ITUJournal
Future and evolving technologies

ITU Journal on Future and Evolving Technologies, Volume 7, Issue 1, March 2026

ARTICLE

Enhancement of geometry-based time synchronization (UWGS) with packet loss management under dynamic node trajectories in underwater acoustic networks

Matin Ghalkhani[1], Jianyu Zhang[1], Filippo Campagnaro[1,2], Michele Zorzi[1,2]

[1] Department of Information Engineering, University of Padova, Padova, Italy, [2] SubSeaPulse s.r.l., Padova, Italy

Corresponding author: Matin Ghalkhani, matin.ghalkhani@phd.unipd.it

In underwater networks, achieving precise time synchronization is essential for various critical functions, such as localization and channel access. However, synchronization remains a significant challenge due to substantial propagation delays and the mobility of nodes in underwater environments. While existing methods like D-Sync [1], Mobi-Sync [2], and MU-Sync [3] have been developed to address these issues in dynamic underwater settings, they primarily depend on estimating the radial velocity using Doppler shift effects. Additionally, these protocols often involve two-way message exchanges, which contribute to increased signaling overhead and response delays. In our previous study [4], we introduced a novel time synchronization approach based on the geometric shape of movement. This method, which we call Underwater Geometry-Based Synchronization (UWGS), employs a geometric framework to optimally estimate clock skew and offset for an unsynchronized node. Unlike traditional approaches that estimate radial velocity using Doppler shift, UWGS utilizes multiple one-way pulse signals to establish a geometric relationship between node velocity and range, and its preliminary evolution in [4] depicted simulated results in ideal conditions. In this paper, we further investigate the functionality of UWGS and compare its performance with existing methods, such as D-Sync, under more challenging and realistic conditions. Specifically, we consider the presence of random Gaussian noise in the motion of a mobile node and a non-ideal, slightly curved trajectory instead of the straight-line motion assumed in previous studies. Furthermore, we examine the impact of packet loss to assess whether UWGS provides improvements in synchronization accuracy and efficiency across a range of realistic underwater environments.

Keywords: Doppler-based time correction, geometry-based time synchronization, mobile underwater nodes, time synchronization, UWGS

1. INTRODUCTION

In recent years, Underwater Sensor Networks (UWSNs) have garnered significant interest from both academic and industrial communities due to their vast potential in applications such as oceanographic data collection, environmental monitoring, offshore exploration, disaster prevention, and military surveillance [5]. Accurate time synchronization among sensor nodes is essential, as discrepancies between local clocks can significantly degrade network performance. For instance, in distributed ocean current mapping systems [6], asynchronous sensor nodes may introduce substantial measurement errors, thereby reducing the accuracy of the collected data.

Time synchronization methods generally fall into two main categories. The first involves synchronizing to a universal external time source, most commonly Coordinated Universal Time (UTC), which is derived from Global Positioning System (GPS) or other Global Navigation Satellite Systems (GNSS). GPS receivers on the ground receive signals from multiple satellites equipped with atomic clocks. These signals are processed to determine GPS time, which is then converted to UTC by applying known leap second corrections

and offset values. Such receivers typically generate a Pulse Per Second (PPS) signal aligned with the UTC second, which is used by Stratum-1 Network Time Protocol (NTP) or Precision Time Protocol (PTP) grandmaster clocks for precise timekeeping. NTP clients perform timestamp exchanges (e.g., t_0, t_1, t_2, t_3) with these servers to estimate the clock offset θ and round-trip delay δ using statistical techniques. This allows the client to correct its local clock. NTP can achieve synchronization accuracy on the order of milliseconds over wide area networks (typically around 400 µs when GPS-based), while PTP enables much finer precision, ranging from microseconds down to nanoseconds, especially in local area network environments [7].

The second category focuses on synchronization to a designated reference node within the network itself. This approach is critical in scenarios where access to external timing sources like GPS is not feasible, such as underwater acoustic sensor networks. For instance, the Time Synchronization for High Latency (TSHL) protocol estimates clock skew and offset by employing a combination of one-way and two-way message exchanges, specifically designed to address the high latency typical of underwater acoustic communication [8]. Similarly, MU-Sync [3] employs a cluster-based linear regression technique using beacon messages to estimate synchronization parameters while compensating for mobility-induced timing variations. These in-network synchronization protocols are tailored to handle the challenges of multipath propagation, dynamic delays, and bandwidth limitations inherent in underwater environments.

In underwater acoustic sensor networks, where operations are highly time-sensitive, any clock misalignment can disrupt communication schedules and network coordination. This misalignment is primarily caused by clock drift, a common issue with crystal oscillator-based clocks, which tend to deviate over time. Therefore, periodic synchronization is necessary to correct these drifts and maintain network efficiency.

Addressing time synchronization in underwater environments poses unique challenges. Unlike terrestrial networks, where propagation delay is minimal due to the high speed of electromagnetic waves, underwater networks rely on acoustic signals, which travel much more slowly, making propagation delay a critical factor [9]. Additionally, the presence of variable medium access times, interrupt handling delays, and other nondeterministic behaviors further complicate precise synchronization [10]. The dynamic nature of the underwater environment, characterized by changes in temperature, salinity, and pressure, further contributes to variations in signal propagation speed, requiring adaptive synchronization mechanisms to mitigate these effects.

Achieving effective network time synchronization necessitates message exchanges between nodes to establish a shared time reference. However, the underwater environment introduces additional complexities, such as fluctuating propagation delays, which make high precision synchronization difficult, particularly in mobile scenarios. The movement of nodes in underwater environments leads to Doppler effects, which impact synchronization accuracy. Prior research has extensively examined the Doppler effect in underwater communication [1, 2, 5, 11, 12, 13]. The authors in [1] discussed these challenges in the context of underwater time synchronization, highlighting the importance of considering Doppler effects alongside high latency communication.

Furthermore, in underwater communication networks, the slow propagation speed of acoustic waves, combined with the movement of Autonomous Underwater Vehicles (AUVs) at speeds of approximately 3 m/s, can exacerbate synchronization errors due to mobility-induced delays. Most existing Doppler models are constrained to specific conditions, underscoring the need for more adaptive solutions. These challenges necessitate the development of more resilient synchronization frameworks that can dynamically adjust to variations in underwater conditions.

This work focuses on improving time synchronization in underwater acoustic networks by incorporating packet loss effects into the UWGS framework. Compared to the earlier study in [4], this paper examines more realistic operating conditions, including random Gaussian noise in the motion of a mobile node and slightly curved movement trajectories rather than ideal straight line paths. The synchronization performance of UWGS is systematically evaluated against the D-Sync method under different Packet Delivery Rates (PDRs). Furthermore, we introduce an alternative messaging scheme in which synchronization messages are transmitted continuously over the entire simulation duration, instead of being confined to predefined transmission slots as adopted in [4]. Packet losses in underwater environments substantially degrade the synchronization performance by increasing resynchronization intervals and reducing estimation accuracy. Despite these impairments, the simulation results in Section 4 show that UWGS maintains a clear performance advantage over the selected benchmark even in the presence of packet drop mechanisms.

2. RELATED WORK

Time synchronization is a critical component in Underwater Sensor Networks (UWSNs), necessitated by functionalities such as Medium Access Control (MAC), sleep scheduling, localization, and accurate time-stamping of sensor events. Unlike terrestrial RF networks, UWSNs rely on slow acoustic propagation (on the order

of 1 500 m/s) and operate over bandwidth-limited, error-prone channels; as a result, propagation delays are long, distance-dependent, and often *time-varying*. Mobility from currents and vehicles induces Doppler scaling and continually changes inter-node distances; meanwhile, environmental factors (temperature, salinity, depth) make sound speed, and thus one-way delay, non-stationary. These characteristics invalidate negligible delay assumptions common in RF-based protocols and bias RTT-based estimators via asymmetric and random queuing delays. Consequently, specialized schemes have emerged: delay-aware designs that explicitly model long latency (e.g., TSHL [8]); mobility-aware regressions or clustering (e.g., MU-Sync [3] and Mobi-Sync [14]); and Doppler-aware approaches that jointly estimate skew/offset under motion (e.g., D-Sync [1], DE-Sync [15], and TSMU [16]).

2.1 Time Synchronization for High Latency (TSHL)

The Time Synchronization for High Latency (TSHL) [8] protocol was among the pioneering efforts tailored for underwater environments characterized by significant propagation delays. TSHL operates in two phases. Initially, it estimates clock skew through linear regression on timing data from multiple beacon transmissions; subsequently, it corrects clock offset via a two-way message exchange. This method presumes a static network with constant inter-node distances, an assumption that often does not hold in dynamic underwater settings where nodes are subject to movement due to currents and other factors.

2.2 MU-Sync protocol

To address the limitations of TSHL in mobile scenarios, the MU-Sync protocol was introduced in [3]. MU-Sync employs a cluster-based approach, estimating clock skew by performing linear regression twice on local timing information gathered through two-way message exchanges with a cluster head. While this method enhances synchronization accuracy in the presence of mobility, it relies on the assumption that the one-way propagation delay is approximately half of the round-trip time. This simplification can introduce biases, particularly in networks with significant node movement, as varying distances between nodes affect the validity of this approximation over time. Additionally, as network size increases, channel contention leads to random transmission delays, further degrading MU-Sync's performance.

2.3 D-Sync protocol

The D-Sync protocol [1] was developed to further improve synchronization accuracy in mobile underwater networks by incorporating Doppler shift measurements. In D-Sync, a central beacon node broadcasts a request message to unsynchronized nodes, which respond after random delays. These exchanges provide timing measurements that account for variations in distance and relative velocity, allowing for the estimation of both clock offset and skew. By leveraging Doppler shift data, D-Sync effectively addresses the challenges posed by node mobility without assuming fixed inter-node distances, thereby enhancing synchronization precision in dynamic underwater environments.

