Underwater 3D positioning on smart devices

Tuochao Chen, Justin Chan and Shyamnath Gollakota
Paul G. Allen School of Computer Science & Engineering
University of Washington, Seattle, WA, USA
underwaterGPS@cs.washington.edu

ABSTRACT

The emergence of water-proof mobile and wearable devices (e.g., Garmin Descent and Apple Watch Ultra) designed for underwater activities like professional scuba diving, opens up opportunities for underwater networking and localization capabilities on these devices. Here, we present the first underwater acoustic positioning system for smart devices. Unlike conventional systems that use floating buoys as anchors at known locations, we design a system where a dive leader can compute the relative positions of all other divers, without any external infrastructure. Our intuition is that in a well-connected network of devices, if we compute the pairwise distances, we can determine the shape of the network topology. By incorporating orientation information about a single diver who is in the visual range of the leader device, we can then estimate the positions of all the remaining divers, even if they are not within sight. We address various practical problems including detecting erroneous distance estimates, addressing rotational and flipping ambiguities as well as designing a distributed timestamp protocol that scales linearly with the number of devices. Our evaluations show that our distributed system running on underwater deployments of 4-5 commodity smart devices can perform pairwise ranging and localization with median errors of 0.5-0.9 m and 0.9-1.6 m.

Project page with code: https://underwatergps.cs.washington.edu/

CCS CONCEPTS

• **Networks** → **Mobile networks**; • **Applied computing** → *Environmental sciences*; • **Human-centered computing** → **Ubiquitous and mobile devices**.

KEYWORDS

Underwater GPS, ocean sciences, acoustic tracking, smart watches, distributed localization, anchor-free

ACM Reference Format:
Tuochao Chen, Justin Chan and Shyamnath Gollakota. 2023. Underwater 3D positioning on smart devices. In *ACM SIGCOMM 2023 Conference (ACM SIGCOMM '23), September 10–14, 2023, New York, NY, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3603269.3604851

1 INTRODUCTION

In recent years, both industry and academia have shown interest in developing underwater capabilities for mobile and wearable

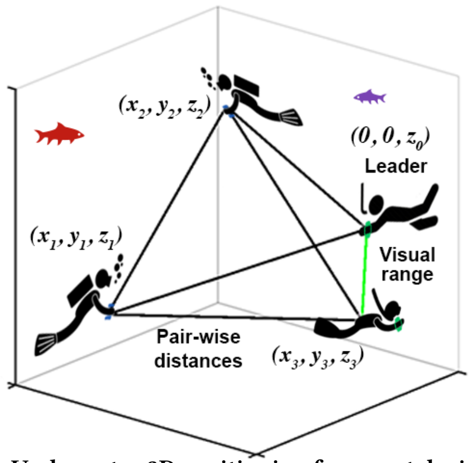

Figure 1: Underwater 3D positioning for smart devices. Our system computes the pair-wise distances between divers using a distributed protocol, and only requires a single diver to be within the leader's visible range. Sound can propagate farther underwater than light.

devices. While the research community has been investigating the potential of smart devices for communicating underwater [32], tech companies [10, 12, 17, 19] are rolling out waterproof versions made for underwater use like the Garmin Descent Watch series [18]. In 2022, among the noteworthy mobile releases was the *Apple Watch Ultra*, designed for professional scuba diving at depths of 40 meters, complete with all the dive computer functions scuba divers need. With a depth gauge sensor, the watch provides a dive profile, water temperature readings, and even safety alerts such as decompression limits, fast ascents, and mandatory safety stops [10, 16, 17].

Our paper takes an important next step in this domain — we introduce the first acoustic system that enables underwater 3D positioning capabilities on smart devices. Just as, in-air positioning systems (e.g., GPS) have been transformative for mobile devices, localization can bring a vital new capability to underwater scenarios. For example, maintaining close proximity with a dive leader is critical to ensure that the divers can help each other in the event of an emergency such as injury or being trapped by ropes or nets [2, 5]. This can be challenging in low visibility situations such as turbid waters or during a silt out, which can cause some divers to become visually separated from their dive leader [1, 3].

Ideally, a dive instructor should be able to locate all their divers, even if they are not all within sight (Fig. 1). Because sound travels much farther underwater, current localization techniques for underwater sensor networks and robots [89, 91, 106] rely on acoustic anchor nodes like floating buoys placed at known locations [30, 92, 94]. These methods involve having multiple anchor nodes that use acoustic signals to either trilaterate the location of an underwater device or use microphone arrays to compute angle-of-arrival (AoA).

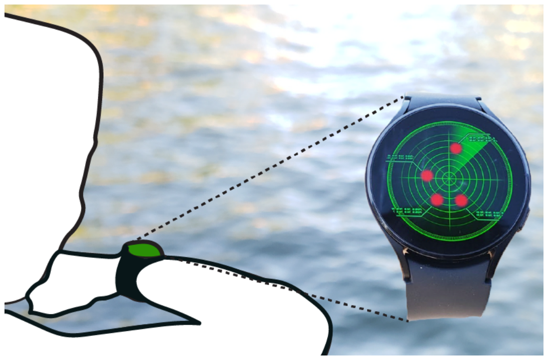

Figure 2: Conceptual interface for our underwater system.

Deploying a dedicated underwater anchor infrastructure is challenging for two key reasons: 1) custom anchors[1] with floating buoys must be manually deployed before any underwater activities, which can be cumbersome to power and maintain in dynamic aquatic environments, and 2) all the divers must have a clear unobstructed path to the anchors for accurate acoustic ranging.

We present a novel system that enables underwater localization capabilities on mobile devices without the need for any infrastructure support (e.g., anchor buoys). In our design, the dive leader device computes the 3D positions of all the other divers in their group (see Fig. 1). Achieving this is challenging since localization techniques developed in our community for radios (e.g., Wi-Fi) [52] do not translate to mobile devices in underwater scenarios. These methods either use multiple routers as anchors at fixed locations [49] or estimate angle of arrival using multiple antennas at each router [100]. While phones and watches have multiple microphones, their separation is an order of magnitude smaller than the required half wavelength.

Instead, we take a distributed approach to this problem. Our system assumes that there is a visible diver in range of the dive leader and that the dive leader first orients themselves towards the visible diver. The devices then run a distributed protocol to find the pair-wise distances between divers. We leverage an important result in graph embeddings [41, 62, 83]: even in a network where the number of links is linear in the number of nodes (versus quadratic in a fully-connected network), the pair-wise distances can uniquely determine the shape of the network topology, as long as the links are well-distributed across the nodes (see §2.1.2). Combining this with the depth sensor data, allows us to calculate the relative diver positions in the 3D space.

A network topology-based approach to underwater positioning is attractive for two reasons: 1) it does not require all the devices to be in range of each other and can work with some missing links in the network (Fig. 3a), and 2) it has the ability to handle some erroneous distance measurements between devices caused by severe multipath (Fig. 3b).

Our design has four key components.

• **Pairwise distance estimation.** Underwater channels are challenging due to multipath and noise from signal reflections between the waterbed and surface, as well as from aquatic creatures and plants, and from scattered particles in the water. The speed of sound also spreads these reflections across time causing a large delay spread [40]. Mobile devices have a limited 3-4 kHz underwater bandwidth and a low sampling rate of 44.1 kHz compared to commodity hydrophones [32]. To improve distance estimation, we use multiple microphones, such as the 2-3 microphones on smartphones or the three-microphone array on the Apple Watch Ultra [10]. Since the time difference of arrival for the direct path between the microphones is upper bounded by the physical distance between them, the sample offset between the direct path in the two microphone channels should be lower than the acoustic propagation time between them. Additionally, each microphone may have a different hardware-based noise profile. Thus, our approach identifies the direct path as the earliest non-negligible peaks *across* microphones whose sample offset satisfies the physical distance constraint between the microphones on the device (see §2.2).

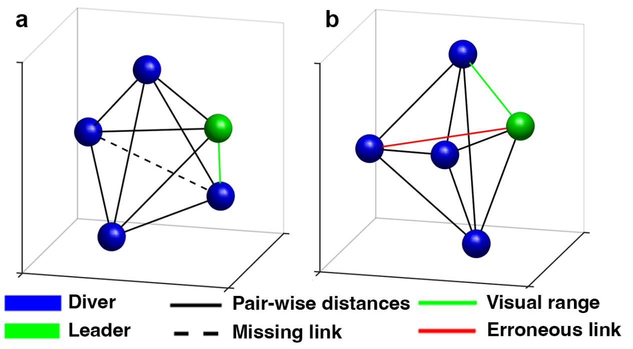

Figure 3: Device localization using a network topology-based approach. Our system can localize (a) even with missing links in the network and (b) some links have severe multipath resulting in wrong distance estimates.

• **Underwater topology estimation with outlier detection.** Given the pairwise distances, we need to estimate the diver positions. This is challenging for two reasons. First, we might not have a fully-connected network and as a result we only have a subset of pairwise distances. Second, some of the pairwise distance estimates might be erroneous and outliers due to severe underwater multipath, which might significantly change the estimated network topology. To address these challenges, we start with multidimensional scaling algorithms [27] to estimate the topology. However even a small number of erroneous pair-wise distance estimates can significantly change the network topology output by these algorithms. To address this, we design an iterative outlier detection algorithm that drops different subsets of links and re-runs the multidimensional scaling algorithm to identify the outlier measurements and compute a uniquely realizable network topology (§2.1.3).

• **Resolving rotational and flipping ambiguities.** While the above method can output the network topology, we still need to address rotational and flipping ambiguities. Specifically, the whole network can rotate along the vertical axis going through the leader device, while still maintaining the depth measurements at each device. As a result, we have an infinite number of possibilities for locations for the devices in the network. We show in §2.1.4 that by imposing the condition that the dive leader first points towards a visible diver, we can resolve rotational ambiguities. This leaves us with flipping ambiguity where we have to pick between two

[1]By anchors, we mean devices whose absolute positions are apriori known.

networks that are mirror images across the line joining the leader and the device that the leader is pointing towards (Fig. 5). While the multiple microphones on smart devices do not have good AoA resolution underwater, we design a novel technique that uses them to formulate a binary classification problem that addresses flipping ambiguity and uniquely determine 3D positions.

• **Distributed timestamp protocol.** The challenge is that the protocol should scale linearly with the number of devices, even when not all devices are in range of each other and the network topology is unknown. At a high level, in our protocol, the leader device broadcasts an acoustic query to initiate the protocol. All other devices broadcast back a response after a fixed delay in their allocated time slots. The time slots are determined by the device ID and set with respect to when the transmission from the leader is initiated. Each device records the timestamp at which it received messages from other devices in its range. This data is then sent back to the leader device, which uses the timestamps to compute the pair-wise distances between devices. Further our protocol can achieve global synchronization and correct for clock drifts even when not all devices are in range of each other.

We implemented our software system and tested it in different network topologies using old Android smartphones placed in water-proof cases, which we use for cost-effective proof-of-concept demonstration. Since Android phones do not have an underwater depth sensor, we estimate depth using the pressure sensors common on smartphones (see §3.1). We evaluated our system in four different underwater environments. Our key findings are as follows:

• The median errors for pairwise 1D distance estimation were 0.48, 0.80 and 0.86 m at 10, 20 and 35 m respectively. The median 2D localization errors were 0.8 m and 0.9 m for a 4- and 5-device network deployment, with a protocol latency of 1.56 s and 1.88 s.

• In the presence of device mobility at 15-56 cm/s, the median localization error for the mobile device was 0.8 m. Further, the flipping disambiguation algorithm could detect the correct positions with 90.1% and 100% accuracy, using signals from 1 and 3 devices with unknown positions.

• With missing links, the system achieved a median localization error of 1.0 m. The median error with occlusions resulting in outlier distance estimates was 1.4 m.

Contributions. Our key contributions are in the distributed system design and the methods to address practical problems like outlier detection, rotational and flipping ambiguities as well as a low-latency distributed timestamp protocol. Together this enables us to build the first underwater acoustic positioning system for smart devices. Our software system does not require any anchor infrastructure. We evaluate our system in various underwater conditions and demonstrate its practical feasibility. We believe that our proof-of-concept system brings 3D localization capabilities on smart devices to the next frontier, i.e., underwater settings.

2 SYSTEM DESIGN

At a high level, our distributed 3D localization system measures the pairwise distances. Since RF signals attenuate quickly underwater [65] and light is susceptible to turbid water and high-ambient light [55], we use acoustics. The distance d between two devices is, $c\Delta t$, where c is the speed of sound and Δt is the time-of-flight. The underwater sound speed can be approximated using Wilson's equation [98]: $c = 1449+4.6T-0.055T^2+0.0003T^3+1.39(S-35)+0.017D$. Here, T, S, D are the temperature, salinity, and depth. The depth limit for recreational divers is 40 m [7, 8]. Prior work [54] shows that at these depths, the maximum change in the speed of sound is $30m/s$, which is only a 2% relative error at $1500m/s$. One can improve accuracy by using depth sensors and configuring the temperature and salinity for different water bodies.

