D-Sync: Doppler-Based Time Synchronization for Mobile Underwater Sensor Networks

Feng Lu, Diba Mirza, Curt Schurgers
Electrical and Computer Engineering Department
University of California, San Diego
{f1lu, diba}@ucsd.edu, curts@ece.ucsd.edu

ABSTRACT

Time synchronization is an essential service in underwater networks, required for many functionalities such as MAC, sleep-scheduling, localization, and time-stamping of sensor events. However, there exist two fundamental challenges to underwater synchronization, namely, large propagation delays and substantial node mobility during the synchronization process. While existing underwater time sync solutions have been proposed to address these challenges, they rely on heavy signaling, which is undesirable due to high energy costs. In this paper, we introduce a powerful new approach that incorporates physical layer information, namely an estimate of the Doppler shift. Large Doppler shift has been identified as a major challenge to underwater communication, and current systems implement sophisticated solutions to estimate and track such Doppler shift for each data exchange. While an impediment to communication, we will show that the Doppler shift contains highly useful information that can be leveraged to greatly improve time synchronization. Specifically, it provides an indication of the relative motion between nodes. Our new protocol, called D-sync, strategically exploits this feature to address the timing uncertainty due to node mobility. As such, D-sync can handle substantial mobility, without making any assumptions about the underlying motion, and without extensive signaling. Simulation results show that D-sync significantly outperforms existing time synchronization both in terms of accuracy and energy.

Categories and Subject Descriptors

C.2.1 [**Computer Communication Networks**]: Network Architecture and Design

General Terms

Algorithms, Design, Analysis

WUWNet'10, Sept. 30 - Oct. 1, 2010, Woods Hole, Massachusetts, USA

Keywords

Time Synchronization, Doppler Shift, Mobility, Underwater Networks

1. INTRODUCTION

A number of important network functionalities require that nodes have a common notion of time. These include time stamping of events, distributed data aggregation, MAC and localization. However, the local clock of nodes has an intrinsic drift due to which nodes go out of sync as time elapses. Therefore, time synchronization protocols are crucial to any distributed system. Although the accuracy of time synchronization required for underwater networks is much lower than that for terrestrial networks due to the fact that communication takes place at a much coarser time-granularity, still, accurate time-synchronization can alleviate the need for frequent re-synchronizations, which saves energy. Therefore, precise time-synchronization, using minimal signaling, is of primary importance to underwater networks given that underwater devices are highly energy constrained.

All network time synchronization methods rely on some sort of message exchange between nodes to establish a common notion of time. However, non-determinism in the network dynamics such as medium access time, propagation time or interrupt handling time makes synchronization challenging[12]. Unfortunately, traditional time sync protocols[2, 3, 8] developed for terrestrial sensor networks cannot be applied to underwater networks because of long and unknown propagation delays of acoustic signals[7]. The propagation delay, if not estimated, translates into a timing uncertainty between a pair of synchronizing nodes. This problem becomes more aggravated when nodes are mobile. Because the propagation delay changes over time, measurements obtained from time-stamped messages are no longer sufficient to accurately synchronize nodes. While a few time-sync protocols[13, 1] have been specifically designed for underwater networks, they are more suitable for networks with very limited mobility.

Two main factors contribute to the pronounced effect of mobility on the performance of underwater time synchronization. First, the variation in the propagation delay depends on the relative speed on nodes. So, while devices that are not propelled such as underwater drifters and gliders, have maximum speeds of up to $1m/s$, their relative speed can be as high as $2m/s$. Similarly, the relative speeds of self-propelled vehicles such as AUVs, can go up to $4m/s$. The second reason for the high impact of mobility is the con-

siderable delay before nodes can transmit. This is because they need to randomly back off to avoid collisions which can be significant even with moderate number of contenders. With relative speeds of about 2m/s, nodes could drift tens of meters between transmissions. Given the speed of sound underwater, this translates to a timing uncertainty in the order of milliseconds. In this paper we will specifically focus on designing a time-sync protocol that is robust to mobility by incorporating physical layer information, namely an estimate of the Doppler shift.

The Doppler shift due to node mobility is one of the major impairments to underwater communication. Consequently, estimating the Doppler shift and compensating for it is a well-studied problem[9, 10, 11, 4, 5]. It has been shown that the Doppler shift can be estimated from a single packet exchange between a pair of nodes without the use of any additional hardware[11, 5]. Mason et al.[9] show that their Doppler estimation algorithm can estimate relative speeds up to 5m/s with standard deviation of 0.1 m/s using experimentally obtained data. Instead of estimating the Doppler information at the packet level, Nathan et al.[10] proposed a symbol by symbol Doppler rate estimation method for highly mobile underwater systems. This can potentially give us an even better estimate of Doppler velocity. We believe that the Doppler shift estimate as provided by physical layer algorithms can be a crucial input for time synchronization. Our new protocol, called D-sync, strategically exploits this feature to address the timing uncertainty due to node mobility as discussed earlier. Further, the accuracy of D-Sync outperforms existing time synchronization protocols.

Among existing time-sync schemes, TSHL[13] has the most energy efficient signaling scheme while MU-Sync[1] obtains the best performance. We propose a light-weight version of D-Sync, called B-D-Sync which uses the signaling scheme of TSHL while achieving the performance of MU-Sync. The accuracy achieved by D-Sync and B-D-Sync with few messages makes them promising solutions for underwater time synchronization. We begin by reviewing the existing work on underwater synchronization protocols.