2.4 Mobi-Sync protocol

Building upon previous efforts, the Mobi-Sync protocol [14] introduces the concept of spatial correlation among node velocities to estimate varying propagation delays. In this scheme, nodes are categorized into surface nodes equipped with GPS, super nodes that can communicate directly with surface nodes, and ordinary nodes. Surface nodes provide global time references, facilitating synchronization across the network. However, Mobi-Sync's applicability is constrained to scenarios where the exact correlation among neighboring nodes' velocities is known, a condition that is challenging to meet in many real-world underwater networks.

2.5 Doppler-Enhanced Synchronization (DE-Sync)

The Doppler-Enhanced Synchronization (DE-Sync) protocol [15] further refines synchronization accuracy by explicitly considering the effect of clock skew during the estimation of the Doppler scaling factor. Unlike D-Sync, DE-Sync utilizes the Doppler scaling factor directly in linear regression, leading to improved synchronization performance. This approach effectively mitigates the impact of initial clock skew on Doppler shift estimation, resulting in more accurate synchronization, especially in environments with significant node mobility.

2.6 Time Synchronization for Mobile Underwater Sensor Networks (TSMU)

The Time Synchronization for Mobile Underwater Sensor Networks (TSMU) protocol [16] leverages Doppler effects caused by relative motion between nodes during synchronization. It incorporates the Kalman filter and a calibration process to enhance synchronization accuracy. However, TSMU's reliance on accurate sound velocity

estimation introduces potential errors, as sound speed in water varies with depth, temperature, and salinity. Additionally, the necessity to adjust multiple parameters, including initial skew, affects both the accuracy and the efficiency of the synchronization process.

2.7 Probabilistic graphical model-based synchronization

Recent advances have explored novel mathematical frameworks for underwater time synchronization. [17] proposed an underwater network time synchronization method based on probabilistic graphical models in 2024. This approach utilizes positional and motion status information of sensor networks to construct a factor graph model for distributed network synchronization. The method adopts a distributed hierarchical network structure where nodes reduce the rate and energy consumption of acoustic communication while ensuring synchronization accuracy. By simplifying marginal probability calculations through message-passing algorithms, the probabilistic graphical model approach addresses the limitations of traditional centralized methods in large-scale networks. The factor graph representation enables efficient fusion of multiple information sources including reference position information, relative ranging measurements, and time delay differences, offering improved scalability and robustness in complex underwater environments with mobility and varying network topologies.

2.8 Lightweight synchronization for high-speed mobiles networks

Addressing the specific challenges of high-speed mobile underwater acoustic sensor networks, [18] introduced Lightweight Time Synchronization (LT Sync) in 2025. LT-Sync is designed for scenarios where nodes exhibit high mobility with low energy consumption requirements. The protocol adopts a Doppler-shift-estimating method to derive propagation delay when the unsynchronized node moves uniformly in a single direction. Unlike previous schemes such as Tri-Message, which perform poorly under node mobility, LT-Sync utilizes an underwater spread-spectrum method for signal reception to obtain accurate Doppler shift measurements. Simulation results demonstrate that LT-Sync achieves approximately 5 microseconds of synchronization error after 10^6 seconds of operation, significantly outperforming existing lightweight protocols. The scheme maintains consistent performance across different clock skews and node speeds while offering superior energy efficiency compared to TSHL and Tri-Message, making it particularly suitable for applications involving Autonomous Underwater Vehicles (AUVs) and other high-speed mobile platforms.

2.9 Joint Time Synchronization and Localization (JTSL)

Recognizing the interdependence between time synchronization and localization in underwater networks, [19] proposed a Joint Time Synchronization and Localization (JSTL) mechanism in 2023. The JTSL approach addresses the dual challenges of node mobility and time-varying sound speed in oceanic environments. The protocol employs three AUVs that send synchronization request signals to the moving unknown node three times, enabling the estimation of both clock skew and offset based on information transfer delays as sound speed changes. After achieving time synchronization, the unknown node receives location information from the AUVs and derives a sound speed change function. This function, combined with kinematic equations, enables accurate distance calculation between the unknown node and AUVs. Additionally, JTSL incorporates a multilayer fast meshing localization method to enhance both accuracy and speed of node localization. Simulation experiments demonstrate that JTSL effectively overcomes the difficulties associated with mobile node synchronization and localization, exhibiting excellent performance in terms of synchronization accuracy, energy efficiency, localization accuracy, and localization time. By jointly addressing both services, JTSL significantly reduces the number of required message exchanges, thereby conserving energy in bandwidth-limited underwater acoustic channels.

In summary, while various protocols have been developed to address the unique challenges of time synchronization in underwater sensor networks, each comes with its own set of assumptions and limitations. The choice of protocol depends on specific network conditions, including node mobility, propagation delays, and environmental factors.

A large number of underwater time synchronization system research works have established solutions based on the estimation of the Doppler factor of the underwater acoustic signal. However, due to the extremely low propagation speed of underwater acoustic signals and the complex and highly time-varying movement of the seawater medium itself, the ratio of the speed of the underwater vehicle to the speed of the underwater acoustic signal is larger. Consequently, the Doppler effect is also more significant and complex. Although there are many studies on suppressing the Doppler effect in underwater communications, there is a consensus in underwater mobile acoustic communications that the Doppler factor is non-uniform and time-varying. Using a single Doppler factor estimate as a description of the dynamics of an underwater mobile vehicle is insufficient to provide effective information, even in the radial velocity component relative to the acoustic platform along the acoustic path. To achieve more effective underwater mobile time syn-

chronization, we need to handle the Doppler effect using a more sophisticated motion model [15, 11, 20].

3. STRUCTURE OF UWGS

UWGS employs one-way messaging, providing a more efficient approach to synchronization compared to traditional underwater time synchronization protocols. By eliminating the need for acknowledgments during the synchronization process, UWGS is expected to offer faster and more energy-efficient synchronization.

Consider two nodes, as depicted in Fig. 1: a reference node (denoted as *Ref*) and another potentially unsynchronized node (denoted as *Node*).

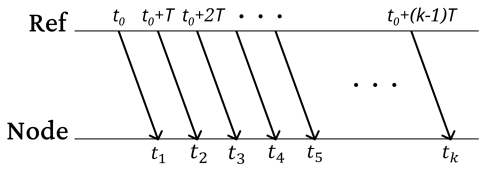

Figure 1 – Two-node synchronization setup with reference node *Ref* and moving node *Node*.

The synchronization process is initiated by the *Ref* node, which transmits messages at discrete time instants given by $t_0,\ t_0+T,\ t_0+2T,\ \ldots,\ t_0+(k-1)T$, where T denotes the fixed transmission intervals in seconds. These transmissions are received by the *Node* at times $t_1,\ t_2,\ t_3,\ \ldots,\ t_k$.

We now define a model that characterizes the transformation from the transmission time of the *Ref* node to the reception time at the mobile node *Node*. This model accounts for both signal propagation delay and the local clock's skew and offset at the *Node*. Specifically, the *Node*'s clock is modeled with a clock skew α and clock offset β.

We consider a mobile node (*Node*) that follows a time-varying trajectory while exchanging acoustic packets with a fixed reference node (*Ref*) located at $(x_R, y_R) = (0, 0)$. The *Node* starts from the initial position (x_0, y_0) at transmission reference time t_0 and moves downward with constant vertical velocity V, while simultaneously undergoing horizontal sinusoidal motion.

The trajectory of the *Node* is modeled as

$$x_N(t) = x_0 + A_x \sin(\omega t) + n_x(t), \quad (1)$$

$$y_N(t) = y_0 - V \cdot (t - t_0) + n_y(t), \quad (2)$$

where A_x and ω denote the amplitude and angular frequency of the horizontal oscillation, respectively, and $n_x(t) \sim \mathcal{N}(0, \sigma_x^2)$, $n_y(t) \sim \mathcal{N}(0, \sigma_y^2)$ represent independent Gaussian perturbations modeling positioning noise.

The signal transmitted at time t_x reaches the *Node* at time t_r, so the traveled distance satisfies [21]

$$p(t_r) = C\,(t_r - t_x). \quad (3)$$

Where C is the constant sound speed in this case. Considering the position of Node and Ref, we have

$$p(t_r) = \sqrt{\big(x(t_r) - x_R\big)^2 + \big(y(t_r) - y_R\big)^2}. \quad (4)$$

Substituting *(4)* into Equation *(3)*, we obtain:

$$C^2\,(t_r - t_x)^2 = \big(x(t_r) - x_R\big)^2 + \big(y(t_r) - y_R\big)^2. \quad (5)$$

To determine the receive time t_r, we solve Equation *(5)* numerically. Since the exact propagation equation depends on the receiver's motion and possible oscillations, it is generally implicit and does not admit a closed-form solution. Therefore, t_r is typically obtained using numerical root-finding techniques [22]. Initially, the solution is bracketed by estimating bounds from the receiver's extreme positions. Subsequently, iterative algorithms such as *bisection* or *Newton-Raphson* [23] are employed: a residual function is defined as the difference between the receive time predicted by the geometric model and the current candidate time, and the algorithm iteratively adjusts the candidate until the residual converges to zero. This procedure ensures that the computed t_r accurately captures the receiver's trajectory and oscillatory motion, even in the presence of small perturbations or noise.