If we know the timestamps at which the sender and receiver sent and received the acoustic signals, we can compute Δt. Our system has three key components: a topology-based localization technique, a method to estimate pairwise distances for challenging underwater multipath channels and a distributed timestamp protocol that operates in unknown network topologies.

2.1 Topology-based localization

At a high level, given a large number of nodes in the 3D space and the exact pairwise distances between all the nodes, one can uniquely determine the shape and network topology [62, 83]. There are four key challenges in using this underwater.

• *Pairwise distance estimates.* Instead of outputting the extra pairwise distances, our system has median errors of 0.5-0.9 m.

• *Occlusions.* Links blocked by rocks, marine life or humans, or with severe multipath lead to large distance estimate errors.

• *Missing links.* Some divers may be out of range of each other.

• *Ambiguity.* In addition to network shape, for 3D positions, we need to address rotation and flipping ambiguities.

Say, there are N devices, including the leader and we have the matrix $D \in \mathcal{R}^{N\times N}$, where $D_{i,j}$ represents the pairwise distances between device i and j. The depth of each device is also known from onboard sensors as h_i. Our objective is to estimate 3D positions, $P_i = [x_i, y_i, z_i], \forall i \in [0, N-1]$, to minimize a stress function $S(\cdot)$.

$$\min_{P_i} \quad S(P_0, \ldots, P_{N-1}) = \sum_{0\le i<j<N} w_{i,j}(D_{i,j} - \|P_i - P_j\|)^2$$

Where $z_i = h_i$ and P_0 is the leader position. $w_{i,j}$ is the element of a symmetric and non-negative weight matrix. When the link between i and j exists, $w_{i,j}$ is 1 but is 0 for missing links. We break the problem down into three separate steps: (1) project onto the 2D space, (2) estimate the topology in that 2D space, and (3) resolve any ambiguity regarding rotation and flipping.

2.1.1 Projection to 2D space. Given that we get the depth measurements from the on-device sensors, the 3D localization problem can be simplified to a 2D localization problem using projection [90]. We can compute the pairwise distances between devices i and j in the projected 2D space as, $D^{2D}_{i,j} = \sqrt{D^2_{i,j} - (h_i - h_j)^2}$. After projection, the problem now is to optimize the 2D locations, $P^{2D}_i = [x_i, y_i]$:

$$\min_{P^{2D}_i} \quad S(P^{2D}_0, \ldots, P^{2D}_{N-1}) = \sum_{0\le i<j<N} w_{i,j}(D^{2D}_{i,j} - \|P^{2D}_i - P^{2D}_j\|)^2$$

2.1.2 Handling link loss. Link losses can be caused by being out of range, packet drops, and blocking. When a link loss happens, the pairwise distance measurement for this link is missing. In this

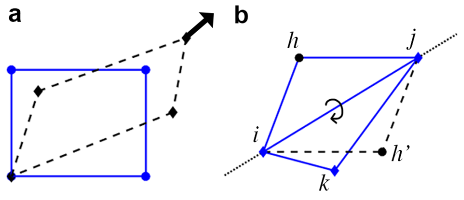

Figure 4: (a) Non-rigid graph. (b) Partial reflection.

setting, we set the weights $w_{i,j}$ to 0 for missing links. We then solve the above problem as a multi-dimensional scaling optimization [27]. Specifically, we use the weighted SMACOF (Scaling by MAjorizing a COmplicated Function) algorithm [36, 51]. Different from the steepest descent approach which directly applies iterative minimization on the stress function, SMACOF finds a convex function $\mathcal{T}(\cdot)$ that majorizes the stress function $S(\cdot)$, i.e., $\mathcal{T}(\cdot) \geq S(\cdot)$ [36] and iteratively minimizes $\mathcal{T}(\cdot)$ at each step. Compared to other approaches [35], SMACOF performs better in terms of accuracy and rate of convergence.

Requirements. The topology of a fully connected network with N nodes can be uniquely determined since the $\binom{n}{2}$ links constrain its shape. However, we do not need a quadratic number of links to determine the shape. At a high level, N nodes in a 2D space have $2N$ degrees of freedom. Since the 3 degrees of freedom from rotation and translation cannot be determined by pairwise distances, we have $2N - 3$ degrees of freedom [41], which is linear in the number of nodes. In fact, a graph is defined as rigid if it does not admit a continuous deformation other than global rotation, translation, and reflection. Fig. 4(a) shows an example graph that is not rigid and can be continuous deformed. Laman's theorem [56] provides the necessary and sufficient conditions: *A graph with n nodes and $2n-3$ links is rigid in two dimensions if and only if no subgraph with n' nodes has more than $2n' - 3$ links.*

However, rigid graphs are still susceptible to discontinuous non-uniqueness, such as partial reflections. Fig. 4(b) shows an example where one node has two possible configurations corresponding to a reflection across a set of mirror nodes. For $N = 3$, three links that satisfy the triangle inequality, uniquely determine a triangle. For $N > 3$ nodes, localizability is used in graph theory to describe if the graph can be uniquely realized given the distance information. [41] provides the necessary and sufficient condition: *A graph is uniquely realizable in two dimensions iff it is redundantly rigid and deleting any two nodes results in a connected graph.*

A graph is redundantly rigid if it can remain rigid upon removal of any single link. If the graph satisfies the above condition, then it can be uniquely determined. In practice, however, the pairwise distances are estimated and not exact. As a result, we have errors in the 2D localization results. In §2.1.5, we simulate different network topologies and present analytical results.

2.1.3 Handling outlier measurements. Outlier measurements can occur when the direct path is blocked or severely attenuated, and multi-path is mistaken for the direct path. Even a small number of outliers can significantly change the topology since our optimization function gives equal weight to every pairwise distance. While [25] uses the triangle inequality to detect outliers, in our deployments, the outlier errors are often not large enough to break the triangle inequality. [39] uses regularization to jointly optimize the topology and outlier detection. However, this is known to be very sensitive to the selection of regularization parameters [25].

Algorithm 1: Outlier Detection

```
**Input:** Set of nodes $0, 1, \ldots N-1$
Pairwise 1d ranging matrix $D = [d_{i,j}]_{M \times M}$
Maximum dropping outlier number $O_{max}$
**Output:** 2D location of the nodes $P = [p_i]_{N \times 1}$
$W_0 \leftarrow Ones(M, M)$; // Init the weight matrix with all ones
$E_0, P_0 \leftarrow SMACOF(D, W_0)$; // $E_0$ is normalized stress
  function, $P_0$ is output of 2D positions
if $E_0 < 1.5m$ then
    $P \leftarrow P_0$;
    return $P$
for $n_{drop} \leftarrow 1$ to $O_{max}$ do
    $E_{min} \leftarrow E_0$
    $P_{min} \leftarrow P_0$
    for $s_{drop} \leftarrow Subsets(M, n_{drop})$ do
        // Subsets() outputs all possible drop subsets of $M$
          links with the size $n_{drop}$
        $W \leftarrow W_0$;
        $W(s_{drop}) \leftarrow 0$;
        $E', P' \leftarrow SMACOF(D, W_0)$;
        if $E_0 - E' > 0.9 * E_0$ and $E' < E_{min}$ then
            $E_{min} \leftarrow E'$;
            $P_{min} \leftarrow P'$;
    if $E_{min} < 1.5m$ then
        $P \leftarrow P_{min}$;
        return $P$
    $E_0 \leftarrow E_{min}$
    $P_0 \leftarrow P_{min}$
return $P_0$
```

Our intuition instead is that in scenarios without outliers, the pairwise distance errors as well as the output of the stress function in the SMACOF algorithm are both within certain ranges. So we normalize the output of the stress function $S(\cdot)$ with the number of links in the network. If this normalized value surpasses a threshold (1.5 in our implementation), we assume that there are outliers. To detect the outliers, we iteratively drop different subsets of links by setting their corresponding weights $w_{i,j}$ to 0 and re-run the SMACOF algorithm. If the new normalized stress function have significantly decreasing, the dropped links may be the outliers. We repeat this process with more outlier links until the stress function does not see a significant reduction (we set this threshold to 90%). Since dropping too many links may lead to the unstable output [39], we set a maximum number of outliers to 3. Further, we cannot drop all subsets of links since dropping some subsets may make the graph not uniquely realizable. Thus we only run the SMACOF algorithm when the resulting graph is still uniquely realizable.

2.1.4 Rotation and Flipping ambiguity. The estimated network topology from the above optimization still has rotation and flipping ambiguities. A rotation ambiguity occurs since as shown in Fig. 5a the topology can rotate while still satisfying the height requirements. This rotational ambiguity can be addressed by rotating the edge between leader and the pointed device along the arrow in the figure, since we require the leader is oriented themselves towards

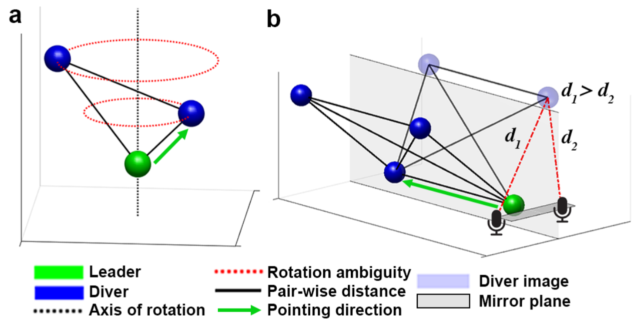

Figure 5: (a) Rotational and (b) flipping ambiguity.

the device. This leaves us with flipping ambiguity where we have to pick between two networks that are mirror images across the line joining the leader and the device that the leader is pointing towards. To resolve this, we leverage the dual microphones in the leader's device. At a high level, while the physical separation of the two microphones on mobile devices is not large enough to provide a good AOA resolution in underwater scenarios, we can still them to perform a simple binary classification task by determining at which microphone the diver's signal arrives first. For nodes that are to the left (right) side of the line in Fig. 5b, the left (right) microphone receives the signal before the right (left). We can use this information to resolve flipping ambiguity. Specifically, when the signal from diver i arrives, we use the dual-microphone channel estimation algorithm in §2.2 to estimate and compare the direct paths (we denote the direct path indexes in dual microphones are m_i and n_i). However, in the real-world, this might be incorrect due to multipath. To boost flipping disambiguation accuracy, we use a voting mechanism to jointly estimate the flipping from all the diver signals (user $2, 3, \dots N-1$) (excluding the leader 0 and the pointed user 1). Since flipping gives two options: $\{P_i\} = \{[x_i, y_i, z_i]\}$ or $\{P'_i\} = \{[x'_i, y'_i, z'_i]\}$, we compute the function,

$$V(\{P_i\}) = \sum_{i \in [2,N-1]} sgn(m_i - n_i) sgn\Big((x_i - x_0)(y_1 - y_0) - (y_i - y_0)(x_1 - x_0)\Big).$$

If $V(\{P_i\}) > V(\{P'_i\})$ we output $\{P_i\}$, otherwise we output $\{P'_i\}$. Note that the leader only needs to point the device to the nearby diver and does not need to rotate to a different position or angle.

2.1.5 Analytical evaluation. To understand our topology-based algorithm, we analyze it in simulation. We randomly generate N devices in a $60 \times 60 \times 10m$ 3D space. We place the leader at the center of the 3D space and randomly generate its height. User 1 is generated with the distance between leader and user 1 randomly selected between 4 and 9 m. We then randomly generate the positions for the remaining divers in the 3D space. We add uniformly-distributed errors to the ground-truth as our measurements: $[-\epsilon_{1d}, \epsilon_{1d}]$ for pairwise distances, $[-\epsilon_h, \epsilon_h]$ for height, and $[-\epsilon_\theta, \epsilon_\theta]$ for pointing angle. For each test, we randomly generate 200 samples and report the mean 2D localization error across all divers, excluding the leader. Fig. 6a plots 2D localization error as a function of the error in pairwise distances. We set $N = 6$, $\epsilon_h = 0.4m$, and $\epsilon_\theta = 0$. As expected, errors in pairwise distances translate to larger 2D localization errors. Fig. 6b shows that as the number of devices N increases, the 2D localization error decreases. Here, we set $\epsilon_h = 0.4m$, $\epsilon_\theta = 0$, and $\epsilon_{1d} = 0.8m$. Fig. 6c sees a trend with the pointing error with the other parameters set to: $N = 6$, $\epsilon_h = 0.4m$, and $\epsilon_{1d} = 0.8m$. Fig. 6d shows the results with different number of link drops where $N = 6$, $\epsilon_h = 0.4m$, $\epsilon_{1d} = 0.8m$, and $\epsilon_\theta = 0$.