2. RELATED WORK

Two unique characteristics of underwater sensor networks make underwater time synchronization challenging. The first one is large propagation delays. The second relates to the inherent mobility in underwater systems. Even for static underwater systems, sensor nodes tend to experience some degree of mobility due to ocean currents or wind. As a result, the propagation delay will not remain constant.

TSHL[13] is the first method specifically designed to deal with long propagation delays. Their synchronization protocol is organized in two phases. In the first phase they perform linear regression over timing information from multiple beacon transmissions so that nodes are skew synchronized. In the second phase, the clock offset is corrected by exchanging two-way messages. The fundamental assumption is that the distance and thus the propagation delay is a constant throughout the skew estimation phase. Essentially, they assume a static network which does not hold for most underwater systems.

To account for the time variability in the propagation delay due to the relative motion of nodes, MU-Sync[1] employs frequent two-way messaging to estimate both the offset and skew. In MU-Sync, the clock skew is estimated by performing linear regression twice over a set of local timing information collected via a two-way message exchange with a cluster-head. Because of using a large number of two-way messages, MU-Sync is not as energy efficient as TSHL. Furthermore, it assumes that the one-way propagation delay can be estimated as the average round trip time. However, for underwater mobile systems with nominal mobility, the estimate of the one-way propagation delay using MU-Sync becomes quickly biased. Finally, due to channel contention, nodes have to defer their transmission for random periods before responding to the cluster head. As the number of nodes increases, this time duration becomes longer, which significantly deteriorates the performance of MU-Sync.

The closest work to our proposed solution is Mobi-Sync[6], in which the spatial correlation of nodes' velocities is exploited to estimate the time varying propagation delay. Nodes are classified into three groups: surface buoys, super nodes, and ordinary nodes. It is assumed that surface buoys are equipped with GPS to obtain global time reference and super nodes can communicate directly with them to maintain synchronization. An ordinary node launches time synchronization by broadcasting request messages to its neighboring super nodes. Upon receiving the request message, each super node responds with a measurement of its absolute velocity. Nodes use a correlation model to estimate their velocity given the velocity of the super-node. 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. Also for networks with self-propelled vehicles, there may not be any correlation among neighboring nodes. On the contrary, our proposed scheme does not make any assumption about the underlying motion model nor does it require the motion correlation statistics of nodes for time synchronization. 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 in order to have a good estimate of velocity.

3. PROTOCOL DESCRIPTION

We begin by describing at a high level the operation of D-Sync. The goal of the protocol is to synchronize nodes to a single node within their transmission range, which we refer to as the beacon. The beacon essentially takes the role of a cluster-head and is responsible for estimating the clock offset and skew for all nodes that can communicate with it. The beacon can either be a super-node with a higher energy budget or it can be periodically elected among the nodes in the network. Once nodes are synchronized to the beacon, they are also synchronized to each other and network wide synchronization is achieved.

We first describe the basic signaling block when a single node has to synchronize to the beacon. For the network case, the same signaling pattern follows. Figure 1 summarizes the messages exchanged between a beacon node and an unsynchronized node B. We also introduce some important signaling parameters that we will refer to later in this section. As shown in Figure 1, a beacon initiates the synchronization process by broadcasting a request message and storing the send time of the message, $t_A(t_1)$. Node B records the receive time of the message as per its local time, $t_B(t_2)$. As mentioned earlier, the message sent by the beacon can be used by node B to estimate its relative speed based on the estimated Doppler shift[11, 9]. Now, since random ac-

cess methods are the preferred mode of Medium Access for mobile networks, node B has to back off for a random time, $T_{backoff}$ before it transmits a reply message back to the beacon. We will refer to this time as the response time of node B. (Later in Section 4 we will discuss the effect of the response time on the performance of D-Sync and MU-Sync). Finally, node B transmits a reply message to the beacon at time t_3, time stamped by its local send time, $t_B(t_3)$. As before, the beacon estimates its relative speed to node B from the reply message and also records the receive time of this message, $t_A(t_4)$. This process is repeated every $T_{message}$ (s) which is another signaling parameter that affects performance as we will discuss in Section 4. Given all the timing measurements and estimates of the relative node speeds obtained from Doppler, we will now derive the set of equations used by D-Sync to synchronize nodes.

We begin with a formal description and discussion of the synchronization problem, and continue to relate the unknown skew and offset of a node to the Doppler and timing measurements obtained from the messages described earlier.

The local time of any node A is related to the true global time, t by equation (1).

$$t_A(t) = \delta_A.t + \Delta_A \tag{1}$$

Where, $t_A(t)$ denotes the local time of node A at time t, δ_A is the clock skew and Δ_A is the clock offset.

The synchronization problem is to estimate the clock offset and skew for all nodes with respect to a beacon node. To derive our estimator in the light of existing schemes, we begin with a set of basic equations that relate the propagation delay to the local time of nodes, when a request and reply message is exchanged between two nodes.