The local clock of the *Node* is modeled as an affine transformation of the true reception time:

$$t_i' = \alpha \cdot t_i + \beta, \quad (6)$$

where α represents the clock skew and β denotes the clock offset.

As in prior underwater synchronization schemes such as MU-Sync [3] and D-Sync [1], the objective of synchronization is to estimate α and β from a sequence of received timestamps. However, unlike static or linear-motion models, the nonlinear propagation delay induced by sinusoidal motion and noise introduces a systematic bias in the arrival times, which directly manifests as estimation error in both clock skew and offset.

Fig. 2 illustrates the configuration under the assumption that the *Ref* node remains stationary during the synchronization process, while the *Node* moves downward, exhibiting slight bending and noisy motion. This assumption is justified by considering the *Node* to move at a constant velocity along its dominant direction, and

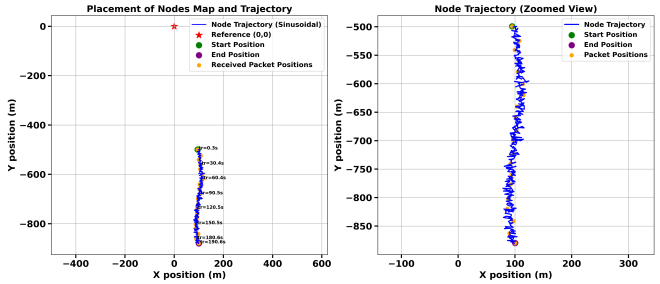

Figure 2 – Positions of a stationary reference node, *Ref*, and a moving node, *Node*, following a downward trajectory with noisy oscillations.

can be used as an approximation for different types of movement.

As *Node* advances downward, it sequentially receives the messages transmitted by *Ref*. For example, the first message sent at time t_0 is received at t_1, followed by the second message sent at $t_0 + T$, which is received at t_2, and so on. This process continues until the final message is received at t_k. In our protocol, we continue the procedure until we reach five successful messages. We then estimate α and β based on these five successful messages. These five messages are required to reliably estimate the clock skew and offset parameters.

By rotating Fig. 2 by 90° clockwise, we obtain Fig. 3, which presents a simplified view to enhance the clarity of node movement. This orientation was selected to better align with the geometric configuration used in the subsequent analysis. As illustrated in the figure, the segments p_n (for $n = 1, 2, \ldots, 5$) represent the distances traveled by the packets from the *Ref* node to the *Node*. The variable h denotes the perpendicular distance between the *Ref* node and the trajectory of the *Node*. Although the motion of the *Node* may exhibit a slight bend and noisy behavior, its trajectory is approximated as a straight line for the purpose of analysis. Under this assumption, V represents the constant velocity of the *Node* along its dominant direction of motion during the synchronization process.

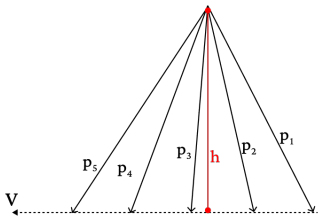

Figure 3 – Simplified representation of the *Node*'s movement.

To further analyze the *Node* movement and the synchronization process, we apply Stewart's geometric theorems. These theorems enable the incorporation of geometric principles into our real-world scenario involving underwater mobile nodes, in a manner similar to the approach taken in [21] for underwater node ranging.

In Fig. 4, let a, b, and c represent the sides of a triangle, with a cevian d drawn to side a. If the cevian divides a into two segments of lengths m and n, Stewart's theorem is applied as follows [24]:

$$b^2 m + c^2 n = a(d^2 + mn) \tag{7}$$

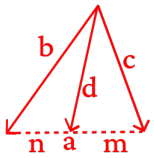

Figure 4 – Stewart's theorem.

Consider the first three packets, received at times t_1, t_2, and t_3. Assuming constant underwater sound speed C, the distances traveled by the packets are:

$$p_1 = (t_1 - t_0)C, \quad p_2 = (t_2 - t_0 - T)C, \quad p_3 = (t_3 - t_0 - 2T)C. \tag{8}$$

We can apply Stewart's Theorem [24] to solve for the speed of a mobile node:

$$p_1^2 (t_3 - t_2) V + p_3^2 (t_2 - t_1) V = \left(p_2^2 + V^2 (t_3 - t_2)(t_2 - t_1)\right)(t_3 - t_1) V \tag{9}$$

Hence, the mobile node's speed V can be expressed as:

$$V^2 = \frac{p_1^2}{(t_3 - t_1)(t_2 - t_1)} + \frac{p_3^2}{(t_3 - t_1)(t_3 - t_2)} - \frac{p_2^2}{(t_3 - t_2)(t_2 - t_1)}. \tag{10}$$

Similarly, for packets received at t_4 and t_5, we define:

$$p_4 = (t_4 - t_0 - 3T)C, \quad p_5 = (t_5 - t_0 - 4T)C, \tag{11}$$

and we can calculate V^2 using times t_2, t_3, t_4 or t_3, t_4, t_5:

$$V^2 = \frac{p_2^2}{(t_4 - t_2)(t_3 - t_2)} + \frac{p_4^2}{(t_4 - t_2)(t_4 - t_3)} - \frac{p_3^2}{(t_4 - t_3)(t_3 - t_2)}. \tag{12}$$

$$V^2 = \frac{p_3^2}{(t_5 - t_3)(t_4 - t_3)} + \frac{p_5^2}{(t_5 - t_3)(t_5 - t_4)} - \frac{p_4^2}{(t_5 - t_4)(t_4 - t_3)}. \tag{13}$$

Equating the right-hand sides of equations *(10)*, *(12)*, and *(13)*, we derive:

$$\frac{p_1^2}{(t_3-t_1)(t_2-t_1)} + \frac{p_3^2}{(t_3-t_1)(t_3-t_2)} - \frac{p_2^2}{(t_3-t_2)(t_2-t_1)} = \frac{p_2^2}{(t_4-t_2)(t_3-t_2)} + \frac{p_4^2}{(t_4-t_2)(t_4-t_3)} - \frac{p_3^2}{(t_4-t_3)(t_3-t_2)}. \tag{14}$$

$$\frac{p_1^2}{(t_3-t_1)(t_2-t_1)} + \frac{p_3^2}{(t_3-t_1)(t_3-t_2)} - \frac{p_2^2}{(t_3-t_2)(t_2-t_1)} = \frac{p_3^2}{(t_5-t_3)(t_4-t_3)} + \frac{p_5^2}{(t_5-t_3)(t_5-t_4)} - \frac{p_4^2}{(t_5-t_4)(t_4-t_3)}. \tag{15}$$

Since the Left-Hand Sides (LHS) of equations *(14)* and *(15)* are the same, it follows that their Right-Hand Sides (RHS) must also be equal. By setting LHS = RHS_1 for Equation *(14)*, and LHS = RHS_2 for Equation *(15)*, and by recalling the relationship described in Equation *(6)* between the received time t_n and local time t'_n at *Node*, $n = 1, \ldots, 5$, we can solve the resulting system of equations in the two remaining unknowns α and β. Hence, we obtain the following system of equations:

$$\begin{aligned} f_1(\alpha, \beta) &= \text{LHS} - \text{RHS}_1 = 0, \\ f_2(\alpha, \beta) &= \text{LHS} - \text{RHS}_2 = 0. \end{aligned} \tag{16}$$

To determine α and β, we use a numerical method based on five received timestamps t_1, t_2, t_3, t_4, t_5, each corresponding to a transmitted packet with index h_i, where $h_i \in \{1, 2, 3, 4, 5\}$. The transmission time of the i^{th} packet is given by:

$$t_A(h_i) = t_0 + (h_i - 1)T, \tag{17}$$

where $t_A(h_i)$ denotes the transmission time from the *Ref* node, T is the fixed time slot duration in seconds, and $t_0 = 0$.

The *Node*'s local timestamps follow the linear clock model based on equations *(6)* and *(17)*:

$$t'_i = \alpha \cdot (t_A(h_i) + p_i/C) + \beta. \tag{18}$$

To simplify estimation, we define:

$$a = \frac{1}{\alpha}, \qquad b = -\frac{\beta}{\alpha}, \tag{19}$$

Rearranging Equation *(18)* by dividing by α:

$$a \cdot t'_i + b = t_A(h_i) + p_i/C = (h_i - 1)T + p_i/C. \tag{20}$$

Define the residual for each timestamp [23]:

$$\Delta t_i = \hat{a} \cdot t'_i + \hat{b} - (h_i - 1)T - p_i/C. \tag{21}$$

Here, Δt_i represents the residual error for timestamp i, i.e., the difference between the estimated local timestamp and the true transmission time including propagation delay. When the clock parameters $(\hat{a}, \hat{b})$ are correctly estimated, all residuals Δt_i should approach zero.