2.2 Pairwise distance estimation

To compute these distances, we estimate the exact timestamp when the acoustic signal arrives. This is challenging since the direct path can be severely attenuated underwater and thus, we can not rely on the assumption that the highest peak or the first non-negligible peak in the multipath profile is the direct path. Fig. 7 shows that there can be some peaks before the direct path with amplitude greater than the average noise level ("Wrong peak" in Fig. 7).

To reduce the probability of picking these wrong peaks, we use the two microphones on the mobile devices. The basic idea of our joint synchronization algorithm is that the time difference of arrival at the microphones (e.g., bottom and top microphones on phones) is physically constrained by the distance between them. Thus, the sample offset between the direct path at the top microphone channel and the direct path at the bottom microphone should be lower than the acoustic propagation time between the two microphones (two black cross symbols in Fig. 7). Furthermore, the multipath created by the water-proof case is different at the two microphones. Finally, the two microphones may have a different noise profile.

Thus, our algorithm identifies the direct path as the earliest non-negligible peaks in both channels whose sample offset satisfies the physical distance constrain between the two microphones. Specifically, we denote the estimated channel for the first microphone as $h_1(n)$ and the second microphone as $h_2(m)$, where n and m are the channel tap numbers. Then, we normalized h_1 and h_2 to be between 0 and 1. We then check whether the sample n in channel h_1 is a peak. Next, we estimate the channel noise level for the two microphone channels by calculating the average power in the last 100 channel taps, respectively denoted as w_1 and w_2.

$$\begin{aligned} \min_{\tau_{LOS}} \quad & \tau_{LOS} = (n+m)/2, \quad \forall m, n \in [0, L/fs) \\ s.t. \quad & h_1(n) > w_1 + \lambda, h_2(m) > w_2 + \lambda, \\ & IsPeak(n, h_1) \cap IsPeak(m, h_2), |n - m| \leq d/c \end{aligned}$$

Here τ_{LOS} is the delay of the direct path, n and m are the tap numbers in the channels h_1 and h_2. w_1 and w_2 are the estimated noise levels in these two channels. λ is a conservative parameter (we set it empirically to 0.2). d is the physical distance between the two microphone ($d = 16cm$ in our implementation), c is the speed of sound and L is the entire length of the channel (1920 samples).

2.2.1 Signal processing pipeline. We use OFDM symbols between 1-5 kHz as the preamble. We use this frequency band given the underwater response of mobile devices [32]. We fill the OFDM bins with a ZC sequence [97] which is phase-modulated and orthogonal to its delayed version [104]. ZC-modulated OFDM symbols can achieve much better performance than their well-known counterpart, chirps [28, 81]. We then concatenate 4 such identical OFDM symbols and multiply each with a PN sequence with different signs ([1, 1, -1, 1]), to increase robustness to noise [72]. Between each OFDM symbol, we insert a cyclic prefix to avoid inter-symbol interference. The length of each OFDM symbol is 1920 samples and

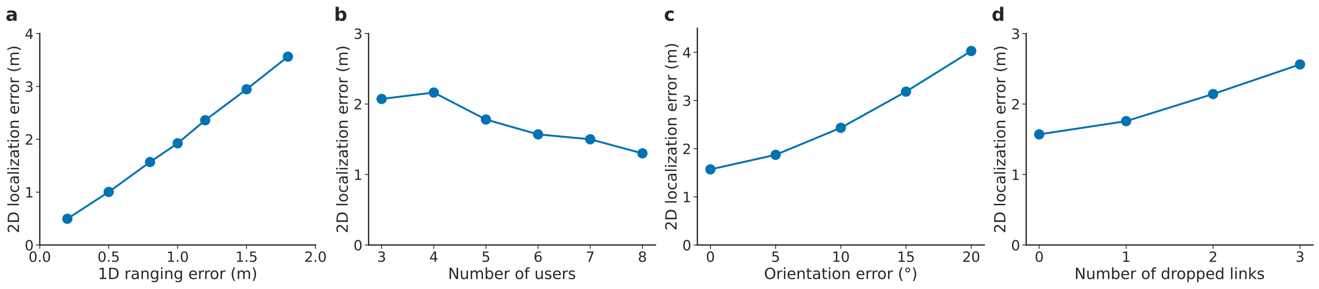

Figure 6: Analytical evaluation of mean error versus different parameters in a $60 \times 60 \times 10m$ 3D space.

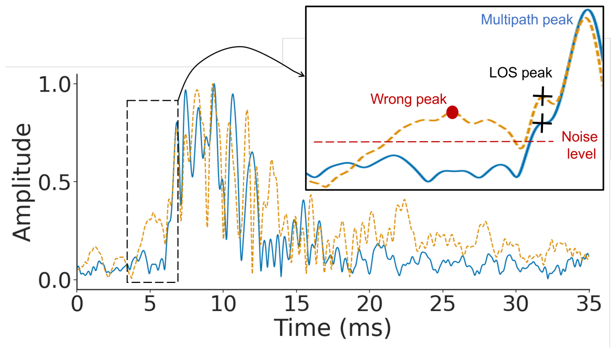

Figure 7: The yellow and blue curves correspond to the channel estimates at two microphones.

the length of the cyclic prefix is 540 samples. Our preamble synchronization algorithm at the receiver is composed of three steps.

First, we perform cross-correlation between the microphone stream and the sending preamble. In the presence of a preamble, this results in a correlation peak. However, the height of this peak could vary a lot as the SNR decreases at long distances. Meanwhile, some underwater spiky noise like bubbles would also cause high peaks in the cross-correlation, leading to plenty of false positives.

To address this we use auto-correlation. Since our preamble has 4 OFDM symbols that are encoded with a 4-bit sequence, we split the received signal into 4 segments corresponding to the 4 OFDM symbols, multiply each segment by the 4-bit sequence, and apply correlation among them [72]. Auto-correlation is helpful because the spiky noise rarely has such a complex encoded pattern (PN sequence) and since the 4 received OFDM symbols suffer from nearly the same multi-path, the correlation value between two received OFDM symbols would be much higher than the correlation value between the received and transmitted symbols. We set a threshold of 0.35 in our design for valid preamble detection.

Due to the severe underwater multi path profiles, the side-lobe height in the correlation curve is usually higher than the direct path. Hence, coarse synchronization error based on only correlation is usually hundreds of samples, corresponding to over 6 m error. To achieve more fine-grained synchronization, we apply channel estimation, where we leverage the channel profiles to identify the direct path. While MUSIC-like estimators [28] could achieve super-resolution channel profiles, the signal space decomposition is difficult due to the extremely dense underwater channel and it has a high computational complexity. Therefore, we use an LS channel estimator [50]. Specifically, based on the coarse synchronization of cross-correlation and auto-correlation, we segment out 4 received OFDM symbols y_1, y_2, y_3, y_4 from the microphone stream. Then we apply FFTs on these 4 symbols to get Y_1, Y_2, Y_3, Y_4. We denote the FFT of the original OFDM symbol before multiplication with PN sequence by X and denote the PN sequence by PN_1, PN_2, PN_3, PN_4. The channel model can be written as $Y_i(k) = H(k)(PN_i \cdot X(k)) + N_i(k)$, where k represents the k^{th} frequency bin. The estimated channel is $\hat{H}(k) = \frac{1}{4}\sum_{i=1}^{4} \frac{1}{PN_i} \cdot X(k)^{-1} Y_i(k)$.

2.2.2 *Low-level audio timing.*

Our distributed timestamp protocol in §2.3 requires each of the responding devices to transmit at a fixed time after it receives the message from the leader or other devices. A key challenge in achieving this is that we need to address the buffer delays. Specifically, at each device, the microphone and speaker buffers are not synced with each other [6]. These buffers are filled in independently by the OS. Thus, we do not know the timestamps corresponding to the samples in the two buffers. At a high level, we use the low-level audio timing in the OS to achieve self-synchronization between the microphone and speaker buffers on each device. During initialization, when the microphone and speaker data streams are open, an initial calibration signal is sent from the speaker to its own microphone (green signal in Fig. 8). Then the offset between the speaker and microphone buffers, Δn, is estimated by subtracting the microphone buffer index when the calibration signal is detected and the speaker buffer index when the calibration signal is written (as shown in Fig. 8). Once we open the microphone and speaker data streams, we do not close them so as to keep this offset constant. When the microphone detects the leader at index m, the device write the response message at index n in speaker buffer where $n = m - \Delta n + fs \cdot t_{reply}$ (fs is the sampling rate). Thus, the device can reply at a desired interval t_{reply} after receiving the message at the microphone (see Appendix).

2.3 Distributed timestamp protocol

An approach is for each pair of devices to independently measure their pairwise distances. This scales quadratically with the number of devices. Our protocol should satisfy four key requirements.

- *Efficiency.* Since divers move, it is critical for the protocol to compute pairwise distances across the network in 1-2 s.
- *Collisions.* While there is no global clock underwater, the protocol should avoid packet collisions between devices.
- *Unknown topology.* Since our goal is to find the topology shape, we can not assume a known topology. Further, the protocol has to work in networks that are not fully-connected.

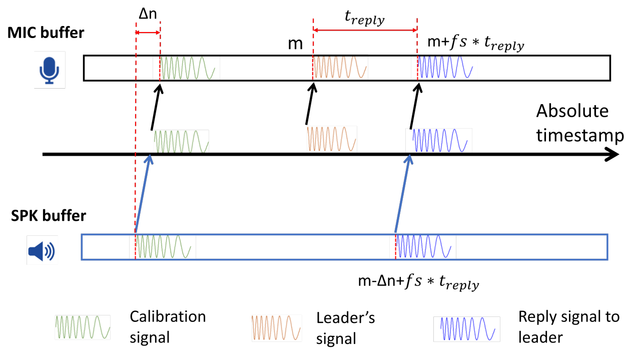

Figure 8: Synchronization of microphone & speaker buffers.

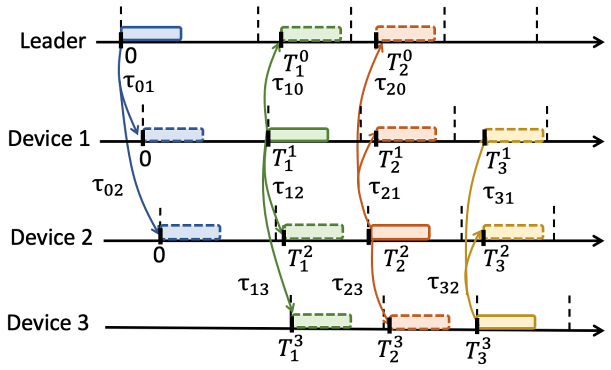

Figure 9: Distributed timestamp protocol.

• *Not all devices are connected to leader.* The protocol should work even when not all devices can hear the leader, making time synchronization with the leader challenging.

In our design, the dive leader initiates the protocol by transmitting a short message with a preamble and its ID and the other devices respond to it using time-division multiplexing (TDM). Each device is pre-assigned a unique ID, where the leader device is assigned the ID 0, and other user IDs are from 1 to N-1. After all devices respond, the timestamp from each user can be used to compute the pairwise propagation time and distances. We define the propagation time between device i and j as τ_{ij}. The distance between them can be computed as $D_{i,j} = c\tau_{ij}$. Since no global synchronization clock exists underwater, device i records its local time T^i. We define T_j^i as the time when the message from device j arrived at the microphone buffer of device i.

All devices are in leader's range. All devices use the leader's message to synchronize and respond in a TDM fashion. When device i receives the leader's message, it sets its local time to 0 i.e., $T_0^i = 0$. Then each device divides its local time into slots and responds in the slot ordered by their ID. Specifically, device i will send its message containing a preamble and its ID at $T_i^i = \Delta_0 + (i-1)\Delta_1$ based on its local clock. Δ_0 is the maximum time it takes to process the message from the leader in real-time as well as the smartphone audio input and output latency. $\Delta_1 = T_{packet} + T_{guard}$, where T_{packet} is the duration of the message and T_{guard} is the guard interval which accounts for the maximum propagation delay to avoid packet collisions. When all devices are in leader's range, t_{guard} should be larger than twice the maximum possible propagation time within the diver group τ_{max} (i.e. $T_{guard} > 2\tau_{max}$) to guarantee no packet collisions.

The protocol stops after N slots which is the number of devices in the network. At the end of this protocol, each device records the timestamps at which they received messages from all devices in their range and transmit this timestamp information to the leader, as described in §2.4. Given these timestamps, the leader can compute the pairwise distances between all devices. Specifically, the distance between device i and j ($i < j$) can be computed as follow: $D_{ij} = \frac{c}{2}[(T_j^i - T_i^i) - (T_j^j - T_i^j)]$. Here, we ignore the propagation time from the device's speaker to its own microphone, because it is small compared to underwater distances.

