important requirement of Mobi-Sync to achieve high-accuracy propagation delay estimation. To be realistic, subsequent simulation experiments use this $2\ \mathrm{ms}$ for t_{r1} although we prefer smaller t_{r1}.

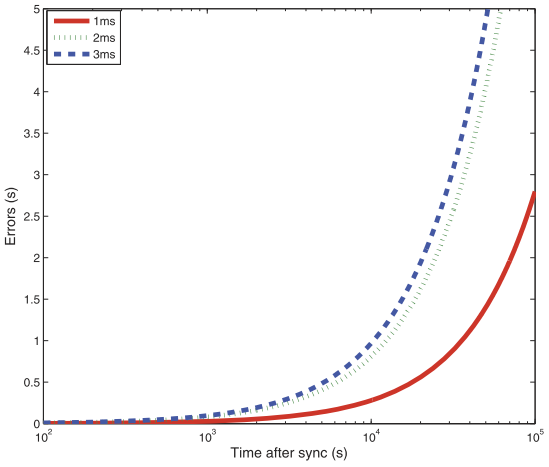


Fig. 8. Effect of t_i.

Fig. 8 shows the effect of t_i on the accuracy of Mobi-Sync. Recall from the discussion in Section 3 that t_i influences the accuracy of the estimated velocity vectors. Shorter t_i produces more precise velocity vectors. The simulation fixes all other parameters including those of the mobility pattern, t_r, t_{r1}, and t_{r2}, and varies the value t_i to investigate its impact on the error. The specific parameter values used were $t_{r1} = 2\ \mathrm{ms}$, $t_{r2} = 6\ \mathrm{ms}$ and $t_r = 8\ \mathrm{ms}$. Mobi-Sync was then simulated with $t_i = 1\ \mathrm{ms}$, 2 ms, and 3 ms. The results indicate that the accuracy of time synchronization deteriorates for larger values of t_i. This result is expected because t_i determines the precision of the recorded velocity vectors. Decreasing t_i improves the precision of the recorded velocity vectors.

Fig. 9 demonstrates the impact of calibration procedure (For MU-Sync, the second run of regression is regarded as calibration). This improvement represents the percentage of clock skew accuracy improved by calibration, which is (estimated skew without calibration—inherent skew)/(estimated skew with calibration—inherent skew). The calibration procedure achieves more accurate clock skew which could cause increasing clock drift. As time goes by, the benefits of the calibration procedure become more and more significant. Comparing with MU-Sync, the calibration procedure of Mobi-Sync is more efficient since it not only considers the impact of initial skew but also the initial distance $r = d_1$. And the calibration procedure does not involve extra message exchanges. Instead, it only requires one more run of the calculations (4), (5), (6), (7) and one

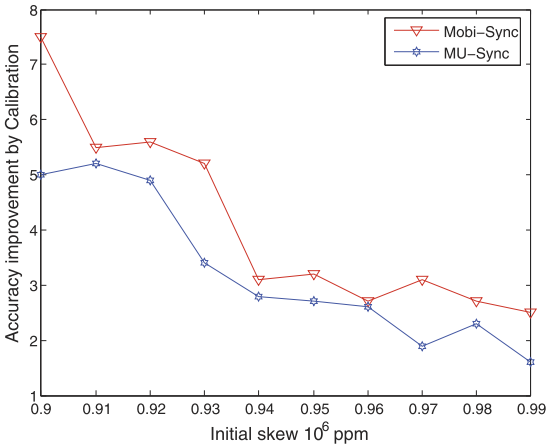


Fig. 9. Effect of calibration procedure.