To estimate $\hat{a}$ and $\hat{b}$, we construct a nonlinear system $\mathbf{f}(\hat{a}, \hat{b}) = \mathbf{0}$ from two algebraic functions based on the residuals. These functions are derived by substituting the residuals Δt_i (from Equation *(21)*) into the geometric constraints from Stewart's theorem (equations *(14)* and *(15)*) [23]:

$$\begin{aligned} f_1(\hat{a}, \hat{b}) = {} & \frac{(\Delta t_1)^2}{(t_3-t_1)(t_2-t_1)} + \frac{(\Delta t_3)^2}{(t_3-t_1)(t_3-t_2)} \\ & - \frac{(\Delta t_2)^2}{(t_3-t_2)(t_2-t_1)} - \frac{(\Delta t_2)^2}{(t_4-t_2)(t_3-t_2)} \\ & - \frac{(\Delta t_4)^2}{(t_4-t_2)(t_4-t_3)} + \frac{(\Delta t_3)^2}{(t_4-t_3)(t_3-t_2)}, \\ f_2(\hat{a}, \hat{b}) = {} & \frac{(\Delta t_1)^2}{(t_3-t_1)(t_2-t_1)} + \frac{(\Delta t_3)^2}{(t_3-t_1)(t_3-t_2)} \\ & - \frac{(\Delta t_2)^2}{(t_3-t_2)(t_2-t_1)} - \frac{(\Delta t_3)^2}{(t_5-t_3)(t_4-t_3)} \\ & - \frac{(\Delta t_5)^2}{(t_5-t_3)(t_5-t_4)} + \frac{(\Delta t_4)^2}{(t_5-t_4)(t_4-t_3)}. \end{aligned} \tag{22}$$

Each function $f_1(\hat{a}, \hat{b})$ and $f_2(\hat{a}, \hat{b})$ represents a consistency condition that equates two different expressions for the squared velocity V^2 derived from different triplets of packet reception times. Setting these functions to zero ensures that the estimated clock parameters $(\hat{a}, \hat{b})$ yield consistent velocity estimates across all available measurements. The detailed derivations of the Jacobian matrix and gradients are provided in Appendix A.

Solving this nonlinear system analytically is highly complex and practically infeasible due to the intricate structure of the functions and their derivatives. Therefore, we use a numerical method to approximate the solution $(\hat{a}, \hat{b})$ [23]. In Python, we can use `root` from

`scipy.optimize` [25], which supports a variety of solvers, including those that accept the analytical Jacobian. The underlying methods, such as the modified Powell hybrid algorithm [26] or Levenberg-Marquardt algorithm, employ Newton-Raphson type iterations that benefit from the analytical Jacobian. Providing the Jacobian (as derived above) improves convergence speed and accuracy.

The estimates for α and β are recovered from the solution $(\hat{a}, \hat{b})$ using:

$$\hat{\alpha} = \frac{1}{\hat{a}}, \qquad \hat{\beta} = -\frac{\hat{b}}{\hat{a}}. \tag{23}$$

The absolute synchronization errors are calculated as:

$$\begin{aligned} \text{Synchronization Error}_{\alpha} &= |\alpha - \hat{\alpha}|, \\ \text{Synchronization Error}_{\beta} &= |\beta - \hat{\beta}| \end{aligned} \tag{24}$$

where α and β are the true clock parameters (known in simulation but unknown in practice), and $\hat{\alpha}$ and $\hat{\beta}$ are their estimated values.

4. SIMULATION SETUP AND RESULTS

The simulations were conducted using the following parameters in Table 1.

Table 1 – Simulation parameters

Parameter	Description	Value
Clock Skew (α)	Normally distributed random variable	$\mathcal{N}(0, (60\,\text{ppm})^2)$
Clock Offset (β)	Uniformly distributed random number	$1\,\text{s} \le \beta \le 5\,\text{s}$
Sound Speed (C)	Speed of sound	1500 m/s
Node Speed (V)	Speed of the *Node*	$2\,\text{m/s} \le V \le 10\,\text{m/s}$
Ref Position	Coordinates of the *Ref* node	$(x_R, y_R) = (0, 0)$
Node Initial Position	Initial coordinates of the *Node* node	$(x_0, y_0) = (100, -500)$
Transmission Interval (T)	Time duration between each transmission	10 s
Back-off Timer (BT)	Duration of the back-off timer	10 s

After setting the simulation parameters, we implement two different message exchange strategies: a two-way ping-pong communication pattern for the D-Sync method and a one-way messaging approach for UWGS.

4.1 Without packet loss consideration

In D-Sync, packets are exchanged every 50 seconds. At the beginning of each 50-second time slot, a message is transmitted from the *Ref* node to the *Node*. Upon reception, the *Node* waits for a duration determined by the back-off timer before replying with a message to the *Ref*. This exchange results in four timestamp measurements, which are then used to perform the synchronization procedure as defined by the D-Sync protocol. This process is repeated every 50 seconds until a total simulation time of 1 000 seconds is reached. The entire simulation is then repeated for 20 iterations to ensure robust results. If we consider that each node has a local hardware clock modeled as in Equation *(6)*, the goal is to estimate α and β for the *Node* with respect to *Ref* using round-trip messaging. So, at the beginning of the simulation, *Ref* transmits a synchronization packet at time t_1 according to its local clock. *Node* receives the message at time t_2 (or in its local clock t_2'). After a fixed back-off duration BT, *Node* replies at time $t_3 = t_2 + BT$, and *Ref* receives the response at time t_4.

To model the propagation of messages from the *Ref* node to the *Node*, we define the *Ref-to-Node (RtN)* function. Assuming a constant speed of sound C and that the *Node* moves vertically with constant velocity V from the initial position (x_0, y_0), we introduce the auxiliary term

$$\kappa = x_0^2 + y_0^2 + V^2 t_0^2 - 2 y_0 V t_0, \tag{25}$$

and a distance-dependent propagation factor

$$\xi = \sqrt{(y_0 V - t_0 V^2)^2 + (C^2 - V^2)\kappa}. \tag{26}$$

The signal arrival time at the *Node*, accounting for both motion and propagation delay, is then expressed as

$$t_{\text{arr}} = t_0 + \frac{\xi - y_0 V + t_0 V^2}{C^2 - V^2}. \tag{27}$$

Accordingly, the local time measured by the *Node*, which incorporates its clock skew α and offset β, can be written as

$$\text{RtN}(t_0, \alpha, \beta) = \alpha \cdot t_{\text{arr}} + \beta. \tag{28}$$

For a two-way messaging scenario, a complementary function *Node-to-Reference (NtR)* can be defined to compute the expected reception time at the *Ref* node, given a timestamp t_3 at the *Node*. Using the node positions from *(1)* and *(2)*, this is expressed as:

$$\text{NtR}(t_3, \alpha, \beta) = \frac{\sqrt{x_N(t)^2 + \left(y_N(t) - V \cdot \frac{t_3 - \beta}{\alpha}\right)^2}}{C} + \frac{t_3 - \beta}{\alpha}. \tag{29}$$

To account for the Doppler shift and the relative velocity, we define the instantaneous radial velocity between the nodes as:

$$V_{ab}(t) = V \cdot \frac{y_N(t) - V \cdot t}{\sqrt{x_N(t)^2 + (y_N(t) - V \cdot t)^2}}, \tag{30}$$

where $x_N(t)$ and $y_N(t)$ are given by *(1)* and *(2)*, respectively. Using the timestamps and estimated velocities at transmission and reception, we define the following quantities:

$$\begin{aligned} V_{\text{avg}} &= \frac{V_{ab}(t_1) + V_{ab}(t_3)}{2}, \\ Y &= t_2 + t_3 \cdot \left(1 + \frac{V_{\text{avg}}}{C}\right), \\ A_1 &= t_4 + t_1 \cdot \left(1 + \frac{V_{\text{avg}}}{C}\right), \\ A_2 &= 2 + \frac{V_{\text{avg}}}{C}. \end{aligned} \tag{31}$$

Stacking multiple such synchronization exchanges results in a linear system:

$$A = \begin{bmatrix} A_1^{(1)} & A_2^{(1)} \\ \vdots & \vdots \\ A_1^{(j)} & A_2^{(j)} \end{bmatrix}, \quad Y = \begin{bmatrix} Y^{(1)} \\ \vdots \\ Y^{(j)} \end{bmatrix} \tag{32}$$

The parameter estimates are obtained by solving:

$$\begin{bmatrix} \hat{\alpha} \\ \hat{\beta} \end{bmatrix} = (A^\top A)^{-1} A^\top Y \tag{33}$$

At the final iteration, the synchronization error is calculated as shown in Equation *(24)*.

For the UWGS method, messages are transmitted from the *Ref* at regular intervals, starting at $t_0 = 0$. Each subsequent message is sent after a fixed duration T, such that messages are sent at $t_0 + T, t_0 + 2T, \ldots$, until the reception time at the *Node* reaches or exceeds 1 000 seconds. As with D-Sync, the UWGS simulation is repeated for 20 iterations to gather reliable synchronization performance metrics. We follow the setup as we already have described in Section 3, and Table 1.

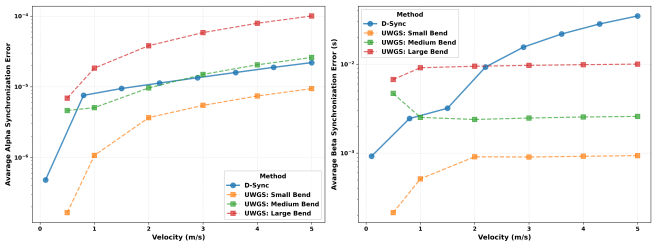

Figure 5 – Average synchronization error as a function of *Node* movement speed for both D-Sync and UWGS methods. The left plot shows the clock skew error α, while the right plot illustrates the clock offset error β (in seconds). For UWGS, different levels of trajectory bending are shown.

Figure 5 presents a comparative analysis of synchronization error versus node velocity for both the D-Sync and UWGS methods. For D-Sync, as the velocity of the *Node* increases from 0.2 m/s to 5 m/s, both the alpha and beta synchronization errors exhibit a monotonically increasing trend. This degradation arises from the structure of the least-squares estimation in Equation *(33)*, where the design matrix A in Equation *(32)* is constructed from timestamps and the velocity-dependent term V_{avg}. As node velocity increases, the propagation asymmetry between the forward and return paths grows, introducing larger Doppler-induced biases in V_{avg} that are not fully compensated by the two-way exchange. This causes the columns of A to become increasingly corrupted by modeling error, thereby degrading the conditioning of the system and amplifying the sensitivity of the numerical solution to measurement noise. Consequently, both $\hat{\alpha}$ and $\hat{\beta}$ estimates deteriorate progressively with velocity, with the alpha error growing from approximately 10^{-7} at low velocity to 2×10^{-5} at 5 m/s, and the beta error increasing from the sub-millisecond range to approximately 10^{-1} s.