Consider a message that is sent by node A at time t_1 and received by node B at time t_2. The propagation delay is related to the distance, $d_{AB}(t_1, t_2)$ defined as the distance between the position of node A at time t_1 and the position of node B at time t_2, equation (2). By substituting for t_1 and t_2 in terms of the local time of nodes A and B, as given by equation (1) and further assuming that node $A's$ clock is perfect, we obtain an equation that relates the skew and offset of node B to the unknown distance $d_{AB}(t_1, t_2)$, equation (3).

$$\begin{aligned} \tfrac{\|P_A(t_1) - P_B(t_2)\|}{c} &= \tfrac{d_{AB}(t_1,t_2)}{c} = t_2 - t_1 \\ \tfrac{d_{AB}(t_1,t_2)}{c} &= \tfrac{t_B(t_2) - \Delta_B}{\delta_B} - \tfrac{t_A(t_1) - \Delta_A}{\delta_A} \end{aligned} \tag{2}$$

$$t_B(t_2) = \delta_B.\left(t_A(t_1) + \frac{d_{AB}(t_1,t_2)}{c}\right) + \Delta_B \tag{3}$$

We can similarly obtain equation (4), when node B replies to node A at a later time, t_3 and node A receives this message at time t_4. This message exchange is summarized in Figure 1.

$$t_B(t_3) = \delta_B.\left(t_A(t_4) - \frac{d_{AB}(t_3,t_4)}{c}\right) + \Delta_B \tag{4}$$

Now, if the nodes are stationary, but the propagation delay is not negligible, as is the case in static underwater networks, then we have a total of 3 unknowns and 2 equations. Since the number of unknowns does not grow with the number of equations, this problem is solvable, by exchanging more messages[13]. However if nodes are mobile, the number of unknowns grows with the number of equations. This is because the relative distance between nodes also varies in time. Specifically, for n equations we have $n+2$ unknowns. Further the equations are not linear in the unknowns. To address this problem, protocols such as MU-Sync make the assumption that the relative node distance does not change during a message exchange and further use a coarse estimate of the skew to estimate the one-way propagation delay. We will show later in Section 4 how these assumptions affect the performance of MU-Sync.

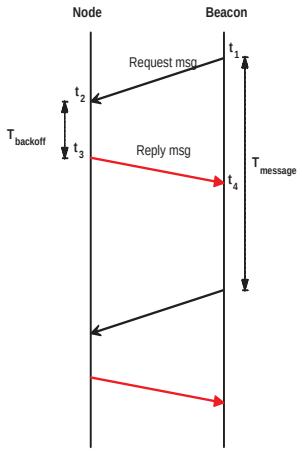

Figure 1: D-Sync Messaging Scheme

In D-Sync, we do not make the above assumptions. Instead we relate the clock offset and skew to the change in the relative distance of two nodes by combining equation (3) and equation (4) to obtain equation (5).

$$\begin{aligned} & t_B(t_3) + t_B(t_2) \\ &= \delta_B.\left(t_A(t_4) + t_A(t_1) + \tfrac{d_{AB}(t_1,t_2) - d_{AB}(t_3,t_4)}{c}\right) + 2\Delta_B \\ &= \delta_B.\left(t_A(t_4) + t_A(t_1) + \tfrac{d_{AB}(t_1,t_1) - d_{AB}(t_3,t3)}{c}\right) + 2\Delta_B \\ &+ \varepsilon_{motion} \end{aligned} \tag{5}$$

where ε_{motion} is the error due to node mobility:

$$\begin{aligned} \varepsilon_{motion} = \tfrac{\delta_B}{c} \cdot \{ & d_{AB}(t_1,t_2) - d_{AB}(t_1,t_1) \\ & + d_{BA}(t_3,t_3) - d_{BA}(t_3,t_4)\} \end{aligned}$$

Further analysis reveals that ε_{motion} can be upper bounded by:

$$\begin{aligned} |\varepsilon_{motion}| &\leq \tfrac{\delta_B}{c} \cdot |d_{AB}(t_1,t_2) - d_{AB}(t_1,t_1)| \\ &+ \tfrac{\delta_B}{c} \cdot |d_{BA}(t_3,t_3) - d_{BA}(t_3,t_4)| \\ &\leq \tfrac{\delta_B}{c^2} \cdot S \cdot (v_A + v_B) \end{aligned}$$

where S is the maximal transmission range and $v_{A(B)}$ is the maximal speed of node $A(B)$. We argue that in a message exchange, the two terms, $(d_{AB}(t_1,t_2) - d_{AB}(t_1,t_1))$ and $(d_{BA}(t_3,t_3) - d_{BA}(t_3,t_4))$, are likely to offset each other on average. By substituting typical values for S, v_A and v_B, worst case estimation error caused by ε_{motion} is around 1ms, which we consider to be acceptable for most underwater network systems. From here onwards, $d_{AB}(t_1,t_2)$ is approximated by $d_{AB}(t_1)$ for ease of presentation.

So far, we have obtained a relationship between the unknown skew and offset and the change in the distance between nodes during a request and reply message. We now formulate the estimation problem by relating the Doppler

measurements obtained at the end of each message to the change in the relative node distance. The Doppler shift is related to the relative speed of nodes by equation (6).

$$\frac{\Delta d_{AB}(t)}{\Delta t} = v_{AB}(t) = f_{doppler}(t).\lambda \tag{6}$$

The total change in distance during a request and reply exchange is given by equation (7).

$$\begin{aligned} d_{AB}(t_3) - d_{AB}(t_1) &= \int_{t1}^{t3} v_{AB}(t)dt \\ &= \lambda \int_{t1}^{t3} f_{doppler}(t)dt \end{aligned} \tag{7}$$

While the continuous Doppler shift is required to estimate the change in distance as shown in equation (7), we use only the two measurements obtained during a two-way exchange. Our estimate of the change in distance is given by equation (8).

$$\begin{aligned} & d_{AB}(t_3) - d_{AB}(t_1) \\ &= 0.5\left(v_{AB}(t_3) + v_{AB}(t_1)\right).(t_3 - t_1) + \varepsilon \\ &= \hat{v}_{AB}(t_1, t_3).\left(\frac{t_B(t_3) - \Delta_B}{\delta_B} - t_A(t_1)\right) + \varepsilon \end{aligned} \tag{8}$$

Where $\hat{v}_{AB}(t_1, t_3)$ is the average of $V_{AB}(t_1)$ and $V_{AB}(t_3)$.