Not all devices are in leader's range. As before, the subset of devices that receive the leader's message respond in their assigned TDM slots. Say device i did not receive the leader's message but received a message from other devices. In this case, device i uses the first message it received to synchronize its local time and compute its assigned transmission time slot (Fig. 9). Here, we have two cases. First, say device i received a message from device j (where $(i-j)\Delta_1 > \Delta_0$) at time T_j^i. It estimates its transmission timeslot as $T_i^i = T_j^i + (i-j)\Delta_1$. In its slot, device i transmits its ID and the ID for device j to ensure that other devices know that its transmit time was set with respect to device j. Next, say the first message received by device i is from a device with an ID j (where $(i-j)\Delta_1 < \Delta_0$). In this case, it will miss its slot and will have to wait for all remaining devices to transmit before it has a chance to respond in the $T_i^i = T_j^i + (N-j+i)\Delta_1$ time slot. We note three points.

• *Packet losses.* If there is some device k such that both device i and j received its message, then we can compute the distance between i and j even if one of the messages from either i to j or j to i is lost.

• *ID encoding.* We use MFSK to encode the ID. We divide the 1-5 kHz frequency band into N bins (N is the dive group size). For user i, we set all bins of i^{th} bin to 1 and all other bins to 0. We use a maximum-likelihood estimator to decode the user ID.

• *Latency analysis.* We set $\Delta_0 = 600\ ms$, $T_{packet} = 278\ ms$, $T_{guard} = 42\ ms$, and $\Delta_1 = 320ms$. When all divers are in the leader's range, the maximum round trip time for a protocol run is $T_{round} = \Delta_0 + (N-1)\Delta_1$. When some divers are out of range of leader, in the worst case, the maximum round trip time is $T_{round} = \Delta_0 + 2(N-1)\Delta_1$.

2.4 Communication system

After the distributed protocol, the users need to send the timestamp and depth information to the leader. Due to the limited bandwidth, we need to compress this data. We discretize depth at a 0.2 m resolution and so we need 8 bits to represent depths between 0-40 m. For timestamps, instead of transmitting the absolute timestamp T_j^i, we transmit the time difference between T_j^i and the assigned time slot for device j, i.e. $\Delta_0 + (j-1)\Delta_1$. This time difference is bounded by $[0, 2\tau_{max})$. We set $2\tau_{max} = 42ms$ which corresponds to a maximum propagation distance of 32 m. For fs=44100Hz, this is around 1852 samples in the microphone buffer. At a 2 sample resolution, the

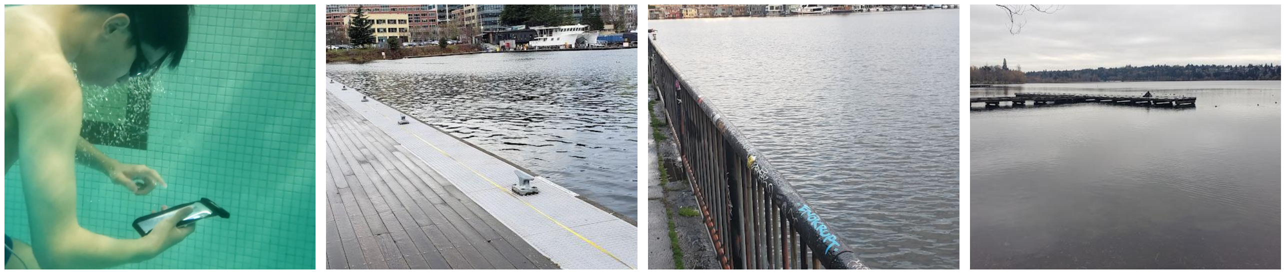

Figure 10: Different underwater scenarios. (a) Swimming pool, (b) Dock, (c) Viewpoint, (d) Boathouse.

timestamp differences require $log_2(1852/2) \approx 10bits$. Hence, with N divers, each device sends $10(N-1)+8$ bits to the leader.

We use FSK which is a widely deployed modulation [32, 57, 86, 93]. The devices transmit simultaneously to the leader device to reduce latency. To do this, we divide the 1-5 kHz bandwidth into N bands and pre-assign each device to a different band. Device i uses FSK within its band. We apply 2/3 convolutional coding to the payload. We note that this communication system takes around 0.9, 1 and 1.2 s when N is 6, 7 and 8 at a bit rate of 100 bps per device.

When some users are out of range of the leader, they cannot directly send the message back. Thus, a multi-hop communication mechanism is required which is not in the scope of this paper.

3 RESULTS

We evaluated our system in the four environments in Fig 10.

- *Swimming pool.* The length of the water here is around 23 m. The depth of the swimming pool varied from 1 to 2.5 m.
- *Dock.* This outdoor location has a length of around 50 m with a depth of 9 m. Boats and seaplanes would frequently sail or dock at this location with aquatic plants and animals.
- *Viewpoint.* By the waterfront of a park with a length of 40 m. The water had a depth of around 1 to 1.5 m.
- *Boathouse.* Fishing dock by the lake with a horizontal distance of 30 m. The lake had a depth of 5 m. This is a busy location with people fishing and kayaking close to the dock.

3.1 Benchmark evaluation

Accuracy versus device separation. Here, we evaluate our system along the dock of a lake with an average water depth of 9 m. We performed experiments using two Samsung Galaxy S9 phones set to transmit at the maximum speaker volume. The phones were set to transmit using the speaker at the bottom of the device, and receive using the microphones at the bottom and top of the device. The experiment was repeated in each location every six seconds. At each distance, the sender and replier are set to exchange messages up to a maximum of 60 times. The measurements were divided into roughly three sessions, where after 20 measurements, the phones were removed and submerged again. The phones were enclosed in a pouch (Hiearcool waterproof phone pouch [9]) and attached to a selfie stick and telescopic extension pole, which was used to submerge the phones to a depth of 2.5 m. The selfie stick and extension pole were attached using waterproof tape and zip ties. This setup was chosen so that the phone's position and angle could be controlled. We used a tape measure to mark distances up to 45 m.

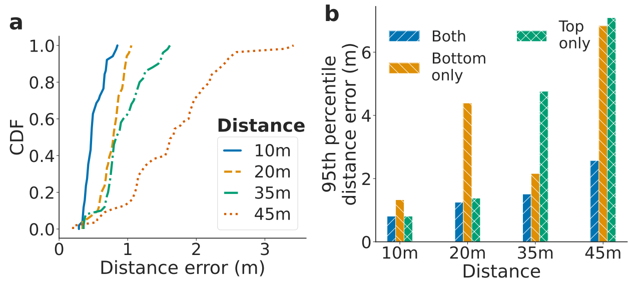

Figure 11: (a) Ranging accuracy v/s separation. CDF of absolute error as a function of separation. (b) 95% errors using both microphones, the bottom and top microphone only.

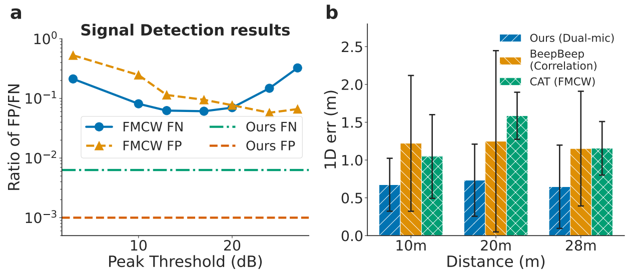

Figure 12: (a) Ratio of false positive and false negative for signal detection (b) 1D ranging error versus distance.

Fig. 11a shows the CDF of the absolute error obtained by our system for four distances of 10, 20, 35 and 45 m. The error in distance increases with separation between the devices because the signal strength is lower at larger separation. We also analyze the effect of using both the top and bottom microphones for ranging versus using only a single microphone in isolation. Fig. 11b shows the 95th percentile distance error for these scenarios at distances of up to 45 m. The figure reveals the following: firstly, utilizing both microphones yields lower ranging errors at all distances. This can reduce error by as much as 4.52 m at a distance of 45 m (while we set the maximum distance to 32m in the distributed protocol, here we relax this to evaluate the limits of 1D localization). Secondly, when a single microphone is used in isolation, there is no clear relationship between microphone position and error.

We also compare our 1D ranging algorithm with previous works on acoustic-based 1D ranging and tracking like [64, 75]. [75] uses the linear chirp signal and applies auto-correlation with specially-designed peak detection. [64] implements an FMCW-based approach where the receiver mixes the received signal with the transmitted signal. For a fair comparison, we control the duration and bandwidth of three types of signals to be the same. We put the

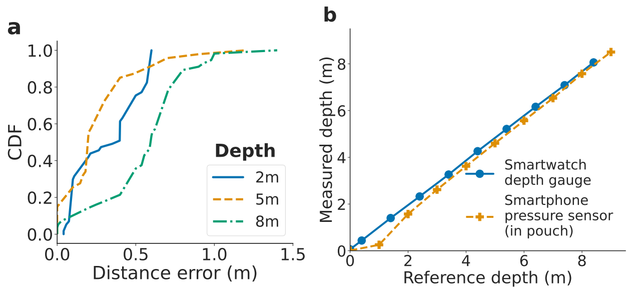

Figure 13: (a) Effect of device depth. Errors for different depths with devices separated horizontally by 18 m. (b) Accuracy of depth measurements from smartwatch and phone.

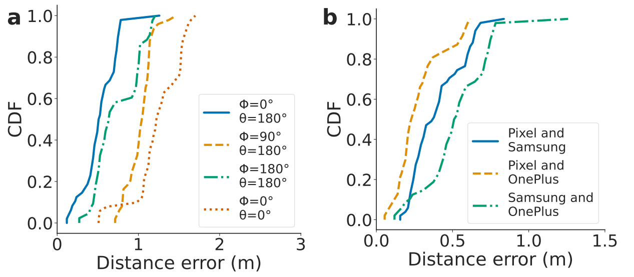

Figure 14: Effect of (a) orientations with phones separated by 20 m, and (b) various smartphone model pairs.

phones at horizontal distances of 10, 20, 28m with a depth $\sim 1m$ at the boathouse location. To compare the robustness of signal detection, we send three types of preambles for 180 times at each distance. We implement the window-based power threshold TH_{SD} in [75] to detect the FMCW signal. Since $TH_{SD} = 3dB$ in [75] is selected for in-air experiment, we try different TH_{SD} to calculate the false positive and false negative for fair comparison. Fig. 12 (a) shows that our preamble detection is more robust than the state of art. To compare the 1D ranging algorithm with [64, 75], the phones exchange each of different signals $\sim$ 60 times. Then we apply our dual-mic channel estimation, auto-correlation [75] and FMCW[64] on these measurements to calculate the 1D ranging error. Fig. 12 (b) shows the mean value and standard deviation of the 1D ranging errors demonstrating that our approach outperforms prior work.

Accuracy versus depth. We place the smartphones at different water depths at the dock location which had a total depth of 9 m. We lowered the smartphones into the water using ropes marked at 2, 5, and 8 m. The phones were weighed down with a bag of pebbles to ensure the ropes were vertical. The phones were positioned at a horizontal distance of 18 m. Unlike previous experiments, the rope would cause the phone to rotate and sway slowly. We repeat measurements thrice at each depth. Fig. 13a shows that the median and 95th percentile error is lowest at 0.28 and 0.73 m for the 5 m depth, which is around the midpoint depth of the dock. This likely is because multipath reflections can be stronger when the devices are close to the surface or floor of the water body.

Effect of phone orientation and make. We evaluate this at the dock at a horizontal distance of 20 m and a depth of 2.5 m. We first positioned the speaker and microphone of both phones to directly face each other so their azimuth ϕ and polar angle θ is set to 0° and

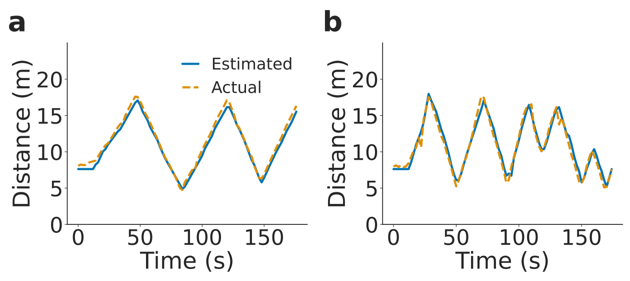

Figure 15: 1D Ranging errors for moving devices.

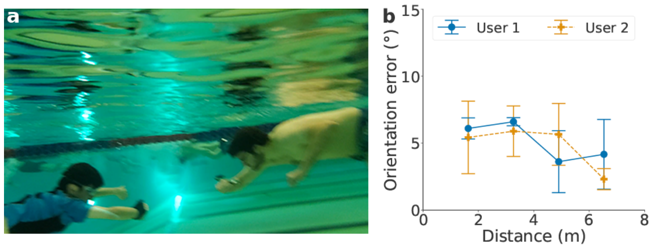

Figure 16: Human orientation evaluation.