In contrast, the UWGS method demonstrates different error characteristics under comparable velocity conditions, starting from 0.5 m/s. This lower velocity threshold is necessary because UWGS requires the formation of geometric triangles by the moving node, exploiting the spatial geometry of node motion for synchronization. As shown in the figure, the synchronization error generally increases with node velocity. A significant portion of the error can be attributed to the trajectory characteristics of the moving *Node*. Specifically, when the node follows a slightly curved trajectory, modeled as sinusoidal motion with an amplitude of up to 3 m, the synchronization accuracy is comparable to that of D-Sync, despite UWGS employing a more complex geometric structure. With increased trajectory curvature, where the sinusoidal amplitude rises to 5 m or even 10 m, the synchronization error remains below 10^{-4} for node velocities approaching 5 m/s in the estimation of α, while the error in β remains within 10 ms.

The synchronization errors in UWGS arise from multiple sources. First, motion-induced noise significantly affects the overall accuracy, as deviations from the assumed constant-velocity model introduce geometric uncertainties. Second, trajectory characteristics, particularly the degree of bending, directly influence how well the geometric triangle formation captures the true motion. As we saw in the comparison with D-Sync, UWGS exhibits sensitivity to trajectory bending, with the small-bend configuration achieving significantly lower alpha and beta errors across all velocities, while large-bend trajectories result in substantially higher errors. Additionally, utilizing only a finite number of packet timestamps introduces discretizations effects as estimation is performed on a sampled representation of the trajectory rather than on the continuous model. For better approximation, more than five packets can be used; however, we intentionally demonstrate the performance under only five

successfully received packets. The propagation delay computation, which depends on the instantaneous receiver position obtained through numerical root-finding of the implicit equation in Equation *(5)*, accumulates these approximation errors across all five packet measurements used in the synchronization process.

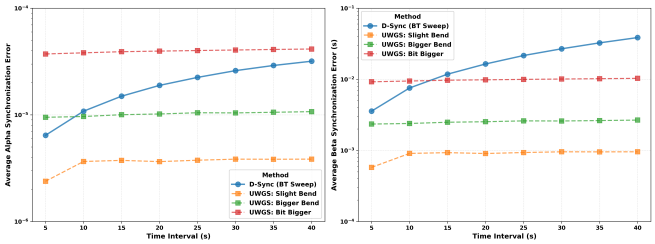

Figure 6 – Average synchronization error as a function of transmission time interval for both methods. The left plot shows the error for α, while the right plot illustrates the error for β (in seconds). For D-Sync, the interval is the Backoff Time (BT), while for UWGS it is the transmission interval T under different trajectory bending conditions.

The behavior observed in Fig. 6 can be explained by examining the regression systems constructed in each method. For D-Sync, the nearly linear growth of the error with increasing Backoff Time (BT) arises from two factors. First, as BT increases, fewer message exchanges occur within the observation window, which reduces the number of equations available for the least-squares estimation and degrades the solution accuracy. Second, the affine clock model of Equation *(6)* implies that errors scale with elapsed time: skew errors grow slowly but proportionally with the interval length, while offset errors accumulate directly with each longer back-off period. Consequently, α errors remain small (order 10^{-5}), whereas β errors increase more significantly with BT.

In contrast, the transmission interval T has minimal impact on UWGS synchronization accuracy. In this experiment, the node velocity is fixed at $V = 2$ m/s for all cases. The results indicate that increasing the interval T produces negligible changes in synchronization error. This behavior can be attributed to the constant velocity assumption maintained during the synchronization process, under which movement noise and trajectory characteristics are primarily influenced by changes in velocity rather than by the transmission interval duration. Consequently, once five successful packets are received, synchronization can be reliably performed across a wide range of time-slot durations. Although variations in T under slight bending conditions yield synchronization errors comparable to those of the D-Sync case, this behavior persists even under more complex motion patterns. In practical deployments, however, the choice of T must be made judiciously; it should be sufficiently large to reduce the impact of estimation noise, while remaining short enough to maintain the validity of the constant-velocity assumption within each synchronization interval.

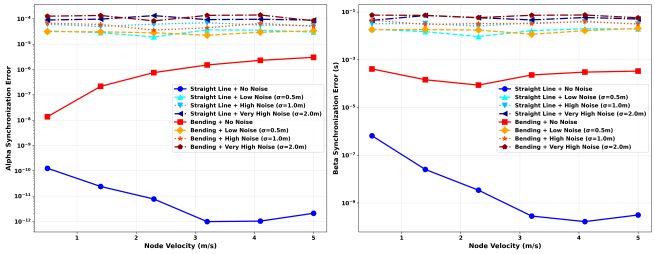

Figure 7 – Alpha (left) and Beta (right) synchronization error of the UWGS versus node velocity under straight-line motion, trajectory bending, and additive Gaussian noise.

Fig. 7 illustrates the performance of UWGS under different operating conditions, including variations in the movement trajectory (i.e., bending) and the presence of measurement noise. The impact of additive Gaussian noise manifests as an approximately horizontal trend in the synchronization error, with distinct error levels corresponding to different noise variances. Similarly, trajectory bending introduces a noticeable degradation in synchronization accuracy, reflecting the increased geometric uncertainty induced by non-linear motion. It is worth noting that a non-zero synchronization error is observed even under ideal conditions, namely straight-line motion in the absence of noise. In the presented simulations, trajectory bending is modeled as a lateral deviation of 10 m.

Despite these limitations, the synchronization accuracy achieved remains well within acceptable bounds for practical underwater networking applications. The synchronization error is consistently small enough to support Medium Access Control (MAC) layer coordination, including time-slotted access schemes and collision avoidance mechanisms. Furthermore, in ranging applications, the resulting timing inaccuracies translate into distance estimation errors that are nearly always below 1 m, even under noisy and non-ideal trajectory conditions. This level of ranging precision is sufficient for a wide range of underwater localization and navigation tasks, highlighting the robustness of the proposed UWGS approach under realistic operating conditions [27].

4.2 With packet loss consideration

A packet dropping mechanism emulates packet loss due to underwater noise, interference, or channel fading. It introduces uncertainty in the estimation process and reflects realistic underwater communication constraints. The variation in Packet Delivery Rate (PDR) allows an analysis of the robustness and accuracy of the synchronization algorithm under realistic channel conditions.

To evaluate the performance of our proposed synchronization method and compare it with the D-Sync protocol, we simulate the effect of packet loss by randomly drop-

ping messages. In the D-Sync protocol, time is divided into synchronization slots of fixed duration. Each slot is considered valid only if all four required timestamps are successfully received; if at least one message is lost within a slot, the entire slot is discarded and does not contribute to the estimation of clock skew and offset.

The simulation setup is identical to the one described in Table 1. Each configuration is evaluated over 100 independent simulation runs to ensure statistical reliability. The simulation spans a total duration of 2 000 seconds. Only slots in which all required messages are successfully delivered are used to compute the synchronization matrices presented in Equation *(31)*; failed slots are excluded from the estimation process.

The results are shown in Fig. 8, which illustrates the average synchronization error for α and β (in seconds) as a function of node velocity ranging from 0 to 5 m/s.

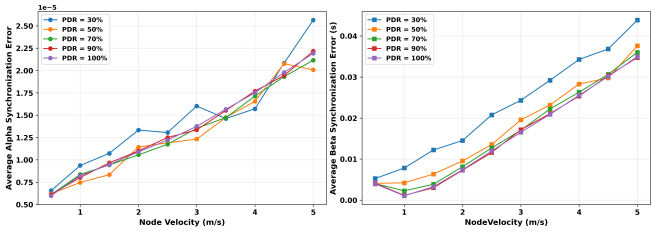

Figure 8 – D-Sync average synchronization error vs. Velocity (V): The left plot depicts the error for α, while the right plot shows the corresponding error for β in seconds.

As observed, the synchronization performance deteriorates as the PDR decreases. Nevertheless, the protocol continues to function reliably within a practical range. At PDR values of 10% and up to around 50%, we observed instances during the simulation time where we did not receive enough packets for estimating α and β, causing the numerical method to fail to converge.

To further analyze the impact of timing, we fix the velocity at 2 m/s and vary the back-off timer duration. The corresponding results are shown in Fig. 9. It is worth noting that the convergence issue at PDR values of 10% to around 50% persists in this scenario as well.

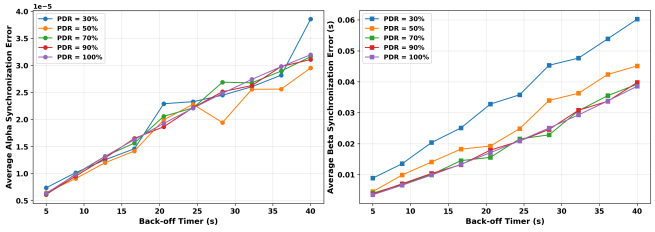

Figure 9 – D-Sync average synchronization error vs. Back-off Timer (BT): The left plot depicts the error for α, while the right plot shows the corresponding error for β in seconds.