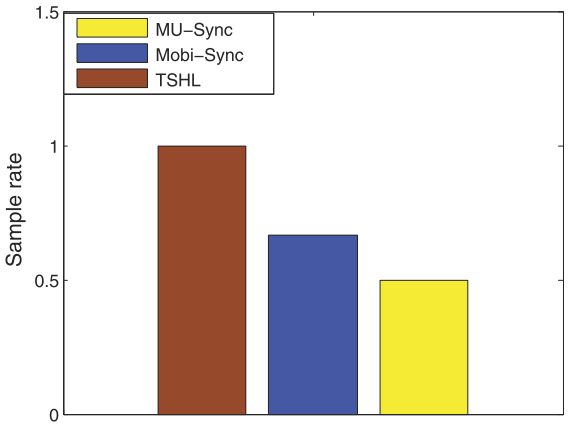


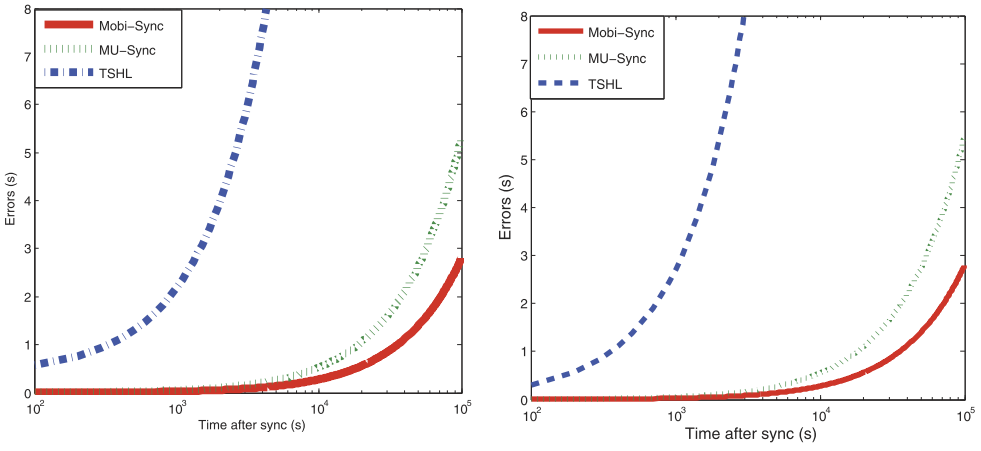
Fig. 11. Sample rate.

more run of linear regression, which makes the calibration procedure effective and efficient.

Figs. 10a and 10b compare how errors accumulate after time synchronization completes. As time progresses, the skew introduces increasing errors, revealing how effective the different algorithms are for estimating the skew. In Fig. 10a, all three algorithms, Mobi-Sync, MU-Sync, and TSHL are compared with the same message overhead of 28 synchronization messages. Fig. 10a compares them with 3 runs of message exchanges. The results indicate that Mobi-Sync achieves higher accuracy than both of its competitors.

TSHL fails to achieve high accuracy in mobile UWSNs because it assumes constant propagation delay during the synchronization process, introducing significant errors. Although both MU-Sync and Mobi-Sync utilize half of the round trip time to compute the draft initial distance r, the two algorithms possess significant differences. MU-Sync assumes that the propagation delay τ_1 is always equivalent to τ_2, during each synchronization handshake. This is impractical, however, since sensor nodes are always moving in an underwater environment. In Mobi-Sync, the propagation delays τ_1, τ_2, and τ_3 are calculated separately as three different edges of two triangles. Half of the round trip time is used to obtain a draft value of the initial distance, but this value is updated in the calibration process. Given the round trip time $\tau_1 + \tau_2$, MU-Sync makes the simplifying assumption that $\tau_1 = \tau_2$, but Mobi-Sync estimates τ_1 and τ_2 separately. This enables Mobi-Sync to achieve higher accuracy in mobile networks than either TSHL or MU-Sync.

Fig. 11 shows the different sample rates of the three algorithms. The sample rate represents the efficiency of the exchanged messages to linear regression. In the three algorithms, TSHL has highest sample rate. Without considering dynamic propagation delays, TSHL employs linear regression over single direction communication during the skew estimation process. This means every exchanged message except the message used to calculate offset can be taken as a sample data point during linear regression. Therefore, its message sample rate is close to 100 percent. MU-Sync has the lowest message sample rate, as it adopts two-way message exchanges, and assumes the propagation delays on the way back and forth are identical. Accordingly, only one way or half of those exchanged messages can be utilized as sample data. In this regard, the messages sample of MU-Sync is as low as 50 percent. In Mobi-Sync, although super nodes respond with two messages RS_1 and RS_2 for one request message SR, both of the two response messages can be applied as sample points. This is because RS_1 and RS_2 are estimated respectively and there is a certain time interval between RS_1 and RS_2. Furthermore, if the synchronization request message "SR" is considered as a broadcasted message, its sample rate can be higher.



(a) Same amount of messages (b) Same runs

Fig. 10. Error versus Time since Sync.

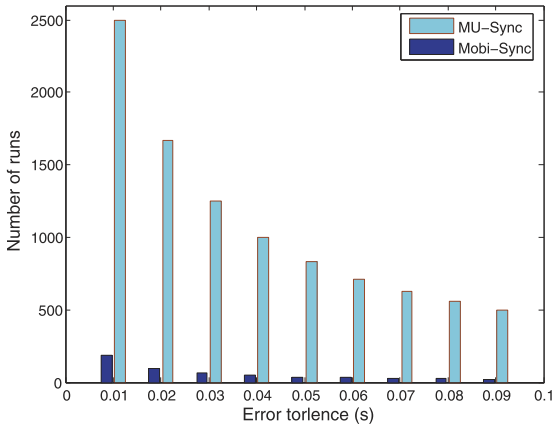


Fig. 12. Energy efficiency.