Substituting equation (8) in equation (5), we obtain how the clock offset and skew is related to measurements of the relative speed of nodes, equation (9).

$$\begin{aligned} & t_B(t_2) + t_B(t_3)\left(1 + \frac{\hat{v}_{AB}(t_1,t_3)}{c}\right) \\ &= \delta_B.\left(t_A(t_4) + t_A(t_1)\left(1 + \frac{\hat{v}_{AB}(t_1,t_3)}{c}\right)\right) \\ &+ \left(2 + \frac{\hat{v}_{AB}(t_1,t_3)}{c}\right)\Delta_B + \varepsilon + \varepsilon_{motion} \end{aligned} \tag{9}$$

To summarize, when a request and reply message is exchanged between a beacon and an unsynchronized node, we can obtain a linear equation in 2 unknowns as given by equation (9). A linear estimator can then solve the set of equations obtained when N such messages are exchanged. The estimator of the clock skew and offset for the above problem is given by equation (10).

$$\begin{aligned} \hat{\Lambda} &= \left[\hat{\delta}_B, \hat{\Delta}_B\right]^T = \left(A^T A\right)^{-1} A^T Y \\ Y\left[i\right] &= t_B(t_2^i) + t_B(t_3^i)\left(1 + \frac{\hat{v}_{AB}(t_1^i,t_3^i)}{c}\right) \\ A\left[i,1\right] &= \left(t_A(t_4^i) + t_A(t_1^i)\left(1 + \frac{\hat{v}_{AB}(t_1^i,t_3^i)}{c}\right)\right) \\ A\left[i,2\right] &= \left(2 + \frac{\hat{v}_{AB}(t_1^i,t_3^i)}{c}\right) \end{aligned} \tag{10}$$

Where Y is a $N \times 1$ vector and A is a matrix with $N \times 2$ entries. The elements of Y and A are given in equation (10). We next describe the sources of error in D-Sync.

3.1 Error Analysis

There are two main sources of error in D-sync: the error due to random noise in Doppler measurements and the error due to the fact that Doppler measurements are not available continuously. As a result, the change in distance during a message exchange can only be estimated using the average of the two measurements obtained at the end of each message transmission, as described earlier. We will refer to the resulting error as the interpolation error. The timing error in equation (9) can be expressed in terms of afore mentioned errors as per equation (11). Equation (11) also shows the dependence of the error on the response time, $T_{backoff}$ shown in Figure 1. This response time affects the performance of MU-Sync as well which we will discuss in the next section.

$$\begin{aligned} \varepsilon &= (\varepsilon_{noise} + \varepsilon_{interp}).\left(\frac{t_3 - t_1}{\delta_B.c}\right) \\ &= (\varepsilon_{noise} + \varepsilon_{interp}).\left(\frac{T_{backoff} + T_{prop}}{\delta_B.c}\right) \\ & T_{backoff} = t_3 - t_2; T_{prop} = t_2 - t_1 \end{aligned} \tag{11}$$

$$\begin{aligned} \varepsilon_{interp} &= \frac{1}{t_3 - t_1}.\int_{t_1}^{t_3} v_{AB}(t)dt - \left(\frac{v_{AB}(t_1) + v_{AB}(t_3)}{2}\right) \\ &\leq \frac{(t_3 - t_1).\left(a_{\max}^2 - a_{avg}^2\right)}{4.a_{\max}} \end{aligned} \tag{12}$$

In Appendix A, we derive an upper bound for the interpolation error in terms of the maximum and the average relative acceleration of a pair of nodes, equation (12). We next compare the performance of D-Sync and MU-Sync over a set of relevant parameters.

4. COMPARISON BETWEEN D-SYNC AND MU-SYNC

In this section we will compare the D-Sync and MU-Sync via simulations for a pair of nodes. For this comparison we use the same signaling scheme as MU-Sync, which we described at the beginning of Section 3. Later, in Section 5 we will discuss the performance of the two schemes in a network setting.

In the following simulations, nodes follow smooth curved paths with a constraint on their maximum speed and maximum instantaneous acceleration. Further, all simulation results show the mean and standard deviation of the synchronization error 2 hours after a node is synchronized. A total of 10 request and reply messages are used for synchronization. The simulation parameters are summarized in Table 1 unless specified otherwise. We now describe the parameters that affect the performance of D-Sync and MU-Sync.

Table 1: Simulation Parameters

Notation	Value
packet length: L	60bytes
Data rate: R	$240bps$
Max distance: D	$1000m$
Max offset error	$0.03s$
Std. Doppler error	$0.1m/s^2$
$T_{backoff}$	$30s$
Slot length: t_{slot}	$0.976s$

4.1 Response Time

As mentioned earlier, the response time is defined as the time elapsed before an unsynchronized node replies to a request message sent by the beacon node, indicated by $T_{backoff}$ in Figure 1. While this time is assumed to be very small in MU-sync[1], it can be quite substantial in underwater networks. Due to the low data rates of underwater acoustic modems, the duration of packet transmission is in the order of hundreds of milliseconds. As the number of nodes competing to respond to a reference message increases, nodes will need to back off tens of seconds to avoid collision, depending on the number of contenders. Consequently, the response time of nodes also increases.