For the UWGS protocol, we employ the same simulation configuration used in the ideal scenario without packet loss, as outlined in Table 1. Packet arrival times at the *Node* are computed using the function defined in Equation *(28)*, which incorporates clock skew and offset effects. To simulate the unreliable nature of underwater communication, packets are probabilistically dropped according to a predefined PDR.

Each synchronization packet is transmitted periodically every T seconds, representing the time slot interval according to the local clock of the *Ref* node. Due to the clock skew α and offset β, the actual arrival time of each packet at the *Node* is affected, and is computed using Equation *(6)*.

Given a predefined PDR, the simulation continues until at least five packets are successfully received. Once the first five valid arrival times are obtained (as shown in equations *(17)* and *(18)*), the UWGS protocol estimates the synchronization parameters by solving a non-linear system of equations. This system arises from the quadratic interpolation constraints detailed in Section 3.

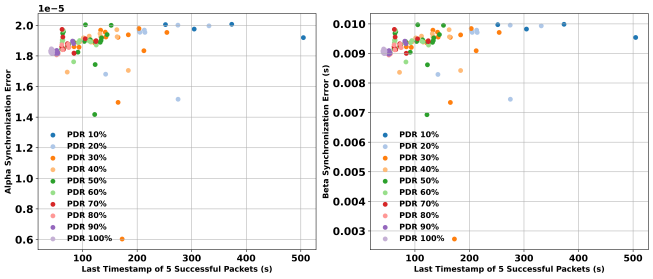

Figure 10 – UWGS synchronization error vs. Last timestamp of 5 successful packets used for estimation(s): The left column shows for α, while the right column shows the corresponding results β (in seconds).

As illustrated in Fig. 10, increasing the PDR enables the synchronization process to complete more quickly, since the required number of packets (five) is likely to be received sooner. The PDR is defined over the entire synchronization simulation period, and packets are randomly dropped according to this rate. For example, a PDR of 10% corresponds to a 90% probability of packet loss. At first sight, one may expect that the synchronization accuracy should not depend on the Packet Delivery Ratio (PDR), since in principle the recovery of α and β is exactly solvable once five successful packets are available. However, the plots show that higher PDR values actually lead to larger estimation errors. The key reason is that with a high Packet Delivery Rate (PDR), the first five successful packets are almost always the first five consecutive transmissions, which are closely spaced in time. Using such clustered timestamps in the synchronization algorithm provides only a limited geometric spread, similar to a small baseline in an Ultra-Short Baseline (USBL) array, which reduces angular resolution and makes the

system more sensitive to noise, leading to higher relative positioning errors [28]. When PDR is lower, it takes longer to collect five successful packets, and these timestamps are more spread out over time. This larger temporal spacing effectively increases the "baseline" of the measurements, analogous to the wide beacon spacing in a Long Baseline (LBL) system, which strengthens the geometric configuration, improves stability, and reduces the amplification of errors in the nonlinear solver [28]. As a result, even though the absolute error of the estimated parameters remains approximately the same, the relative error decreases due to the improved geometry, highlighting how both temporal distribution and baseline size play a critical role in synchronization and positioning accuracy. Since the synchronization algorithm relies on the first five successfully received packets, the procedure can continue even under lower PDR conditions, as long as five successful packets are eventually received. By extending the simulation time, it is ensured that five successful packets are received even at low PDR, albeit after a longer waiting period. Thus higher PDR accelerates the acquisition of packets, but yields a weaker estimation geometry, whereas lower PDR delays synchronization but improves accuracy due to the greater temporal diversity of the chosen packets.

Fig. 11 illustrates the simulation results obtained under different Packet Delivery Rate (PDR) levels, where T denotes the transmission interval as defined in Table 1. The figure depicts the average synchronization error of the clock skew α and the clock offset β (in seconds) as a function of the node velocity, which varies from 0.5 to 5 m/s. The left column corresponds to the synchronization error in α, while the right column presents the corresponding error in β. To ensure statistical reliability and minimize fluctuations in the plots, each configuration is simulated 100 times, and the average synchronization error across all runs is plotted against the transmission interval. This setup assumes small ambient noise and bending effects with an oscillation amplitude of up to 10 m. In the presence of packet errors, the average synchronization error increases with increasing velocity. This behavior can be attributed to the fact that higher node speeds result in greater vertical displacement during the period before the first five successful packets are received, which increases the positional uncertainty and degrades the accuracy of the geometric synchronization process. Under higher PDR conditions, the first five successful packets are received more quickly, allowing the synchronization procedure to complete in less simulation time with the *Node* having traveled a shorter distance. However, as PDR decreases, the reception of the first five successful packets is delayed, during which the *Node* continues to move. This extended period before completing the initial packet collection increases the difficulty of accurately estimating the clock offset and skew, as the *Node*'s position has deviated further from its reference trajectory, ultimately reducing synchronization accuracy and resulting in higher synchronization error. The convergence of β synchronization error at higher velocities stems from motion dynamics dominating packet availability effects. At low velocities (0.5–2 m/s), sparse packet sampling creates different available timestamps for each PDR level, increasing sensitivity and error spread. At higher velocities (3–5 m/s), the strong time–distance signature from substantial node displacement governs synchronization parameters, making PDR influence secondary to velocity-induced dynamics and causing β errors to converge across PDR levels. This timescale separation effect masks packet sparsity on the estimation process.

In our previous work [4], an increase in synchronization error with the *Node* velocity was also observed. However, that study assumed a TDMA-like communication structure in which five packets were transmitted at fixed 50-second intervals. In contrast, the current framework continuously transmits packets until the first five successful receptions are obtained. This adaptive packet acquisition strategy enhances synchronization robustness and improves accuracy, particularly at higher velocities.

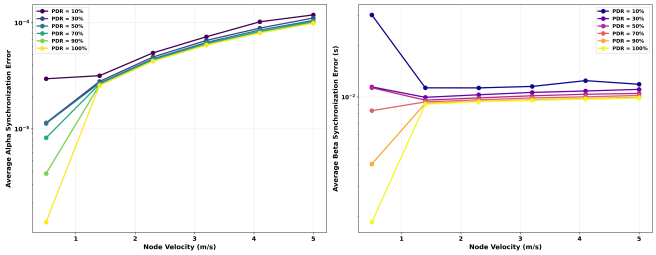

Figure 11 – Average synchronization error versus velocity using the UWGS approach. The left column shows the clock skew error α, while the right column presents the corresponding clock offset error β (in seconds), evaluated under different Packet Delivery Rate (PDR) levels.

Similar to the D-Sync configuration used to analyze the synchronization performance as a function of the back-off timer, the UWGS framework evaluates the performance with respect to the transmission interval. By fixing the *Node* velocity at 2 m/s and varying the transmission interval from 5 to 50 seconds, the corresponding simulation results are shown in Fig. 12. To ensure statistical reliability and minimize fluctuations in the plots, each configuration is simulated 100 times, and the average synchronization error across all runs is plotted against the transmission interval. This setup assumes small ambient noise and bending effects with an oscillation amplitude of up to 10 m.

The results exhibit a trend analogous to that observed in the velocity-based analysis. Specifically, the average synchronization error for both α and β increases as the transmission interval becomes longer. Since the synchronization algorithm relies on the first five successfully received packets, the procedure can continue even under lower PDR conditions, as long as five successful packets are eventually received. By extending the simulation time, it is ensured that five successful packets are re-

ceived even at low PDR, albeit after a longer waiting period. However, longer transmission intervals result in larger temporal separations between received packets. During this extended synchronization period, the *Node* continues to move, increasing the distance between the *Node* and the *Ref* point, which in turn degrades the estimation accuracy and leads to higher synchronization error. Conversely, with shorter transmission intervals, the synchronization procedure completes more rapidly, limiting the *Node* displacement during the process and thereby yielding better synchronization accuracy.

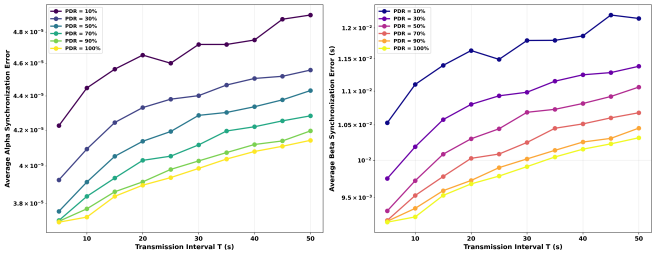

Figure 12 – Average synchronization error versus the transmission interval using the UWGS approach. The left column presents the clock skew error α, while the right column shows the corresponding clock offset error β (in seconds), evaluated under different Packet Delivery Rate (PDR) levels.

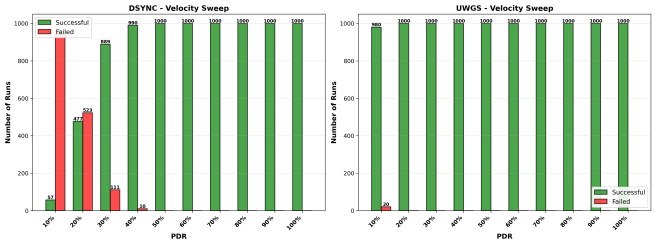

Figure 13 – Velocity sweep comparison of D-Sync and UWGS convergence performance across PDR variations

Fig. 13 presents a side-by-side comparison of D-Sync and UWGS performance during velocity sweep simulations across ten PDR levels (10%–100%). The results demonstrate markedly different robustness characteristics between the two algorithms.