180° respectively. We measure the error when the sender phone is rotated to different azimuth and polar angles. We first rotate the sender phone in the azimuth angle to 90° and 180° while keeping the polar angle constant. We then reposition the phone so its speaker and microphone faces upwards with $\phi = 0°$, $\theta = 0°$. Fig. 14a shows that median error ranges from 0.54 to 1.25 m. Note that when the phone faces upwards it had the largest error likely because the phones are closer to the water surface resulting is higher multipath when pointing towards the surface. We also evaluated our system with different smartphone model pairs. In Fig. 14b, we evaluated three different Android phones models.

Depth accuracy. We evaluate the depth gauge sensor on the Apple Watch Ultra and use the pressure sensor on the Samsung Galaxy S9 smartphone in a waterproof case for estimating underwater depth. The depth gauge readings from the smartwatch were obtained from the Oceanic+ app. The smartphone's pressure sensor values P in units of Pascals are converted to depth measurements h in units of meters using the equation [73]: $h = \frac{P - P_0}{\rho g}$, where $\rho = 997\ kg/m^3$ is the average density of water, $g = 9.81\ m/s^2$ and $P_0 = 101325\ Pa$ is atmospheric pressure at sea level. We performed this evaluation in the dock location which had a depth of 9 m, and lowered each smart device underwater with a rope in increments of 1 m using markings on the rope as a measure of ground truth. The devices were held in place at each depth for 30 s. Fig. 13b shows the accuracy for depth from the smartwatch and smartphone. Across all measurements, the average depth error on the smartwatch and smartphone were $0.15 \pm 0.11\ m$ and $0.42 \pm 0.18\ m$, respectively.

Effect of motion. We first evaluate how our 1D ranging algorithm works when the device keeps continuously moving. In this experiment, we keep a phone static and attached another phone to the extension pole in the dock location. The transmitter sends the preamble every 1 second. Then we moved the extension pole along different 1D trajectories parallel to the coast. To obtain the ground-truth trajectory, we placed a measurement tape along the dock coast, mounted a camera on the extension pole, and pointed it to

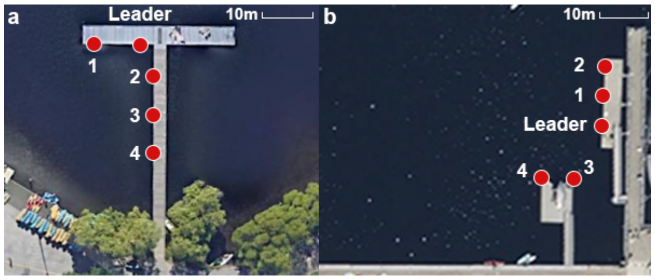

Figure 17: Testbeds used to evaluate our positioning system at the (a) dock and (b) boathouse locations.

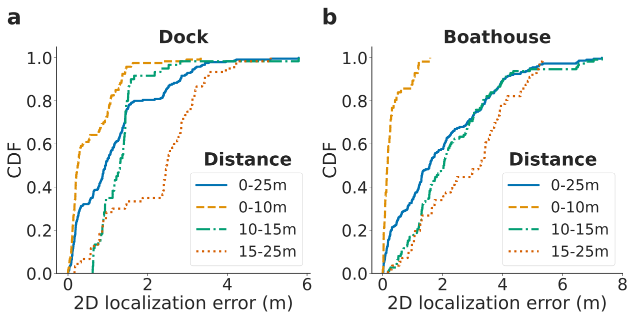

Figure 18: 2D localization error broken down by links of different distances at the (a) dock and (b) boathouse.

the measurement tape. Since the movement of the phone on the extension pole is parallel to the measurement tape, the ground truth of distance changes between the two phones can be recorded by the camera video (yellow lines in Fig. 15). As shown in Fig. 15a,b, the smartphone was moved at an average speed of 32 and 56 cm/s. The median and 95th percentile 1D errors were 0.51 and 1.17 m.

Leader orientation accuracy. We evaluate the ability for a dive leader to orient themselves to face a diver within their visual range. To do this, we measure the orientation error with respect to two users in a swimming pool at different distances. Both users wore a wristband (HCcolo Wristband Phone Holder) that held a smartphone in a waterproof case. One user (the diver) stayed in a stationary position at the end of the pool with their smartphone and also held a 4×5 checkerboard pattern used for camera calibration. The other user (the leader) was positioned at a fixed distance initially at a random orientation, and would then rotate their body and arm to directly face the stationary user. The leader's smartphone was set to capture video footage of the orienting motion, and the process was repeated at different distances. To compute the orientation error, we leverage algorithms from prior work [14] to calculate the position in world coordinates of the camera $C = (x_C, y_C, z_C)$, the stationary user's checkerboard/phone $P = (x_P, y_P, z_P)$, and the center of the camera frame $D = (x_D, y_D, z_D)$. We define the vector between the camera and the checkerboard/smartphone as $\vec{v}_{PC} = P - C$ and the vector between the camera and the center of the camera frame as $\vec{v}_{DC} = D - C$. We then compute the orientation error as the angle between the two vectors $cos^{-1}(\frac{\vec{v}_{PC} \cdot \vec{v}_{DC}}{|\vec{v}_{PC}||\vec{v}_{DC}|})$. If the checkerboard/smartphone is at the center of the image frame, the orientation error would be $0°$. Fig. 16 shows our evaluation with two different users as the lead diver. The average error across both users and distances is $5.0°$.

Battery life. We evaluate the power consumption of our system on an Apple Watch Ultra and Samsung Galaxy S9 smartphone. To do this, we had the smartwatch continuously play the Emergency SOS [20] siren, and had the smartphone continuously transmit the system preamble at maximum volume every three seconds. The sound levels produced by both devices were 85 and 88 dB SPL respectively at a distance of 1 m in air. The smartwatch and smartphone's battery power reduced by 90% and 63% respectively after a duration of 4.5 hours which is longer than the maximum recommended dive times for recreational scuba diving [13, 15].

3.2 Network testbed evaluation

2D localization accuracy versus device separation. We evaluate our system at the dock and boathouse locations using a network of five devices as shown in Fig. 17. The topologies were chosen such that the pair-wise distances between smartphone nodes spanned a range of distances from 3 to 25 m from the leader device. Each smartphone was submerged underwater with a rope at different depths. The ground truth for the 2D locations in the dock location was measured using a measuring tape. At the boathouse location, the 2D ground truth was obtained using the distance measuring tool on Google Maps as the devices were distributed onto two separate islands that were separated by a body of water. In each configuration, a total of approximately 240 measurements were collected. Each measurement was split across roughly 5 sessions, where the devices were retrieved from the water and submerged between measurements. Fig. 18 shows the CDF of the 2D localization errors for the devices at the dock and boathouse broken down by link distance to the leader device. The median (95%) errors across all devices were 0.9 m (3.2 m) and 1.6 m (4.9 m) in the two deployments. As excepted the error increases as the distance to the leader diver increases. This is acceptable since localizing a device at 20 m to within 3 m, is sufficient for our target application.

Effect of erroneous links. We evaluate this in the dock location by blocking the link between the leader and user 1 with a thick solid sheet attached to a telescopic extension pole. To ensure the link was completely occluded, we place the leader and user 1 at the same depth of 1.5 m. Note that despite being occluded, the devices can hear each other but have an erroneous distance estimate. The median and 95% error in this network deployment with our outlier detection algorithm applied were 1.4 and 3.4 m respectively. In Fig. 19a, we focus on the worse 10% of the localization errors, i.e., 90–100th percentile. We plot the CDF with and without our outlier detection algorithm applied. The plot shows that without outlier detection, our error estimates have a long tail due to the presence of occlusions. However, our outlier detection algorithm can address erroneous distance estimates and reduce 2D localization errors.

Effect of link removal. We use the data collected from the dock location and randomly remove a single random link for each of the measurements. Fig. 19b shows the 2D localization error with and without link removal. The main observation is that although the median errors between the two scenarios are similar with an error of 1.0 and 0.9 m respectively, the 95th percentile error with link removal is higher than a fully connected network with an error of 6.2 and 3.2 m respectively. The reason for this is that certain links

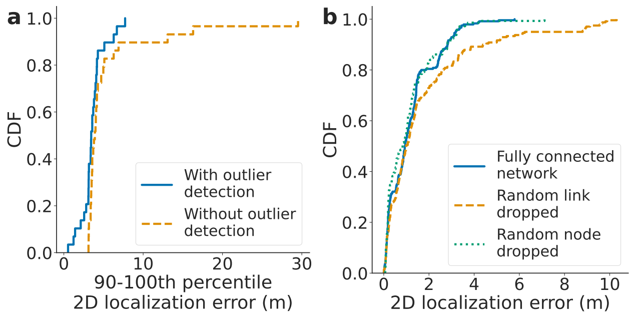

Figure 19: Effect of (a) erroneous links due to occlusions, and (b) link and node removal.

play a greater role in constraining the rotational ambiguities needed to distinguish between nodes that are positioned in a straight line. Dropping these links increases localization error. For randomly dropping nodes, the localization error does not increase and even become slightly better. This is because some of the far away devices have larger ranging errors. Sometimes dropping the "bad" far-away devices can improve topology estimation for the remaining devices.

4-device networks. We evaluate 4-device network deployments by randomly removing a node from the network using data from the dock location. Here, we measure the 2D localization error when randomly removing a device from the network, except for the leader and user 1. Fig. 19b shows that the CDF of 2D localization error of the 5-device and 4-device networks are similar with a median error of 0.9 and 0.8 m, and a 95th percentile error of 3.2 and 3.2 m.

Effect of mobility. We also evaluate whether the motion of the device affects our 2D localization. Specifically, we attach 5 phones to the ropes and place them as shown in Fig. 17. Then we moved a single device forward and backward around its original positions. Since the phone is attached to the rope, its orientation also keeps changing during movement. The speed of motion was between 15-50 cm/s. We performed this evaluation twice, first by moving only user 1 then by moving only user 2. The ground truth for the moving device was set to the midpoint of the trajectory. Our evaluation results in Fig. 20 show that the change in 2D localization error is modest with the median error for user 1 increasing from 0.2 to 0.3 m when it is moving, while the median error for user 2 increases from 0.4 to 0.8 m when it is moving. This is due to the distributed nature of our protocol which is able to tolerate multi-path variations caused by mobility in the environment.

Flipping disambiguation accuracy. We ran experiments at the dock environment with 5 devices (Fig. 17(a)). We used a long stick instead of the rope so that we can control the orientation of the leader device. We collect a total of 50 sets of localization data points with the leader device oriented towards a closeby device (either device 1 or 2). We run the flipping disambiguation algorithm in 2 settings: (1) we use signals from only one of the 3 devices (excluding leader and the visual device that the leader is pointing to) to resolve flipping as described in §2.1.4. (2) we use the signals from all other 3 devices to resolve flipping. Across all 50 experiments, when using only one device's signal the flipping disambiguation accuracy is 90.1%. When we use all three devices' signals, the flipping disambiguation accuracy improved to 100%, which is expected since this is a binary classification task and does not require precise AoA values.

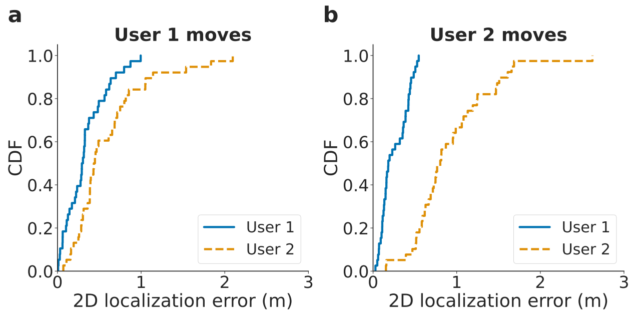

Figure 20: 2D localization errors for moving devices.

Localization protocol round-trip time measurement. Finally, we ran our distributed protocol with different number of devices in a testbed and measured the round-trip time for each 2D localization protocol run. For each configuration with a specific number of devices, we ran the protocol 40 times and calculated the average round time for the protocol. For 3, 4, 5, 6 and 7 devices, the mean round times were 1.2, 1.6, 1.9, 2.2, and 2.5 seconds, respectively.

4 RELATED WORK

While there is prior work on underwater communication and messaging [32, 47], here we focus on localization.

Anchor-based underwater localization. There has been prior interest in achieving underwater tracking for dive computers, sensors and robots [24, 30, 63, 68, 69, 71, 78, 85, 88, 89, 91, 92, 94, 106]. These proposals use time of arrival [31], time difference of arrival [33], angle of arrival [45] or signal strength [77] to estimate distances and angles from known anchor buoys (see [46] for a detailed survey).