Figure 2 shows the mean and standard deviation of the synchronization error for MU-Sync and D-sync when the response time of a node is varied between $5s$ and $100s$. In our simulations, the maximum relative speed of the node is $2m/s$, which is nominal for self-propelled vehicles and corresponds to a higher mobility regime for devices that are not propelled.

We observe that both D-Sync and MU-Sync show an increasing trend with the response time, however for different reasons. In the case of MU-Sync the error increases because it assumes that nodes remain stationary during a message exchange. While in reality nodes would have moved a distance proportional to the response time. This translates to an error in the one-way propagation delay for MU-Sync. The error in D-sync increases with the response time because Doppler measurements are available only when a message is exchanged between nodes, as opposed to a continuous estimate. When the response time grows, so does the interpolation error as given in equation (12). However, D-Sync still outperforms MU-Sync because unlike MU-Sync, it takes node mobility into account during a message exchange. We will now compare the two schemes vs. network mobility.

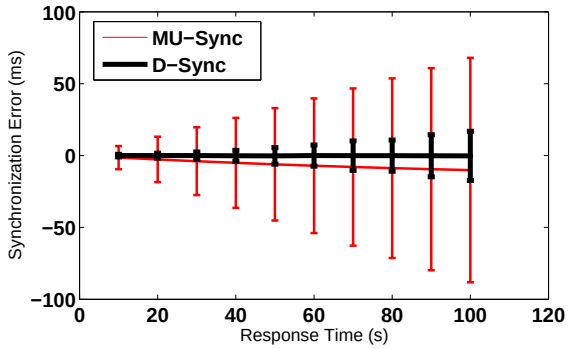

Figure 2: Performance with response time.

4.2 Extent of Mobility

We define the extent of mobility as the relative speed of nodes. While the performance of MU-Sync is affected directly by the relative node speed, the performance of D-Sync depends on the rate of change of relative speed as we have shown in Appendix A, equation (12). To further illustrate this point, we compare the two protocols via simulations when the maximum relative speed of a node is varied from $.01m/s$ to $5m/s$. The maximum relative acceleration was fixed at $.04m/s^2$. All other simulation parameters are given in Table 1.

From Figure 3 we observe that at low node speeds (below 1m/s), the performance of MU-Sync and D-Sync are comparable. This is expected, since the assumptions made by MU-Sync are valid at low mobility. However, with increased mobility, D-Sync significantly outperforms MU-Sync and has consistent performance across different speeds. On the other hand, the accuracy of D-Sync is affected by the relative acceleration of nodes. This is shown in Figure 4 where the maximum relative acceleration is varied between $.01m/s^2$ to $0.1m/s^2$, while the maximum relative speed is kept constant at $2m/s$. The error in MU-Sync had a standard deviation of around $200ms$ and did not vary with rel-

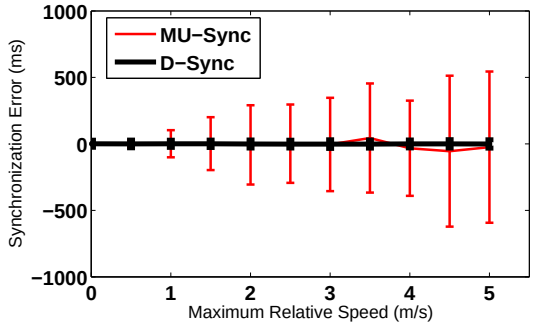

Figure 3: Performance of D-Sync and MU-Sync with relative node speed.

ative acceleration, so it is not shown in Figure 4. We next evaluate the robustness of D-Sync to the error in speed estimates obtained from Doppler.

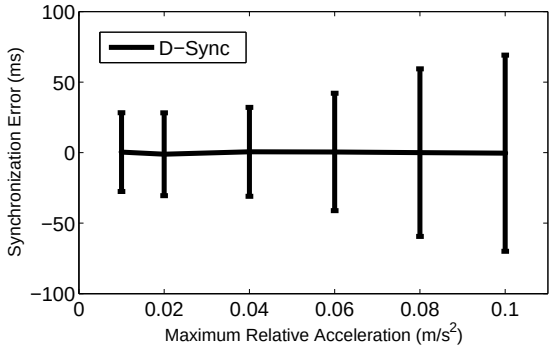

Figure 4: Performance of D-Sync with relative node acceleration.

4.3 Error in Doppler measurements

The error in the estimate of relative node speeds based on Doppler measurements only affects the performance of D-Sync. Figure 5 shows how the synchronization error of D-Sync varies when the standard deviation of the error in speed is increased from $.01m/s$ to $0.5m/s$. While the nominal reported error is $0.1m/s$[9], we observe that D-Sync still significantly outperforms MU-Sync at errors of above $0.4m/s$. The error in MU-Sync was once again around 200ms and showed no variation with the Doppler error as expected. We will next describe why the interval at which request messages are sent by the beacon affects synchronization performance.