D-Sync exhibits significant PDR dependency, with convergence improving gradually from 57 successful runs (6.12% of 1000) at 10% PDR to 1000 successful runs (100%) at 50% PDR and above, yielding an overall success rate of 83.96%. Specifically, the algorithm shows notable instability at lower PDR values: at 10% PDR, 943 runs fail; at 20% PDR, 523 runs fail; and at 30% PDR, 111 runs fail. Only beyond 40% PDR does D-Sync achieve consistent perfect convergence.

In contrast, UWGS maintains robust performance across all PDR levels, with 980 successful runs (98%) already at 10% PDR and achieving 1000 successful runs (100%) from 20% PDR onward, resulting in an overall success rate of 99.82%. The algorithm exhibits only 20 failures at 10% PDR and perfect convergence thereafter.

This performance gap stems from fundamental structural differences. In UWGS, clock estimation begins immediately upon receiving five successful packets, rendering the algorithm adaptive and inherently resilient to packet loss. The algorithm does not depend on achieving a minimum packet reception threshold within a fixed time window. Conversely, D-Sync relies on packet reception during a fixed simulation interval and requires a sufficient number of successfully received packets within that window to estimate the clock parameters α and β. Low PDR conditions frequently prevent this prerequisite from being met, resulting in convergence failures.

The quantitative advantage is substantial: UWGS achieves a 15.86% higher overall success rate and demonstrates superior robustness under degraded network conditions. This makes UWGS significantly more suitable for practical deployments where network reliability may be limited.

5. SUMMARY AND CONCLUSION

In our previous work [4], we presented the core principles of the UWGS protocol and evaluated its performance through comparisons with existing time synchronization schemes for mobile underwater nodes, including D-Sync. That study assumed an ideal communication environment without packet losses and modeled node mobility using simplified motion assumptions.

In the present work, we significantly extend this analysis by incorporating more realistic and challenging conditions. In particular, the UWGS method is evaluated under sinusoidal motion trajectories with different levels of curvature (bending), as well as in the presence of motion-induced noise. These conditions are designed to better represent practical underwater mobility patterns. In addition, a packet dropping mechanism based on a predefined Packet Delivery Ratio (PDR) is introduced to emulate realistic underwater acoustic communication impairments.

Our comprehensive simulation studies demonstrate that UWGS achieves a synchronization accuracy comparable to the established two-way D-Sync protocol while offering significant advantages in communication efficiency and energy consumption. Several key findings emerge from our experimental results. First, UWGS maintains clock skew errors on the order of 10^{-4} and clock offset errors in the millisecond range across a wide range of operating conditions, including node velocities up to 5 m/s and trajectory bending amplitudes up to 10 m. Second, unlike D-Sync, whose accuracy degrades linearly

with increasing back-off timer duration, performance of the UWGS remains largely independent of the transmission interval T, provided that the constant-velocity assumption holds within each synchronization window.

Third, the protocol demonstrates robust synchronization performance under realistic packet loss conditions, with an interesting trade-off: lower Packet Delivery Rates (PDR) actually improve the synchronization accuracy by creating wider temporal spacing between successfully received packets, analogous to the improved geometric dilution of precision observed in positioning systems with larger baselines. Despite the combined effects of packet loss, complex motion dynamics, and noise, UWGS consistently demonstrates the ability to maintain synchronization errors below the microsecond level even under low PDR conditions, down to 20%–30%. In contrast, D-Sync frequently fails to achieve reliable synchronization in such adverse scenarios, particularly at PDR values below 50%, where insufficient packet reception often prevents the numerical method from converging.

The superior robustness of UWGS can be attributed to its reliance on geometric relationships derived from the first five successfully received packets, rather than on Doppler shift estimation and precise propagation delay modeling, which are highly sensitive to node mobility and communication degradation. The primary sources of synchronization error in UWGS stem from motion-induced noise, trajectory deviations from the constant-velocity model, and numerical solver limitations rather than fundamental protocol deficiencies. Despite these imperfections, the achieved timing accuracy consistently translates to ranging errors below 1 meter, which is sufficient for most underwater MAC layer coordination and localization applications.

In conclusion, UWGS offers a practical and efficient alternative to traditional two-way synchronization protocols for underwater mobile networks. Its one-way messaging structure reduces both communication overhead and energy consumption while maintaining accuracy levels suitable for time-slotted medium access control and ranging applications. The protocol's resilience to packet loss and its ability to maintain consistant performance across varying transmission intervals make it particularly well-suited for the challenging and dynamic underwater acoustic communication environment.

As future work, we plan to validate the UWGS protocol in real underwater environments using commercial acoustic modems, enabling a direct comparison between theoretical predictions and experimental results. Furthermore, the mobility model will be extended beyond predominantly vertical motion to encompass general three-dimensional trajectories, allowing for a more comprehensive evaluation of the protocol under realistic underwater operational conditions.

A. JACOBIAN MATRIX DERIVATION

The Jacobian matrix $J(\hat{a},\hat{b})$ of $\mathbf{f}(\hat{a},\hat{b})$ is [22]:

$$J(\hat{a},\hat{b}) = \begin{bmatrix} \frac{\partial f_1}{\partial \hat{a}} & \frac{\partial f_1}{\partial \hat{b}} \\ \frac{\partial f_2}{\partial \hat{a}} & \frac{\partial f_2}{\partial \hat{b}} \end{bmatrix} \tag{34}$$

where each entry $J_{ij} = \frac{\partial f_i}{\partial x_j}$ represents the partial derivative of function f_i with respect to parameter x_j.

A.1 Gradient computation

To compute the Jacobian entries, we require the partial derivatives of the residuals Δt_i with respect to the unknown parameters $\hat{a}$ and $\hat{b}$. Recall from Equation *(21)* that:

$$\Delta t_i = \hat{a}\cdot t_i' + \hat{b} - (h_i - 1)T - p_i/C. \tag{35}$$

Since Δt_i is linear in both $\hat{a}$ and $\hat{b}$, the first order partial derivatives are obtained directly via [23]:

$$\frac{\partial \Delta t_i}{\partial \hat{a}} = t_i', \qquad \frac{\partial \Delta t_i}{\partial \hat{b}} = 1. \tag{36}$$

For the squared residuals $(\Delta t_i)^2$ appearing in f_1 and f_2 (Equation *(22)*), we apply the chain rule:

$$\begin{aligned} \frac{\partial}{\partial \hat{a}}(\Delta t_i)^2 &= 2\cdot\Delta t_i\cdot\frac{\partial \Delta t_i}{\partial \hat{a}} = 2\cdot\Delta t_i\cdot t_i', \\ \frac{\partial}{\partial \hat{b}}(\Delta t_i)^2 &= 2\cdot\Delta t_i\cdot\frac{\partial \Delta t_i}{\partial \hat{b}} = 2\cdot\Delta t_i. \end{aligned} \tag{37}$$

These derivatives follow from the fact that Δt_i is a linear function of both $\hat{a}$ and $\hat{b}$ (Equation *(21)*). Consequently, the squared residuals $(\Delta t_i)^2$ are quadratic functions of these parameters. By applying the chain rule, we obtain the partial derivatives in equations *(36)* and *(37)*.

Using the gradient expressions above, the partial derivatives of f_1 with respect to $\hat{a}$ and $\hat{b}$ are:

$$\begin{aligned} \frac{\partial f_1}{\partial \hat{a}} &= \frac{2\Delta t_1 t_1'}{(t_3-t_1)(t_2-t_1)} + \frac{2\Delta t_3 t_3'}{(t_3-t_1)(t_3-t_2)} \\ &\quad - \frac{2\Delta t_2 t_2'}{(t_3-t_2)(t_2-t_1)} - \frac{2\Delta t_2 t_2'}{(t_4-t_2)(t_3-t_2)} \\ &\quad - \frac{2\Delta t_4 t_4'}{(t_4-t_2)(t_4-t_3)} + \frac{2\Delta t_3 t_3'}{(t_4-t_3)(t_3-t_2)}, \\ \frac{\partial f_1}{\partial \hat{b}} &= \frac{2\Delta t_1}{(t_3-t_1)(t_2-t_1)} + \frac{2\Delta t_3}{(t_3-t_1)(t_3-t_2)} \\ &\quad - \frac{2\Delta t_2}{(t_3-t_2)(t_2-t_1)} - \frac{2\Delta t_2}{(t_4-t_2)(t_3-t_2)} \\ &\quad - \frac{2\Delta t_4}{(t_4-t_2)(t_4-t_3)} + \frac{2\Delta t_3}{(t_4-t_3)(t_3-t_2)}. \end{aligned} \tag{38}$$

Similarly, the partial derivatives of f_2 with respect to $\hat{a}$ and $\hat{b}$ are:

$$\begin{aligned}\frac{\partial f_2}{\partial \hat{a}} = &\frac{2\Delta t_1 t_1'}{(t_3-t_1)(t_2-t_1)} + \frac{2\Delta t_3 t_3'}{(t_3-t_1)(t_3-t_2)} \\ &- \frac{2\Delta t_2 t_2'}{(t_3-t_2)(t_2-t_1)} - \frac{2\Delta t_3 t_3'}{(t_5-t_3)(t_4-t_3)} \\ &- \frac{2\Delta t_5 t_5'}{(t_5-t_3)(t_5-t_4)} + \frac{2\Delta t_4 t_4'}{(t_5-t_4)(t_4-t_3)},\end{aligned} \tag{39}$$

$$\begin{aligned}\frac{\partial f_2}{\partial \hat{b}} = &\frac{2\Delta t_1}{(t_3-t_1)(t_2-t_1)} + \frac{2\Delta t_3}{(t_3-t_1)(t_3-t_2)} \\ &- \frac{2\Delta t_2}{(t_3-t_2)(t_2-t_1)} - \frac{2\Delta t_3}{(t_5-t_3)(t_4-t_3)} \\ &- \frac{2\Delta t_5}{(t_5-t_3)(t_5-t_4)} + \frac{2\Delta t_4}{(t_5-t_4)(t_4-t_3)}.\end{aligned}$$

ACKNOWLEDGEMENT

This work has been partially supported by the Italian National Recovery and Resilience Plan (NRRP) under NextGenerationEU, as part of the partnership on "Telecommunications of the Future" (PE0000001 - program "RESTART"). Additionally, EvoLogics GmbH, Berlin, Germany, is one of the sponsors of this project.