[23] uses hydrophone devices on the surface as beacons with known locations. The diver then uses a hand-held display connected to an acoustic communication module [53]. [21] proposes the use of an underwater pinger with a very high precision clock that is synchronized to GPS, prior to deployment. The surface buoys may be equipped with GPS that can be translated to underwater GPS [22]. [31] computes the position based on time of arrival measurements for an underwater autonomous vehicle. [61] proposes the use of directional beacons for localization. [103] deploys a large number of visible passive tags which are used as anchors for underwater localization; in addition to requiring a large number of tags, visual systems have a smaller underwater range than acoustic systems.

Multi-hop localization [44, 66, 79, 82] has been explored for terrestrial and underwater ad-hoc sensor networks. Here, the distance to each anchor device is computed using intermediary devices at different hops. While these works are theoretical in nature, they provide avenues for localizing devices in ad-hoc networks. These methods however are primarily designed for anchor-based sensor networks, which is in contrast to our anchor-free design.

Anchor-free underwater localization. This problem formulation is under-explored for underwater settings. [43] investigates "active-restricted" sensors that are anchored to the bottom of the sea and leverages the limited motion within a hemispherical area centered at the anchor. [74] combines a discovery protocol and pairwise ranging to localize the relative coordination of the underwater sensor networks. [67] propose an algorithm to account for the motion effect of acoustic nodes and improve the self-localization accuracy. Our work is more focused on some practical problems: outlier detection, missing links, rotational and flipping ambiguities.

Anchor-free localization has also been explored for autonomous underwater vehicles to correct for their accumulated inertial errors [58, 80, 99].Autonomous underwater vehicles however can more precisely control and know their motion, velocity and instantaneous direction, which is difficult to do using a smart device that is wore on a mobile human. For example, [96] leverage the high-precision clock (drift rate < 1 ms over 14 hrs) and Doppler sonar to track the robot positions, which are not available on mobile devices. Finally some commercial products [11] claim to achieve anchor-free localization using custom hardware with precise inertial sensors. Our evaluation with smart device IMUs confirmed prior observations that they drift within a few seconds [101], making them challenging to use for anchor-free localization.

Underwater Synchronization. For time of arrival localization, clock synchronization is critical. [96] avoids clock drifting by deploying high-precision clocks (drift rate < 1 ms over 14 hrs). [48] investigates the use of expensive atomic clocks for underwater localization. [64, 95] calibrate the clock drifting during the system initialization. However, such calibration requires known initial positions of devices, which introduces initialization overhead. Moreover, the residual estimation error will still lead to drifting with time. [34, 59, 87] apply a two-way timestamp exchanging protocol for synchronized among different nodes. [60] utilizes the Doppler shift to estimate the clock skew and assumes the estimated velocity keeps constant within the sync interval. These methods require frequent message exchange to estimate the time offset and assume a higher bandwidth than is available on smart devices. In our system, instead of frequently synchronizing among devices with audible queries, synchronization only happens when the leader initiates localization, which is more appropriate for our application.

In-air localization. [75, 105] achieve in-air 1D acoustic ranging between two phones. 2D acoustic localization uses anchor devices as beacons [84] and microphone and speaker arrays [64, 95, 102] to perform either triangulation at distances of a few meters or AoA estimation. In contrast, our target application requires localization at distances of 30-40 m. Accurate underwater AoA estimation requires microphone separation on the order of a meter at 1-4 kHz, which is an order of magnitude larger than a mobile device.

Distributed localization has been theoretically explored for adhoc anchor-free sensor networks [29, 37, 76]. [29, 76] use network embeddings but do not address rotational and flipping ambiguities, while [37] assumes that each device is capable of measuring the angle of arrival from other devices. The closest to our work are [38, 70] which achieve distributed in-air acoustic localization. [70] is designed for a network with 16-40 sensors but assumes that the pair-wise distance errors are 1-5 cm, which is an order of magnitude lower than in underwater scenarios. [38] is limited in three key aspects: 1) it requires at least 10-15 devices to provide localization results. A dive party is typically much less than 10 divers, 2) it does not address rotational and flipping ambiguities, and 3) it explicitly assumes that all devices are in range of each other and no occlusions exist between any device pairs.

5 DISCUSSION AND CONCLUSION

We present the first underwater acoustic positioning system for smart devices. Our software system achieves 3D positioning on commodity devices without external infrastructure. Here, we discuss the limitations of our current design.

When does it fail? Sometimes our localization system, like any wireless scheme, produces high localization errors, as demonstrated by the long tail in Fig. 19. A topology-based design must fulfill three key requirements: the network must be connected, each node must have at least three links, and the links must be "well distributed" across the nodes (see §2.1). Additionally, if a large number of pairwise distances are erroneous, or if we encounter degenerate cases where say all devices are in a straight line, the localization results may be impacted. Finally, our approach necessitates at least three divers. With an increase in the number of divers, the design becomes more resilient to erroneous pairwise measurements. With two divers, we can only provide ranging information.

Two-hop communication. Our distributed timestamp protocol (§2.3) is designed to function even when not all devices are within range of the leader device. Additionally, our missing link evaluation (Fig. 19) drops the direct link to the leader. However, our current implementation assumes that all devices are only one hop away from the leader device, even if the link quality is imperfect.

Localization versus tracking. Our system enables a device to calculate the 3D locations of other devices in relation to itself. However, this is initiated by the user and is not a continuous tracking system. This is a deliberate design choice, as it minimizes the duration of acoustic signals underwater. Future work is necessary to develop a continuous tracking system that could potentially perform sensor fusion with other sensors, without continuous use of acoustics.

Audibility. As with prior work [32], we use 1-5 kHz for acoustic pairwise distance estimation. These are in the human hearing range, similar to many of the commercial and research modems [4, 26, 47]. To limit the time duration we use acoustic signals underwater, our system is designed to be an user-initiated action that is only performed when the leader wants to know the positions of their dive group and is not designed for continuous tracking.

Visual odometry. A design decision we made in our paper is to not use cameras for localization despite they being ubiquitous on smartphones. We opted not to use cameras since our goal is to create a design that works for wearables like smart watches which are more likely than phones to be used in underwater scenarios. Cameras unfortunately are not yet common on smart watches. Light-based methods are also susceptible to turbid water [55].

Diver evaluations. In this paper, we mainly use ropes and long sticks to put the devices underwater during evaluation. Further work is required to evaluate our system with divers in real-world dives (seaside or deep ocean) and with multiple divers constantly moving.

Acknowledgments. The researchers are funded by the Moore Inventor Fellow award #10617 and the National Science Foundation. We thank our shepherd, Fadel Adib, Malek Itani, Antonio Glenn, Bandhav Veluri and Maruchi Kim for their feedback.

This work does not raise ethical issues.

REFERENCES

[1] 2004. Muddy Waters: Techniques for Low-Vis Diving. https://dtmag.com/thelibrary/muddy-waters-techniques-low-vis-diving/. (2004).

[2] 2011. Diver drowns in fishing net. https://scubaboard.com/community/threads/diver-drowns-in-fishing-net-bc-canada.391393/. (2011).

[3] 2018. Buddy Diving For Safety and Support. https://www.divein.com/diving/buddy-diving-for-safety/. (2018).

[4] 2018. Evo Logics 7/17 communication and positioning devices. https://evologics.de/acoustic-modem/7-17. (2018).

[5] 2020. Scuba Equipment Issues Get This Diver in Deep Trouble. https://www.scubadiving.com/scuba-equipment-issues-get-this-diver-in-deep-trouble. (2020).

[6] 2021. Audio Latency. https://developer.android.com/ndk/guides/audio/audio-latency. (2021).

[7] 2021. Become a Certified Scuba Diver FAQs. (2021). https://www.padi.com/help/scuba-certification-faq

[8] 2021. How Deep Can You Scuba Dive? (2021). https://www.scubadiving.com/why-is-130-feet-depth-limit-for-recreational-scuba-diving

[9] 2021. Universal Waterproof Case, Waterproof Phone Pouch Compatible for iPhone 13 12 11 Pro Max XS Max XR X 8 7 Samsung Galaxy s10/s9 Google Pixel 2 HTC Up to 7.0", IPX8 Cellphone Dry Bag -2 Pack. (2021). https://www.amazon.com/gp/product/B08S3SG5KF/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1

[10] 2022. Apple Watch Ultra. https://www.apple.com/apple-watch-ultra/. (2022).

[11] 2022. Ariadna: Personal underwater navigation technology. http://ariadna.tech/. (2022).

[12] 2022. The best waterproof phones you can buy. https://www.androidauthority.com/best-waterproof-phones-718588/. (2022).

[13] 2022. How Long Can You SCUBA Dive? (2022). https://www.wetsuitwearhouse.com/blog/how-long-can-you-scuba-dive/

[14] 2022. Measuring Planar Objects with a Calibrated Camera. (2022). https://www.mathworks.com/help/vision/ug/measuring-planar-objects-with-a-calibrated-camera.html

[15] 2022. PADI Recreational Dive Table Planner. (2022). https://www.a1scubadiving.com/wp-content/uploads/2018/06/PADI-Recreational-Dive-Table-Planner.pdf

[16] 2022. Reach new depths with the Oceanic+ app and Apple Watch Ultra, https://www.apple.com/newsroom/2022/11/reach-new-depths-with-the-oceanic-plus-app-and-apple-watch-ultra/. (2022).

[17] 2022. Use the Depth app on Apple Watch Ultra, https://support.apple.com/en-us/HT213334. (2022).

[18] 2022. Watch-style dive computers. https://www.garmin.com/en-US/c/sports-fitness/dive-computers-smartwatches/. (2022).

[19] 2023. The best waterproof phones 2022. https://www.tomsguide.com/best-picks/best-waterproof-and-water-resistant-phones. (2023).

[20] 2023. Use Emergency SOS on your Apple Watch. (2023). https://support.apple.com/en-us/HT206983

[21] Alex Alcocer, Paulo Oliveira, and Antonio Pascoal. 2004. Study and implementation of an EKF GIB-based underwater positioning system. *IFAC Proceedings Volumes* 37 (07 2004), 383–390. https://doi.org/10.1016/S1474-6670(17)31762-7

[22] Alex Alcocer, Paulo Oliveira, and Antonio Pascoal. 2006. Underwater acoustic positioning systems based on buoys with GPS. (01 2006).

[23] Prasad Anjangi, Amy Gibson, Manu Ignatius, Chinmay Pendharkar, Anne Kurian, Alan Low, and Mandar Chitre. 2020. Diver Communication and Localization System using Underwater Acoustics. In *Global Oceans 2020: Singapore – U.S. Gulf Coast*. 1–8. https://doi.org/10.1109/IEEECONF38699.2020.9389462

[24] Humberto Barberá, Pablo Bernal-Polo, and David Herrero-Perez. 2021. Sensor Modeling for Underwater Localization Using a Particle Filter. *Sensors* 21 (02 2021), 1549. https://doi.org/10.3390/s21041549

[25] Leonid Blouvshtein and Daniel Cohen-Or. 2018. Outlier detection for robust multi-dimensional scaling. *IEEE transactions on pattern analysis and machine intelligence* 41, 9 (2018), 2273–2279.

[26] Joe Borden and Jeffery DeArruda. 2012. Long range acoustic underwater communication with a compact AUV. In *2012 Oceans*. 1–5. https://doi.org/10.1109/OCEANS.2012.6405091

[27] Ingwer Borg and Patrick JF Groenen. 2005. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media.

[28] Chao Cai, Menglan Hu, Xiaoqiang Ma, Kai Peng, and Jiangchuan Liu. 2018. Accurate ranging on acoustic-enabled IoT devices. *IEEE Internet of Things Journal* 6, 2 (2018), 3164–3174.

[29] S. Capkun, M. Hamdi, and J.-P. Hubaux. 2001. GPS-free positioning in mobile ad-hoc networks. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences*. 10 pp.–. https://doi.org/10.1109/HICSS.2001.927202

[30] Gianni Cario, Alessandro Casavola, Gianfranco Gagliardi, Marco Lupia, and Umberto Severino. 2021. Accurate Localization in Acoustic Underwater Localization Systems. *Sensors* 21, 3 (2021), 762.

[31] Thomas Casey, Brian Guimond, and James Hu. 2007. Underwater Vehicle Positioning Based on Time of Arrival Measurements from a Single Beacon. In *OCEANS 2007*. 1–8. https://doi.org/10.1109/OCEANS.2007.4449186

[32] Tuochao Chen, Justin Chan, and Shyamnath Gollakota. 2022. Underwater Messaging Using Mobile Devices. In *Proceedings of the ACM SIGCOMM 2022 Conference (SIGCOMM '22)*. Association for Computing Machinery, New York, NY, USA, 545–559. https://doi.org/10.1145/3544216.3544258

[33] Xiuzhen Cheng, Haining Shu, Qilian Liang, and David Hung-Chang Du. 2008. Silent Positioning in Underwater Acoustic Sensor Networks. *IEEE Transactions on Vehicular Technology* 57, 3 (2008), 1756–1766. https://doi.org/10.1109/TVT.2007.912142

[34] Nitthita Chirdchoo, Wee-Seng Soh, and Kee Chaing Chua. 2008. MU-Sync: a time synchronization protocol for underwater mobile networks. In *Proceedings of the 3rd International Workshop on Underwater Networks*. 35–42.