4.4 Interval between request messages

In its basic form MU-sync recommends that the cluster head sends out a message every $5s$. Further, to achieve accurate synchronization, each node transmits a total of 25 messages. However, we recognize that estimating the skew is equivalent to estimating the slope of a line that best fits the timing data. To get a good fit, the dynamic range over which regression is performed should be far greater than the error in measurements. one-way of increasing this dynamic range is to allow enough time to elapse before initiating the

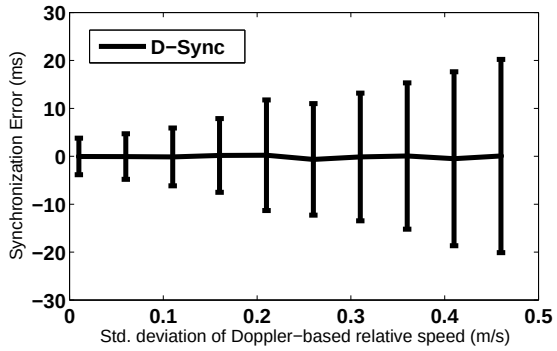

Figure 5: Performance of D-Sync with error in Doppler based relative speed estimate.

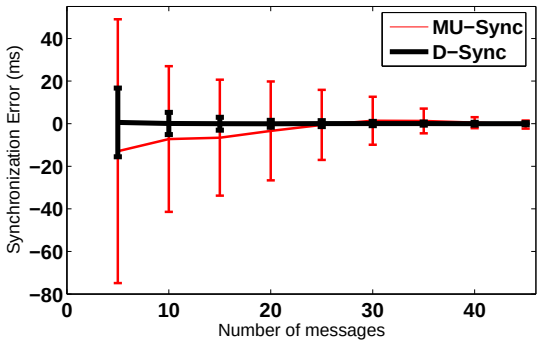

Figure 7: : Performance vs number of messages.

next request-reply exchange. Figure 6 shows how the performance of both MU-Sync and D-Sync can be improved by increasing the interval between request messages, while keeping the number of messages constant. The results show that increasing the message interval is a simple and effective way of improving synchronization performance without additional energy overhead.

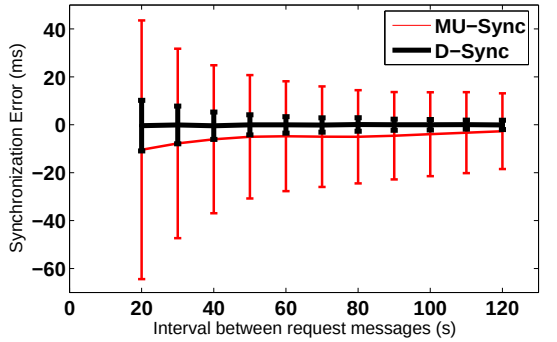

Figure 6: Performance vs message interval.

4.5 Total number of reference messages

Finally, we compare the two schemes when the total number of messages transmitted per node is increased from 5 to 45 as shown in Figure 7. While both protocols benefit from more messages, D-sync outperforms MU-sync especially when the number of messages transmitted per node is small. D-Sync is also able to achieve high accuracy, that is $10ms$ error for 2 hours after synchronization, using only 10 messages.

To summarize, we have shown that D-Sync outperforms MU-Sync over a range of relevant parameters. As a next step we propose a more light-weight version of our protocol, called Broadcast D-Sync (B-D-Sync) that is especially suitable as the network density increases.

5. BROADCAST D-SYNC

We have earlier shown that D-Sync can accurately synchronize a pair of nodes by exchanging several two-way messages. However, if we want to synchronize a group of nodes, then each node must transmit a reply message for every request message sent by the beacon. As a result, the total energy consumption can quickly ramp up. On the other hand, TSHL is known to be very energy efficient for synchronizing a group of nodes. Since TSHL assumes that the network is static, it can estimate the clock skew from a number of consecutive beacon broadcasts. Further, it requires only a single two-way exchange between the beacon and an unsynchronized node to obtain the unknown propagation delay and clock offset. As such, TSHL is not applicable to mobile systems because the propagation delay is no longer constant and the variation in the propagation delay is not known. However, we can strategically exploit Doppler information to overcome this problem as explained below.

Using Doppler measurements we estimate the change in the relative distance or in other words the change in the propagation delay. Instead of relying on two-way messages to obtain the propagation delay, we first obtain an accurate estimate of the one-way delay from a single two-way exchange. We then use Doppler measurements to estimate the one-way delay for every broadcast message transmitted by the beacon (see Figure 8). Once the propagation delays for all messages are known, we apply linear regression to jointly estimate the skew and offset. Therefore, nodes need to respond to only one beacon message to synchronize.

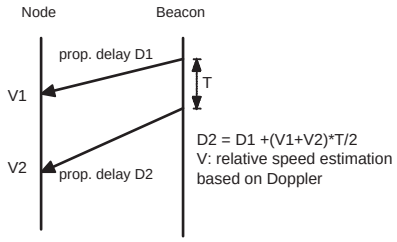

Figure 8: Relationship between consecutive propagation delays.

We now describe the messaging scheme used by B-D-Sync which is slightly different from TSHL. Instead of dividing the synchronization into two phases, we jointly estimate the skew and offset. Here we focus on a single hop network with n nodes. We organize the network into a beacon node and $n-1$ ordinary nodes. The beacon node is in charge of initiating the synchronization process by broadcasting a number of beacon messages. In addition, the beacon node will mark its last message to which nodes respond. Each unsynchronized

node records when a beacon message was received. After the last message is received, nodes send all their previous recorded timestamps to the beacon node in a response message. Finally, the beacon node computes the skew and offset for each unsynchronized node, and broadcasts the result to all nodes. To balance the energy consumption, nodes take turns in playing the role of the beacon. Figure 9 shows the detailed messaging scheme for B-D-Sync.