REFERENCES

[1] F. Lu, D. Mirza, and C. Schurgers. "D-Sync: Doppler-Based Time Synchronization for Mobile Underwater Sensor Networks". In: *Proceedings of the Fifth ACM International Workshop on Underwater Networks (WUWNet)*. Woods Hole, MA, USA, 2010, pp. 1–8.

[2] B. Li, S. Zhou, M. Stojanovic, L. Freitag, and P. Willett. "Multicarrier Communication over Underwater Acoustic Channels with Nonuniform Doppler Shifts". In: *IEEE Conference on Oceanic Engineering (OCEANS)*. 2008, pp. 198–209.

[3] N. Chirdchoo, W.-S. Soh, and K. C. Chua. "MU-Sync: A Time Synchronization Protocol for Underwater Mobile Networks". In: *Proceedings of the Third Workshop on Underwater Networks (WUWNet)*. San Francisco, California, USA, 2008, pp. 35–42.

[4] Matin Ghalkhani, Jianyu Zhang, Filippo Campagnaro, and Michele Zorzi. "UWGS: Geometry-Based Enhanced Time Synchronization in Underwater Acoustic Mobile Networks". In: *ASILOMAR Conference on Signals, Systems, and Computers*. IEEE. Pacific Grove, California, USA, Oct. 2024.

[5] Aijun Song, Milica Stojanovic, and Mandar Chitre. "Editorial Underwater Acoustic Communications: Where We Stand and What Is Next?" In: *IEEE Journal of Oceanic Engineering* 44.1 (2019), pp. 1–6. DOI: 10.1109/JOE.2018.2883872.

[6] C.-F. Huang, T. Yang, J.-Y. Liu, and J. Schindall. "Acoustic Mapping of Ocean Currents Using Networked Distributed Sensors". In: *The Journal of the Acoustical Society of America*. 3. 2013, pp. 2090–2105.

[7] P. A. Crossley, H. Guo, and Z. Ma. "Time synchronization for transmission substations using GPS and IEEE 1588". In: *CSEE Journal of Power and Energy Systems Conference*. Sept. 2016, pp. 91–99. DOI: 10.17775/CSEEJPES.2016.00040.

[8] Zhongyue Chen, Huifang Chen, and Wen Xu. "Simplified Time Synchronization for Underwater Acoustic Sensor Networks with High Propagation Latency". In: *OCEANS 2014 - IEEE/MTS - TAIPEI*. 2014. DOI: 10.1109/OCEANS-TAIPEI.2014.6964500.

[9] Sungryul Kim and Younghwan Yoo. "SMP-Sync: Time Synchronization Using Seawater Movement Pattern for Underwater Wireless Networks". In: *International Conference on Computing, Networking and Communications (ICNC)*. Honolulu, HI, USA, 2014, pp. 842–846. DOI: 10.1109/ICCNC.2014.6785447.

[10] Fikret Sivrikaya and Bulent Yener. "Time Synchronization in Sensor Networks: A Survey". In: *IEEE Network* 18.4 (July 2004), pp. 45–50. DOI: 10.1109/MNET.2004.1316761.

[11] B. S. Sharif, J. Neasham, O. R. Hinton, and A. E. Adams. "A Computationally Efficient Doppler Compensation System for Underwater Acoustic Communications". In: *IEEE Journal of Oceanic Engineering* 25.1 (2000), pp. 52–61. DOI: 10.1109/48.820736.

[12] 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*. New York, NY, USA: ACM, 2009.

[13] Sean F. Mason, Christian R. Berger, Shengli Zhou, and Peter Willett. "Detection, Synchronization, and Doppler Scale Estimation with Multicarrier Waveforms in Underwater Acoustic Communication". In: *IEEE Journal on Selected Areas in Communications*. Vol. 26. 9. Nov. 2008, pp. 1638–1649.

[14] J. Liu, Z. Zhou, Z. Peng, J.-H. Cui, M. Zuba, and L. Fiondella. "Mobi-Sync: Efficient Time Synchronization for Mobile Underwater Sensor Networks". In: *IEEE Transactions on Parallel and Distributed Systems* 24.2 (2013), pp. 406–416. DOI: 10.1109/TPDS.2012.71.

[15] F. Zhou, Q. Wang, D. Nie, and G. Qiao. "DE-Sync: A Doppler-Enhanced Time Synchronization for Mobile Underwater Sensor Networks". In: *Sensors* 18.6 (2018), p. 1710. DOI: 10.3390/s18061710.

[16] J. Liu, Z. Wang, J.-H. Cui, S. Zhou, and B. Yang. "A Joint Time Synchronization and Localization Design for Mobile Underwater Sensor Networks". In: *IEEE Transactions on Mobile Computing* 15.3 (2016), pp. 530–543. DOI: 10.1109/TMC.2015.2410281.

[17] Y. Ouyang, Y. Han, Z. Wang, and Y. He. "Underwater Network Time Synchronization Method Based on Probabilistic Graphical Models". In: *Journal of Marine Science and Engineering* 12.11 (Nov. 2024), p. 2079. DOI: 10.3390/jmse12112079.

[18] C. Zhang and H. Wu. "LT-Sync: A Lightweight Time Synchronization Scheme for High-Speed Mobile Underwater Acoustic Sensor Networks". In: *Journal of Marine Science and Engineering* 13.3 (Mar. 2025), p. 528. DOI: 10.3390/jmse13030528.

[19] Y. Qin, Y. Sun, H. Liu, and C. Xue. "Joint Time Synchronization and Localization of Underwater Mobile Node". In: *Wireless Networks* 29.9 (July 2023), pp. 3737–3746. DOI: 10.1007/s11276-023-03441-2.

[20] F. Qu, Z. Wang, L. Yang, and Z. Wu. "A Journey Toward Modeling and Resolving Doppler in Underwater Acoustic Communications". In: *IEEE Communications Magazine* 54.2 (2016), pp. 49–55. DOI: 10.1109/MCOM.2016.7402267.

[21] Jianyu Zhang, Filippo Campagnaro, Antonio Montanari, and Michele Zorzi. "One-Way Ranging for Mobile Underwater Acoustic Networks with Long Interaction Periods". In: *OCEANS 2024 - IEEE/MTS - Singapore*. 2024.

[22] Richard L. Burden, J. Douglas Faires, and Annette M. Burden. *Numerical Analysis*. 10th. Boston, MA: Cengage Learning, 2015.

[23] C. T. Kelley. "Iterative Methods for Linear and Nonlinear Equations". In: *Society for Industrial and Applied Mathematics* (1995). DOI: 10.1137/1.9781611970944.

[24] H. S. M. Coxeter and S. L. Greitzer. *Geometry Revisited*. Washington, D.C.: Mathematical Association of America, 1967, p. 6.

[25] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17.3 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

[26] Jorge J. Moré, Burton S. Garbow, and Kenneth E. Hillstrom. *User Guide for MINPACK-1*. Tech. rep. ANL-80-74. Argonne, IL: Argonne National Laboratory, 1980.

[27] Antonio Montanari, Filippo Campagnaro, and Michele Zorzi. "An adaptive MAC-agnostic ranging scheme for underwater acoustic mobile networks". In: *Computer Networks* 247 (June 2024), p. 110430. DOI: 10.1016/j.comnet.2024.110430.

[28] Tongwei Zhang, Baohua Liu, and Yeyao Liu. "Positioning Systems for Jiaolong Deep-Sea Manned Submersible: Sea Trial and Application". In: *IEEE Access* 6 (2018), pp. 71644–71650. DOI: 10.1109/ACCESS.2018.2881390.

AUTHORS

Matin Ghalkhani received a bachelor's degree in electrical engineering with a specialization in telecommunications from the University of Isfahan, Isfahan, Iran and a master's degree in ICT for Internet and multimedia from the University of Padova, Padova, Italy, where he is currently working toward a Ph.D. degree in information engineering. His research focuses on underwater communication and telecommunication technologies related to ocean studies.

Jianyu Zhang received a master's degree in software engineering from Jilin University, China, in 2019, and a Ph.D. degree in information engineering from the University of Padova, Italy, in 2025. Since 2016, he has been involved in research on underwater acoustic networks at the Smart Ocean Research Center, Jilin University, and later with the SIGNET group at the University of Padova. His research interests include MAC protocols, time synchronization, positioning, spatial-reuse TDMA, and underwater acoustic propagation modeling.

Filippo Campagnaro received the Ph.D. degree in information engineering from the University of Padova, Italy, in 2019. He is currently an Assistant Professor with the Department of Information Engineering, University of Padova, and a cofounder of SubSeaPulse SRL. His research activities focus on underwater acoustic networks, wireless sensor systems, and experimental sea trials.

Michele Zorzi received the Laurea and Ph.D. degrees in electrical engineering from the University of Padova, Italy, where he is currently a Professor in the Department of Information Engineering. His research interests include wireless communications, sensor networks, Internet of Things, vehicular networks, and underwater communications. He has served as Editor-in-Chief and held several leadership roles within the IEEE Communications Society. He is also a Fellow of the IEEE.