[35] Jan De Leeuw. 2005. Applications of convex analysis to multidimensional scaling. (2005).

[36] Jan De Leeuw and Patrick Mair. 2009. Multidimensional scaling using majorization: SMACOF in R. *Journal of statistical software* 31 (2009), 1–30.

[37] Tao Du, Shouning Qu, Guo Qingbei, and Zhu Lianjiang. 2017. A simple efficient anchor-free node localization algorithm for wireless sensor networks. *International Journal of Distributed Sensor Networks* 13 (04 2017), 155014771770578. https://doi.org/10.1177/1550147717705784

[38] Viktor Erdélyi, Trung-Kien Le, Bobby Bhattacharjee, Peter Druschel, and Nobutaka Ono. 2018. Sonoloc: Scalable Positioning of Commodity Mobile Devices. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '18)*. Association for Computing Machinery, New York, NY, USA, 136–149. https://doi.org/10.1145/3210240.3210324

[39] Pedro A Forero and Georgios B Giannakis. 2012. Sparsity-exploiting robust multidimensional scaling. *IEEE Transactions on Signal Processing* 60, 8 (2012), 4118–4134.

[40] Reza Ghaffarivardavagh, Sayed Saad Afzal, Osvy Rodriguez, and Fadel Adib. 2020. Underwater Backscatter Localization: Toward a Battery-Free Underwater GPS. In *Proceedings of the 19th ACM Workshop on Hot Topics in Networks (HotNets '20)*. Association for Computing Machinery, New York, NY, USA, 125–131. https://doi.org/10.1145/3422604.3425950

[41] D.K. Goldenberg, A. Krishnamurthy, W.C. Maness, Y.R. Yang, A. Young, A.S. Morse, and A. Savvides. 2005. Network localization in partially localizable networks. In *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, Vol. 1. 313–326 vol. 1. https://doi.org/10.1109/INFCOM.2005.1497902

[42] Mario Guggenberger, Mathias Lux, and Laszlo Böszörmenyi. 2015. An analysis of time drift in hand-held recording devices. In *International Conference on Multimedia Modeling*. Springer, 203–213.

[43] Ying Guo and Yutao Liu. 2013. Localization for anchor-free underwater sensor networks. *Computers Electrical Engineering* 39 (08 2013), 1812–1821. https://doi.org/10.1016/j.compeleceng.2013.02.001

[44] Huai Huang and Rosa Zheng. 2018. Node localization with AoA assistance in multi-hop underwater sensor networks. *Ad Hoc Networks* 78 (09 2018), 32–41. https://doi.org/10.1016/j.adhoc.2018.05.005

[45] Huai Huang and Yahong Rosa Zheng. 2016. AoA assisted localization for underwater Ad-Hoc sensor networks. In *OCEANS 2016 MTS/IEEE Monterey*. 1–6. https://doi.org/10.1109/OCEANS.2016.7761388

[46] Tariq Islam and Seok-Hwan Park. 2020. A Comprehensive Survey of the Recently Proposed Localization Protocols for Underwater Sensor Networks. *IEEE Access* 8 (2020), 179224–179243. https://doi.org/10.1109/ACCESS.2020.3027820

[47] Junsu Jang and Fadel Adib. 2019. Underwater Backscatter Networking. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 187–199. https://doi.org/10.1145/3341302.3342091

[48] K Kebkal, O Kebkal, E Glushko, V Kebkal, L Sebastiao, A Pascoal, J Gomes, J Ribeiro, H Silva, M Ribeiro, et al. 2017. Underwater acoustic modems with integrated atomic clocks for one-way travel-time underwater vehicle positioning. In *Proceefings of the Underwater Acoustics Conference and Exhibition (UACE)*.

[49] Benjamin Kempke, Pat Pannuto, and Prabal Dutta. 2015. PolyPoint: Guiding Indoor Quadrotors with Ultra-Wideband Localization. In *Proceedings of the 2nd International Workshop on Hot Topics in Wireless (HotWireless '15)*. Association for Computing Machinery, New York, NY, USA, 16–20. https://doi.org/10.1145/2799650.2799651

[50] Liu Kewen et al. 2010. Research of MMSE and LS channel estimation in OFDM systems. In *The 2nd international conference on information science and engineering*. IEEE, 2308–2311.

[51] Moses A Koledoye, Tullio Facchinetti, and Luis Almeida. 2017. MDS-based localization with known anchor locations and missing tag-to-tag distances. In *2017 22nd IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 1–4.

[52] Manikanta Kotaru, Kiran Joshi, Dinesh Bharadia, and Sachin Katti. 2015. SpotFi: Decimeter Level Localization Using WiFi. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication (SIGCOMM '15)*. Association for Computing Machinery, New York, NY, USA, 269–282. https://doi.org/10.1145/2785956.2787487

[53] Benjamin Kuch, Giorgio Buttazzo, Elaine Azzopardi, Martin Sayer, and Arne Sieber. 2012. GPS diving computer for underwater tracking and mapping. *Underwater Technology The International Journal of the Society for Underwater* 189 (07 2012), 189–194. https://doi.org/10.3723/ut.30.189

[54] William A Kuperman and Philippe Roux. 2007. Underwater acoustics. (2007), 149–204 pages.

[55] Philip Lacovara. 2008. High-bandwidth underwater communications. *Marine Technology Society Journal* 42, 1 (2008), 93–102.

[56] Gerard Laman. 1970. On graphs and rigidity of plane skeletal structures. *Journal of Engineering mathematics* 4, 4 (1970), 331–340.

[57] Ying Li, Bridget Benson, Ryan Kastner, and Xing Zhang. 2009. Bit Error Rate, Power and Area Analysis of Multiple FPGA Implementations of Underwater FSK. (2009).

[58] Yichen Li, Yiyin Wang, and Wenbin Yu. 2019. Multiple Autonomous Underwater Vehicle Cooperative Localization in Anchor-Free Environments. *IEEE Journal of Oceanic Engineering* PP (09 2019), 1–17. https://doi.org/10.1109/JOE.2019.2935516

[59] Li Liu, Yang Xiao, and Jingyuan Zhang. 2009. A linear time synchronization algorithm for underwater wireless sensor networks. In *2009 IEEE International Conference on Communications*. IEEE, 1–5.

[60] Feng Lu, Diba Mirza, and Curt Schurgers. 2010. D-sync: Doppler-based time synchronization for mobile underwater sensor networks. In *Proceedings of the 5th International Workshop on Underwater Networks*. 1–8.

[61] Hanjiang Luo, Yiyang Zhao, Zhongwen Guo, Siyuan Liu, Pengpeng Chen, and Lionel M. Ni. 2008. UDB: Using Directional Beacons for Localization in Underwater Sensor Networks. In *2008 14th IEEE International Conference on Parallel and Distributed Systems*. 551–558. https://doi.org/10.1109/ICPADS.2008.31

[62] Lovász László and Y. Yemini. 1982. On Generic Rigidity in the Plane. *SIAM J. Algebraic Discrete Methods* 3 (03 1982), 91–98. https://doi.org/10.1137/0603009

[63] Tara Maki, Takumi Matsuda, Takashi Sakamaki, Tamaki Ura, and Junichi Kojima. 2013. Navigation Method for Underwater Vehicles Based on Mutual Acoustical Positioning With a Single Seafloor Station. *Oceanic Engineering, IEEE Journal of* 38 (01 2013), 167–177. https://doi.org/10.1109/JOE.2012.2210799

[64] Wenguang Mao, Jian He, and Lili Qiu. 2016. CAT: high-precision acoustic motion tracking. In *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 69–81.

[65] Delphin Raj Kesari Mary, Eunbi Ko, Seung-Geun Kim, Sun-Ho Yum, Soo-Young Shin, and Soo-Hyun Park. 2021. A systematic review on recent trends, challenges, privacy and security issues of underwater internet of things. *Sensors* 21, 24 (2021), 8262.

[66] J. Mass-Sanchez, E. Ruiz-Ibarra, Joaquin Cortez, Adolfo Espinoza, and Luis Castro. 2017. Weighted Hyperbolic DV-Hop Positioning Node Localization Algorithm in WSNs. *Wireless Personal Communications* 96 (10 2017). https://doi.org/10.1007/s11277-016-3727-5

[67] Diba Mirza and Curt Schurgers. 2008. Motion-aware self-localization for underwater networks. In *Proceedings of the 3rd International Workshop on Underwater Networks*. 51–58.

[68] Martín Monteiro and Arturo C. Marti. 2020. Using smartphones as hydrophones: two experiments in underwater acoustics. *arXiv: Physics Education* (2020).

[69] Hyun Moon, Chang Ho Yu, and Jae Weon Choi. 2009. Performance of Sensor Localization in Underwater Wireless Sensor Networks. https://doi.org/10.13140/2.1.5126.4649

[70] David Moore, John Leonard, Daniela Rus, and Seth Teller. 2004. Robust Distributed Network Localization with Noisy Range Measurements. In *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems (SenSys '04)*. Association for Computing Machinery, New York, NY, USA, 50–61. https://doi.org/10.1145/1031495.1031502

[71] Andrea Munafo, Jan Sliwka, and Joao Alves. 2015. Dynamic placement of a constellation of surface buoys for enhanced underwater positioning. 1–6. https://doi.org/10.1109/OCEANS-Genova.2015.7271663

[72] Ali A Nasir, Salman Durrani, and Rodney A Kennedy. 2010. Performance of coarse and fine timing synchronization in OFDM receivers. In *2010 2nd International Conference on Future Computer and Communication*, Vol. 2. IEEE, V2–412.

[73] OpenStax. 2016. College Physics. (2016). http://cnx.org/contents/031da8d3-b525-429c-80cf-6c8ed997733a@9.35

[74] Al-Khalid Othman. 2008. GPS-less localization protocol for underwater acoustic networks. In *2008 5th IFIP International Conference on Wireless and Optical Communications Networks (WOCN'08)*. IEEE, 1–6.

[75] Chunyi Peng, Guobin Shen, Yongguang Zhang, Yanlin Li, and Kun Tan. 2007. Beepbeep: a high accuracy acoustic ranging system using cots mobile devices. In *Proceedings of the 5th international conference on Embedded networked sensor systems*. ACM, 1–14.

[76] N.B. Priyantha, Hari Ram Balakrishnan, Erik D. Demaine, and Seth J. Teller. 2003. Anchor-Free Distributed Localization in Sensor Networks.

[77] Qin Qin, Yi Tian, and Xin Wang. 2021. Three-Dimensional UWSN Positioning Algorithm Based on Modified RSSI Values. *Mobile Information Systems* 2021 (06 2021), 1–8. https://doi.org/10.1155/2021/5554791

[78] Dinesh Kumar Sah, Tu N. Nguyen, Manjusha Kandulna, Korhan Cengiz, and Tarachand Amgoth. 2022. 3D Localization and Error Minimization in Underwater Sensor Networks. 18, 3, Article 31 (sep 2022), 25 pages. https://doi.org/10.1145/3460435

[79] Souvik Saha and Rajeev Arya. 2022. ARCMT: Anchor node-based range free cooperative multi trusted secure underwater localization using fuzzifier. *Computer Communications* 193 (07 2022). https://doi.org/10.1016/j.comcom.2022.07.016

[80] Georgios Salavasidis, Andrea Munafo, Catherine Harris, Thomas Prampart, Rob Templeton, Micheal Smart, Daniel Roper, Miles Pebody, Stephen McPhail, Eric Rogers, and Alexander Phillips. 2018. Terrain-Aided Navigation for Long-Endurance and Deep-Rated Autonomous Underwater Vehicles. *Journal of Field Robotics* 36 (11 2018), 447–474. https://doi.org/10.1002/rob.21832

[81] Stefania Sesia, Issam Toufik, and Matthew Baker. 2011. *LTE-the UMTS long term evolution: from theory to practice*. John Wiley & Sons.

[82] Syed Bilal Shah, Fuliang Yin, Inam Khan, and Seema Begum. 2018. 3D weighted centroid algorithm RSSI ranging model strategy for node localization in WSN based on smart devices. *Sustainable Cities and Society* Volume 39 (03 2018), 298–308. https://doi.org/10.1016/j.scs.2018.02.022

[83] Arkadiy Skopenkov. 2006. *Embedding and knotting of manifolds in Euclidean spaces*. Vol. 347. 248–342. https://doi.org/10.1017/CBO9780511666315.008

[84] Adam Smith, Hari Balakrishnan, Michel Goraczko, and Nissanka Priyantha. 2004. Tracking moving devices with the cricket location system. In *Proceedings of the 2nd international conference on Mobile systems, applications, and services*. ACM, 190–202.