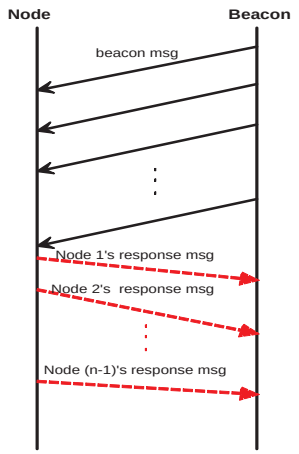

Figure 9: Broadcast D-Sync messaging scheme.

Next, we will evaluate MU-Sync and TSHL together with our proposed schemes in a network setting.

5.1 Performance Comparison and Discussion

In our simulation set up, nodes are placed in a $1000m$ by $1000m$ field. Each node can move on a smooth curved path in this field with a maximum relative velocity of $2m/s$ and a maximum relative acceleration of $0.1m/s^2$. The speed of sound is set to $1500m/s$ and remains constant throughout the synchronization process. The beacon node is assumed to have a precise clock, while the rest of the nodes have a skew of $80ppm$ and an offset of $0.000060s$. Nodes use a slotted contention based MAC protocol to gain channel access. Assuming a maximum initial synchronization offset of $0.03s$, the slot length t_{slot} is calculated based on the packet size, maximum synchronization offset, maximum distance between nodes and data rate as specified in Table 1. Given the above parameters, for a one-hop network with n nodes, each node chooses a random backoff in $[0, 10*(n-1)*t_{slot}]$ before it responds to a beacon message. The interval between consecutive beacon messages is set to $2s$ for B-D-Sync and THSL, and $(10*n*t_{slot}+5)s$ for D-Sync and MU-Sync, respectively. The multiplicative constant 10 is carefully chosen to ensure that all nodes can respond before the next beacon transmission. Finally, the Doppler-based relative speed measurements have a Gaussian error with standard deviation 0.1m/s, which is a typically suggested value[9].

As both D-Sync and MU-Sync rely on two-way messages, they will follow the same signaling scheme as explained in Section 3. Similarly TSHL and broadcast D-Sync share the same signaling scheme as illustrated in Figure 9. We now compare the performance of the four protocols in terms of accuracy and energy consumption, and show how they vary with network size.

We define accuracy as the mean of the absolute timing

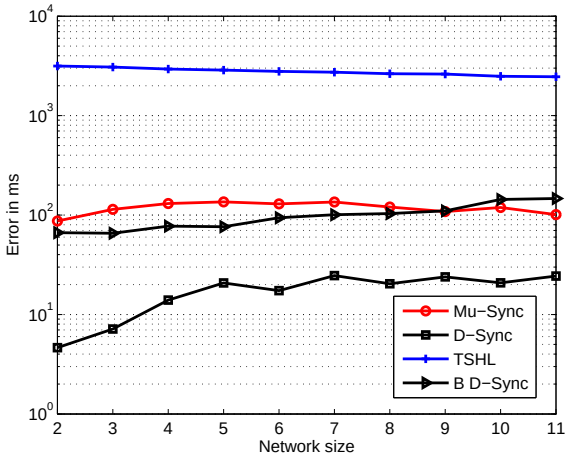

Figure 10: Comparison of the accuracy of protocols versus network size.

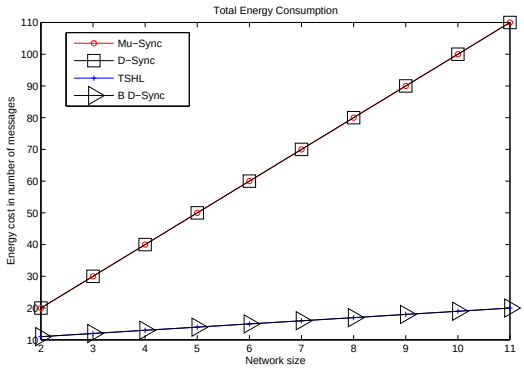

Figure 11: Comparison of different protocols in terms of energy consumption.

error, two hours after synchronization. We define the energy consumption as the total number of messages sent by all nodes in the network. Figure 10 shows the accuracy of each scheme, while Figure 11 depicts the corresponding energy consumption. From Figure 10 and Figure 11, we observe that D-Sync significantly outperforms MU-Sync however with identical energy consumption. The error in D-Sync increases from 7 to $20ms$ when the network size grows from 2 to 11. This makes D-Sync an ideal candidate to avoid frequent re-synchronization. Further B-D-Sync has a performance comparable to MU-Sync, however using far less energy.

Since D-Sync and MU-Sync use the same signaling scheme, their energy consumption curves entirely coincide. Similarly, TSHL and broadcast D-Sync have identical energy consumption. Based on Figure 11, it is clear that B-D-Sync and TSHL consume far less energy as compared to MU-Sync and D-Sync. When energy consumption per synchronization is a constraint, broadcast D-Sync would be a perfect candidate. We have also observed that by using some additional two-way messages the performance of Broadcast D-Sync can be significantly improved. This goes on to show that it may be possible to optimize the signaling scheme for B-D-Sync.

However, we leave this problem as part of future work.