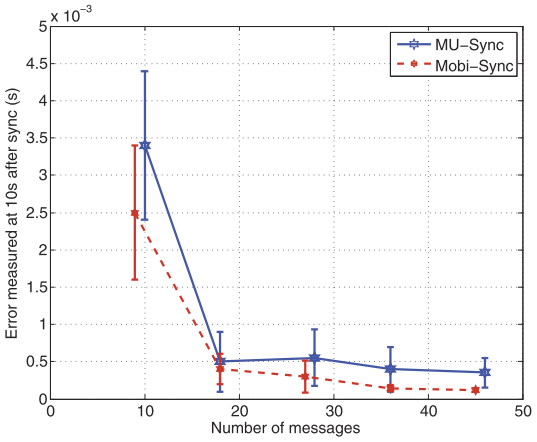


Fig. 13. Effect of message overhead on accuracy.

Fig. 12 compares the energy efficiency of the three synchronization algorithms. During a fixed interval of operation, a large drift between the ordinary and reference nodes will require more frequent resynchronization, consuming additional energy. Fig. 12 illustrates the amount of energy consumed by resynchronization (represented by runs of resynchronization) over a period of 10^5 s of operating time for various values of error tolerance. Mobi-Sync outperforms MU-Sync due to its ability to estimate skew more accurately. The result also shows that for both algorithms the number of resynchronizations decreases as error tolerance decreases.

Fig. 13 demonstrates how message overhead affects the accuracy of Mobi-Sync. Theoretically, from the perspective of linear regression, more messages exchanges transpire, providing more sample data with which to conduct linear regression. This produces more precise estimates of skew and offset. In this scenario, we perform Mobi-Sync with 3, 4, and 5 runs, respectively. In other words, 27, 36, and 45 messages. As expected, the results reveal that Mobi-Sync achieves better accuracy when more messages are exchanged during the time synchronization process. However, this greater number of messages consumes a larger amount of energy. This energy expenditure should be strictly managed because underwater sensor nodes directly depend on energy conservation to extend their life time. This reveals the tradeoff between time synchronization accuracy and message overhead. It also demonstrates that Mobi-sync can achieve higher accuracy than MU-sync when allotted the same number of messages.

6 Conclusions and Future Work

This paper presents Mobi-Sync, a time synchronization scheme for mobile UWSNs. Mobi-Sync is the first time synchronization algorithm to utilize the spatial correlation characteristics of underwater objects, improving the synchronization accuracy as well as the energy efficiency. The simulation results show that this new approach achieves higher accuracy with a lower message overhead.

In the future, the work will be extended in two directions: 1) explore other underwater mobility patterns, including one that involves vertical movement to examine the suitability of our design and; 2) investigate the influence of errors on super node localization as well as velocity estimation, and also the influence on MAC layer activities such as packet loss and re-transmission.

References

[1] 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.

[2] 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, pp. 12-18, May/June 2006.

[3] 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.

[4] 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, http://prisms.cs.umass.edu/brian/pubs/partan.wuwnet2006.pdf, Sept. 2006.

[5] J.-H., C.Z. Zhou, and S. Le, "An OFDM Based MAC Protocol for Underwater Acoustic Networks," *Proc. ACM Int'l Workshop UnderWater Networks (WUWNet)*, Sept. 2010.

[6] 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 (NETWORKING)*, May 2007.

[7] Z. Zhou, J.-H. Cui, and A. Bagtzoglou, "Scalable Localization with Mobility Prediction for Underwater Sensor Networks," *Proc. IEEE INFOCOM '08*, 2008.

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

[9] N.T.M. Hussain, "Distributed Localization in Cluttered Underwater Environments," *Proc. ACM Int'l Workshop UnderWater Networks (WUWNet)*, Sept. 2010.

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

[11] 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.

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

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

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

[15] 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.

[16] 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.