[85] Ramji Srinivasan, S. Rajesh, N.R. Ramesh, S.M. Babu, Raju Abraham, Deepak Raphael, G.A. Ramadass, and Ma Atmanand. 2007. Design and testing of Control and Positioning System for Underwater mining Machine. 1 – 5. https://doi.org/10.1109/OCEANS.2007.4449146

[86] Meihong Sui, Xinsheng Yu, and Fengli Zhang. 2009. The evaluation of modulation techniques for underwater wireless optical communications. In *2009 International Conference on Communication Software and Networks*. IEEE, 138–142.

[87] Affan A Syed and John Heidemann. 2006. Time synchronization for high latency acoustic networks. In *Proceedings IEEE INFOCOM 2006. 25TH IEEE International Conference on Computer Communications*. IEEE, 1–12.

[88] Hwee Tan, Roee Diamant, Winston Seah, and Marc Waldmeyer. 2011. A Survey of Techniques and Challenges in Underwater Localization. *Ocean Engineering - OCEAN ENG* 38 (10 2011), 1663–1676. https://doi.org/10.1016/j.oceaneng.2011.07.017

[89] H.-P. Tan, A. F. Gabor, Z. A. Eu, and W. K. G. Seah. 2010. A Wide Coverage Positioning System (WPS) for Underwater Localization. In *2010 IEEE International Conference on Communications*. 1–5. https://doi.org/10.1109/ICC.2010.5501950

[90] Amin Y. Teymorian, Wei Cheng, Liran Ma, Xiuzhen Cheng, Xicheng Lu, and Zexin Lu. 2009. 3D Underwater Sensor Network Localization. *IEEE Transactions on Mobile Computing* 8, 12 (2009), 1610–1621.

[91] Arkadiusz Tomczak. 2011. MODERN METHODS OF UNDERWATER POSITIONING APPLIED IN SUBSEA MINING. *AGH Journals of Mining and Geoengineering* 35 (01 2011), 381–394.

[92] Inam Ullah, Yiming Liu, Xin Su, and Pankoo Kim. 2019. Efficient and accurate target localization in underwater environment. *IEEE Access* 7 (2019), 101415–101426.

[93] Paul Van Walree, Helge Buen, and Roald Otnes. 2014. A performance comparison between DSSS, M-FSK, and frequency-division multiplexing in underwater acoustic channels. In *2014 Underwater Communications and Networking (UComms)*. IEEE, 1–5.

[94] Keith Vickery. 1998. Acoustic positioning systems. A practical overview of current systems. In *Proceedings of the 1998 Workshop on Autonomous Underwater Vehicles (Cat. No. 98CH36290)*. IEEE, 5–17.

[95] Anran Wang and Shyamnath Gollakota. 2019. MilliSonic: Pushing the Limits of Acoustic Motion Tracking. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems (CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3290605.3300248

[96] Sarah E Webster, Ryan M Eustice, Hanumant Singh, and Louis L Whitcomb. 2009. Preliminary deep water results in single-beacon one-way-travel-time acoustic navigation for underwater vehicles. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2053–2060.

[97] Yang Wen, Wei Huang, and Zhongpei Zhang. 2006. CAZAC sequence and its application in LTE random access. In *2006 IEEE Information Theory Workshop-ITW'06 Chengdu*. IEEE, 544–547.

[98] Wayne D Wilson. 1960. Equation for the speed of sound in sea water. *The Journal of the Acoustical Society of America* 32, 10 (1960), 1357–1357.

[99] Yan-xin Xie, Jun Liu, Cheng-quan Hu, Jun-hong Cui, and Hongli Xu. 2016. AUV dead-reckoning navigation based on neural network using a single accelerometer. 1–5. https://doi.org/10.1145/2999504.3001081

[100] Jie Xiong and Kyle Jamieson. 2013. ArrayTrack: A Fine-Grained Indoor Location System. In *Proceedings of the 10th USENIX Conference on Networked Systems Design and Implementation (nsdi'13)*. USENIX Association, USA, 71–84.

[101] Sangki Yun, Yi-Chao Chen, and Lili Qiu. 2015. Turning a Mobile Device into a Mouse in the Air. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services (MobiSys '15)*. Association for Computing Machinery, New York, NY, USA, 15–29. https://doi.org/10.1145/2742647.2742662

[102] Cheng Zhang, Qiuyue Xue, Anandghan Waghmare, Sumeet Jain, Yiming Pu, Sinan Hersek, Kent Lyons, Kenneth A Cunefare, Omer T Inan, and Gregory D Abowd. 2017. Soundtrak: Continuous 3d tracking of a finger using active acoustics. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1, 2 (2017), 30.

[103] Xiao Zhang, Hanqing Guo, James Mariani, and Li Xiao. 2022. U-Star: An Underwater Navigation System Based on Passive 3D Optical Identification Tags. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking (MobiCom '22)*. Association for Computing Machinery, New York, NY, USA, 648–660. https://doi.org/10.1145/3495243.3517019

[104] Yongzhao Zhang, Wei-Hsiang Huang, Chih-Yun Yang, Wen-Ping Wang, Yi-Chao Chen, Chuang-Wen You, Da-Yuan Huang, Guangtao Xue, and Jiadi Yu. 2020. Endophasia: Utilizing acoustic-based imaging for issuing contact-free silent speech commands. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 1 (2020), 1–26.

[105] Zengbin Zhang, David Chu, Xiaomeng Chen, and Thomas Moscibroda. 2012. Swordfight: Enabling a new class of phone-to-phone action games on commodity phones. In *Proceedings of the 10th international conference on Mobile systems, applications, and services*. ACM, 1–14.

[106] Shuang Zhao, Zhenjie Wang, Kaifei He, and Ning Ding. 2018. Investigation on underwater positioning stochastic model based on acoustic ray incidence angle. *Applied Ocean Research* 77 (06 2018), 69–77. https://doi.org/10.1016/j.apor.2018.05.011

APPENDIX

Low-level audio timing. Our goal is to ensure that the replying device can send a preamble at a precise sample index in the future that corresponds to t_{reply}, after the signal from the sender arrives at the device. To map the sample indices to time, we need to look into how Android transmits and records sound at a low level. The low-level OpenSL ES audio library in Android exposes access to the speaker and microphone audio sample buffers. Specifically, the library provides the ability to directly write audio samples to a future speaker buffer even during speaker playback. During runtime, the library executes a *CallBack* function when the microphone buffer is full or the speaker buffer is empty. In this way, we can acquire a continuous data stream for microphone data and another continuous data stream for speaker data. Thus, the sample indices in the microphone and speaker streams have a linear relationship with timestamps:

$$t_s(n) = n/f_s^s + t_s^0, \quad t_m(m) = m/f_s^m + t_m^0 \tag{1}$$

Here m and n are the sample indices in the microphone and speaker data streams. $t_s(n)$ is the timestamp when sample n in the buffer is send out by the speaker and $t_m(m)$ is the timestamp when sample m arrives in the microphone buffer. t_s^0 is the initial timestamp when the first sample in the stream is sent by the speaker, and t_m^0 is the initial timestamp when the first sample in the stream arrives at the microphone. f_s^s and f_s^m are the sampling rates for the speaker and microphone, which may not be exactly our desired sampling rate (f_s = 44.1 kHz). We assume that $f_s^s = f_s/(1-\alpha)$ and $f_s^m = f_s/(1-\beta)$, where $|\alpha| \ll 1$ and $|\beta| \ll 1$.

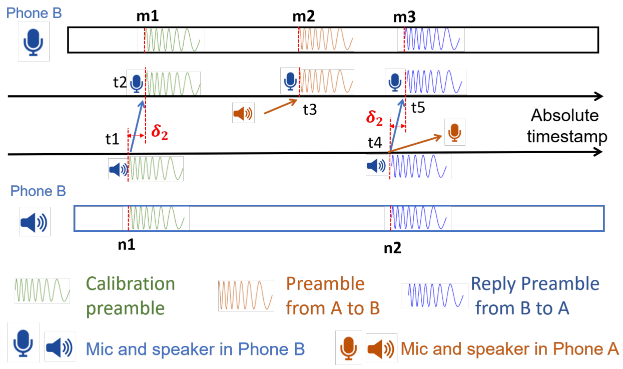

Figure 21: Mapping buffer samples to absolute time.

Self-synchronizing speaker and microphone streams. As shown in Fig. 21, we do not know the exact timestamp t_3 when the preamble arrived at the microphone of device B. Instead, we only know the sample index m_2 of the recorded preamble in the microphone stream. At the speaker side, we also cannot directly know the exact timestamp t_4 when the speaker sends the signal, but we can control the sample index n_2 in the speaker stream. We define the δ_2 is the propagation delay from the speaker to its own microphone. We define t_{reply} as the time interval between the arrival of signal from phone A and the arrival of its own signal at the phone B's microphone. According to Fig. 21 we have $t_{reply} = t_5 - t_3 = t_4 + \delta_2 - t_3$. Combining this with Eq. 1, we have

$$t_{reply} = t_4 + \delta_2 - t_3 = n_2/f_s^s + t_s^0 + \delta_2 - m_2/f_s^m - t_m^0 \tag{2}$$

The microphone and speaker buffers work separately, and there is no guarantee of the relative order between the two buffers. In other words, the initial offsets t_s^0 and t_m^0 can be different each time we open the streams. To address this, once we open the microphone and speaker data streams, we do not close them so as to keep the offset, $t_s^0 - t_m^0$, constant. We write zeros to the speaker stream when we are transmitting nothing to keep the buffer full. Further, after initializing the two streams, the speaker sends a calibration signal (green in Fig. 21) to estimate this offset. We write the calibration signal to the sample index n_1 in the speaker stream. Then the microphone stream would receive this calibration signal at sample index m_1. The propagation time of the calibration signal from the speaker to the microphone on device B is $(t_2 - t_1)$ (i.e., δ_2). By applying Eq. 1, we get:

$$t_2 - t_1 = m_1/f_s^m + t_m^0 - n_1/f_s^s - t_s^0 = \delta_2 \tag{3}$$

Now, we compute the offset $(n1 - m1)$ between the microphone and speaker after initial calibration, which can be used for $(t_s^0 - t_m^0)$ compensation. After initial calibration, our goal is when device B detects the start of the signal from device A at the sample index m_2 in Fig. 21, device B will write the reply signal at the sample index n_2 in the speaker stream, such that they are separated in time by the desired gap, t_{reply}^0, after adjusting for buffer delays. So, by compensating the indices offset acquired from calibration, we set n_2 to,

$$n_2 = m_2 + (n_1 - m_1) + fs \cdot t_{reply}^0, \tag{4}$$

Here f_s is the desired sampling rate. However the real reply interval t_{reply} is shown in Eq. 2. The difference between the real and desired

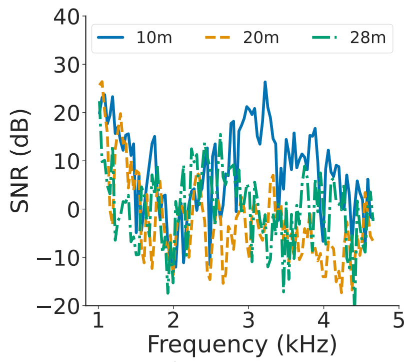

Figure 22: SNR estimation between two Samsung S9 devices at different distances.

reply times is,

$$t_{reply} - t^0_{reply} = n_2/f^s_s + t^0_s - m_2/f^m_s - t^0_m - t^0_{reply} + \delta_2 \tag{5}$$

By combining Eqs. 3 and 5 and using the relationship between f_s, f^m_s, f^s_s, we can rewrite the above equation as,

$$t_{reply} - t^0_{reply} = -\alpha t^0_{reply} + \frac{(m_2 - m_1)(\beta - \alpha)}{f_s} \tag{6}$$

The key observation here is that the main error source is from the difference between the actual sampling rate and the nominal sampling rate. α for Android devices is around 1-80 ppm [42]. As for the second term, while $\beta - \alpha$ is the clock drifting difference between speaker and microphone clock, which is usually quite small in most Android phone. To avoid the second error term accumulate as $m_2 - m_1$ increases, we can utilize the response signal of this device to re-calibrate the offset between the speaker and microphone streams.

SNR measurement. We measure the Signal-to-Noise Ratio (SNR) at 10, 20, and 28 m at the boathouse. During the measurement, there were people fishing and boating nearby. To estimate the SNR, we send a preamble consisting of 8 OFDM symbols from 1-5 KHz. We compute the SNR for each subcarrier by applying frequency-domain MMSE channel estimation [32]. Fig. 22 shows the estimated SNR for each subcarrier between 2 Samsung S9 phones.