6. CONCLUSION

Node mobility in underwater networks combined with low data rates of acoustic modems can have a significant impact on the performance of time-synchronization protocols. While the absolute velocity of nodes is hard to obtain underwater, information about their relative speed is a key input to time-synchronization. Incidentally, the Doppler shift caused by the relative motion of nodes is a very well studied problem since it is a major impairment to underwater acoustic communication. A number of effective physical layer techniques have been developed to estimate the Doppler shift. We have shown that our proposed time sync protocols strategically leverages Doppler information provided by the physical layer to achieve accurate time synchronization. Depending on how energy constrained nodes are, we believe D-Sync or B-D-Sync would be promising candidates for time synchronization in underwater networks.

7. REFERENCES

[1] N. Chirdchoo, W.-S. Soh, and K. C. Chua. Mu-sync: a time synchronization protocol for underwater mobile networks. In *WuWNeT '08: Proceedings of the third ACM international workshop on Underwater Networks*, pages 35–42, New York, NY, USA, 2008. ACM.

[2] J. Elson, L. Girod, and D. Estrin. Fine-grained network time synchronization using reference broadcasts. *SIGOPS Oper. Syst. Rev.*, 36(SI):147–163, 2002.

[3] S. Ganeriwal, R. Kumar, and M. B. Srivastava. Timing-sync protocol for sensor networks. In *SenSys '03: Proceedings of the 1st international conference on Embedded networked sensor systems*, pages 138–149, New York, NY, USA, 2003. ACM.

[4] M. Johnson, L. Freitag, and M. Stojanovic. Improved doppler tracking and correction for underwater acoustic communications. In *Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on*, volume 1, pages 575 –578 vol.1, 21-24 1997.

[5] B. Li, S. Zhou, M. Stojanovic, L. Freitag, and P. Willett. Multicarrier communication over underwater acoustic channels with nonuniform doppler shifts. *Oceanic Engineering, IEEE Journal of*, 33(2):198 –209, april 2008.

[6] J. Liu, Z. Zhou, Z. Peng, and J.-H. Cui. Mobi-sync:efficient time synchronization for mobile underwater sensor networks. In *WuWNeT '09: Proceedings of the fourth ACM international workshop on Underwater Networks*, New York, NY, USA, 2009. ACM.

[7] F. Lu, S. Lee, J. Mounzer, and C. Schurgers. Low-cost medium-range optical underwater modem: short paper. In *WUWNet '09: Proceedings of the Fourth ACM International Workshop on UnderWater Networks*, pages 1–4, New York, NY, USA, 2009. ACM.

[8] M. Maróti, B. Kusy, G. Simon, and A. Lédeczi. The flooding time synchronization protocol. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 39–49, New York, NY, USA, 2004. ACM.

[9] S. F. Mason, C. R. Berger, S. Zhou, and P. Willett. Detection, synchronization, and doppler scale estimation with multicarrier waveforms in underwater acoustic communication. *IEEE Journal on Selected Areas in Communications*, 26(9):1638–1649, 2008.

[10] N. Parrish, S. Roy, and P. Arabshahi. Symbol by symbol doppler rate estimation for highly mobile underwater ofdm. In *WUWNet '09: Proceedings of the Fourth ACM International Workshop on UnderWater Networks*, pages 1–8, New York, NY, USA, 2009. ACM.

[11] B. Sharif, J. Neasham, O. Hinton, and A. Adams. A computationally efficient doppler compensation system for underwater acoustic communications. *Oceanic Engineering, IEEE Journal of*, 25(1):52 –61, jan 2000.

[12] F. Sivrikaya and B. Yener. Time synchronization in sensor networks: A survey. *IEEE Network*, 18:45–50, 2004.

[13] A. Syed and J. Heidemann. Time synchronization for high latency acoustic networks. In *Proceedings of the IEEE Infocom*, page to appear, Barcelona, Spain, April 2006. IEEE.

APPENDIX

The interpolation error is given by:

$$\varepsilon_{interp} = \frac{1}{\Delta t}. \int_{t_1}^{t_1+\Delta t} v_{AB}(t)dt - \left(\frac{v_{AB}(t_1) + v_{AB}(t_1+\Delta t)}{2}\right)$$

The variation in the relative node speed must be such that it satisfies the two boundary data points obtained from Doppler measurements of the relative speed: $v_{AB}(t_1) = v_1$ and $v_{AB}(t_1 + \Delta t) = v_2$.

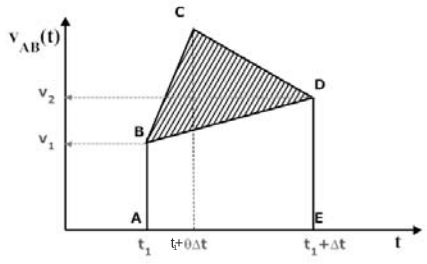

Figure 12: Relative speed of node vs. time

The maximum interpolation error occurs when a node moves with maximum relative acceleration of a_{max} until some time $t1 + \theta T$, and decelerates at a_{max} thereafter. This error is depicted in Figure 12 as the area of the triangular region BCD. The detailed derivation is omitted due to space limit. The maximal interpolation error ε_{interp} is given by:

$$\max\{\varepsilon_{interp}\} = \frac{1}{\Delta t}\left(area(ABCDE) - area(ABDE)\right)$$
$$= \frac{\Delta t\left(a_{\max}^2 - a_{avg}^2\right)}{4.a_{\max}}$$