[17] 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.
[18] F. Sivrikay and B. Yener, "Time Synchronization in Sensor Networks: A Survey," *IEEE Network,* vol. 18, no. 4, pp. 45-50, July/Aug. 2004.
[19] M. Sichitiu and C. Veerarittiphan, "Simple, Accurate Time Synchronization for Wireless Sensor Networks," *Proc. IEEE Wireless Comm. and Networking Conf.,* 2003.
[20] S.A. Saurabh Ganeriwal, Ram Kumar, and M. Srivastava, "Network-Wide Time Synchronization in Sensor Networks," technical report UCLA, Apr. 2002.
[21] A. Syed and J. Heidemann, "Time Synchronization for High Latency Acoustic Networks," *Proc. IEEE INFOCOM,* 2006.
[22] 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.
[23] C.S.F. Lu and D. Mirza, "D-Sync:Doppler-Based Time Synchronization for Mobile Underwater Sensor Networks," *Proc. ACM Int'l Workshop UnderWater Networks (WUWNet),* Sept. 2010.
[24] 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.
[25] A.C. Bagtzoglou and A. Novikov, "Chaotic Behavior and Pollution Dispersion Characteristics in Engineered Tidal Embayments: A Numerical Investigation," *J. Am. Water Resources Assoc.,* vol. 43, pp. 207-219, 2007.
[26] L. Lamport, "Time, Clocks and Ordering of Events in a Distributed System," *Proc. Comm. ACM,* pp. 558-565, July 1978.
[27] D.L. Mills, "Internet Time Synchronization: The Network Time Protocol," *IEEE Trans. Comm.,* vol. 39, no. 10, pp. 1482-1493, Oct. 1991.
[28] P. Bork, H. Grote, D. Notz, and M. Regler, *Data Analysis Techniques in High Energy Physics Experiments.* Cambridge Univ. Press, 1993.
[29] J.R. Taylor, *An Introduction to Error Analysis: The Study of Uncertainties if Physical Measurements.* Univ. Science Books, 1982.

Jun Liu (S'08) received the BEng degree in computer science from the Wuhan University, China, in 2002. Currently, he is pursuing the PhD degree and working as a research assistant at the Underwater Sensor Network (UWSN) Lab, University of Connecticut. His major research focus on 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 Computer Society.

Zhong Zhou (S'06) received the BEng degree in telecommunication engineering and the MEng degree in computer science, both from the Beijing University of Posts and Telecommunications, China, in 2000 and 2003, respectively, and the PhD degree in computer science and engineering at the University of Connecticut (UCONN), Storrs, in 2010. And now, he is working as an software engineer in Amazon, Seattle, Washington. Since January 2006, he has been with the Underwater Sensor NetworkLab and the Ubiquitous Networking Research Lab,UCONN, as a graduate research assistant. His research interests include underwater acoustic communication and networking, localization, and cross layer design for wireless networks. He is a member of the IEEE Computer Society.

Zheng Peng (S'05) received the BS degrees in both computer science and control science from Zhejiang University, China, in 2002, the M.S. degree in computer science from the University of Electrical Science and Technology of China (UESTC) in 2005. Currently, he is pursuing the PhD degree and working as a research assistant at the Underwater Sensor Network (UWSN) Lab, University of Connecticut. His main research interests include underwater acoustic networks, including protocol design, operating system, underwater sensor nodes and testbeds. He is a member of the IEEE Computer Society.

Jun-Hong Cui (M'03) received the BS degree in computer science from Jilin University, China, in 1995, the MS degree in computer engineering from Chinese Academy of Sciences in 1998, and the PhD degree in computer science from UCLA in 2003. Currently, she is on the faculty of the Computer Science and Engineering Department at the University of Connecticut. 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, 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 MCCR (Mobile Computing and Communications Review) and Elsevier Ad Hoc Networks. She cofounded the first ACM International Workshop on UnderWater Networks (WUWNet'06), and she is now serving as the WUWNet steering committee chair. She is a member of ACM, ACM SIGCOMM, ACM SIGMOBILE, IEEE, IEEE Computer Society, and IEEE Communications Society. She is a member of the IEEE Computer Society.

Michael Zuba (S'11) received the BS degree in computer science and engineering from the University of Connecticut in 2010. Currently, he is pursuing the PhD degree and working as a research assistant at the Underwater Sensor Network (UWSN) Lab, University of Connecticut. His main research interests include underwater acoustic networks, autonomous underwater vehicle networks and security. He is a student member of the IEEE Computer Society.

Lance Fiondella is working toward the PhD degree in the Department of Computer Science and Engineering at the University of Connecticut. His research interests include performance, reliability, and security engineering. He was a recipient of the Best Paper award at the International Symposium on Decision Sciences in Global Enterprise Management (ISDSI) in 2009 and Second Place in the 2011 Tom Fagan Reliability and Maintainability Symposium (RAMS) Student Paper Competition. He was a 2007 recipient of a scholarship from the IEEE Reliability Society. In 2011, he was an invited speaker at the Department of Homeland Security (DHS) Student Day Conference and the recipient of an East Asia and Pacific Summer Institutes (EAPSI) for US Graduate Students from the National Science Foundation, a Student Grant from the Intelligent Transportation Society of Connecticut, and a Koerner Family Fellowship. He is a member of the IEEE Computer Society.