Finding MARLIN: Exploiting Multi-Modal Communications for Reliable and Low-latency Underwater Networking

Stefano Basagni⋆, Valerio Di Valerio†, Petrika Gjanci†, Chiara Petrioli†
⋆ ECE Dept., Northeastern University, Boston, MA, basagni@ece.neu.edu
† Dipartimento di Informatica, Università di Roma "La Sapienza," Roma, Italy, {divalerio,gjanci,petrioli}@di.uniroma1.it

***Abstract*—This paper concerns the smart exploitation of multi-modal communication capabilities of underwater nodes to enable reliable and swift underwater networking. To contrast adverse and highly varying channel conditions we define a smart framework enabling nodes to acquire knowledge on the quality of the communication to neighboring nodes over time. Following a model-based reinforcement learning approach, our framework allows senders to select the best forwarding relay for its data jointly with the best communication device to reach that relay. We name the resulting forwarding method MARLIN, for Multi-modAl Reinforcement Learning-based RoutINg. Applications can choose whether to seek reliable routes to the destination, or whether faster packet delivery is more desirable. We evaluate the performance of MARLIN in varying networking scenarios where nodes communicate through two acoustic modems with widely different characteristics. MARLIN is compared to state-of-the-art forwarding protocols, including a channel-aware solution, a machine learning-based solution and to a flooding protocol extended to use multiple modems. Our results show that a smartly learned selection of relay and modem is key to obtain a packet delivery ratio that is twice as much that of other protocols, while maintaining low latencies and energy consumption.**

***Index Terms*—Underwater Wireless Sensor Networks, multi-modal communications, reinforcement learning-based routing.**

I. Introduction

In the past decade a marked shift in underwater sensing and communication capabilities has produced a flurry of research and development activities, and has propelled underwater wireless sensor network technology to new levels of possibilities and usage. Traditional applications, such as underwater monitoring, surveillance, discovery, exploration, and coastal protection, are becoming increasingly sophisticated and produce more and more complex data in need of reliable and swift delivery to collection points on the surface. Examples include pictures and video streams from underwater cameras, as well as data from sonars and other high data rate sensors [1], [2]. In order to sustain the increasing requirements of these applications in the face of adverse and variable channel conditions, a recent trend is that of allowing sensor nodes to communicate through multiple devices (*multi-modal communications*). This best responds to the challenges of underwater communications, which are beset by extreme short-term channel variability and by environmental noise at different frequencies. Switching among multiple devices allows nodes to quickly adapt to these variations, changing frequencies, and therefore power levels, bitrates, ranges, etc., online, avoiding noise and other sudden impairments. For instance, endowing a node with acoustic and optical modems provides a flexible ways to combine long range, low bitrate, robust communication capabilities with short range, high bit rate data transfer. Typical applications benefiting from this multi-modal setting include networking with Autonomous Underwater Vehicles (AUVs) visiting sensors to retrieve data optically at high rate and coordinating with nodes and other AUVs over long range acoustic links [3], [4], [5], [6], [7], [8]. Multiple acoustic modems also provide great adaptability to channel variability and noise. In fact, acoustic modems with widely different characteristics are already available whose frequencies are centered from 24 kHz [9] to 100 kHz [10], up to 160 kHz [11]. These devices obtain bandwidth, bitrates and ranges that are at least one order of magnitude different. For example, communicating on the 24 kHz band allows nodes to transmit data at 4 kb/s to receivers up to 2 km away [9]. Higher data rate modems, transmitting in the 100 kHz band at up to 28 kb/s, reliably deliver data to nodes no more than 100 m away [10]. This wide variety of technologies promptly at disposal of a node provides the per-link reliability unavailable so far to underwater communication.

In this paper we aim at demonstrating how multi-modal communications can be cleverly exploited for reliable and low latency underwater networking, i.e., how the *link* reliability afforded to a node by multiple devices can be extended to *network-wide routes*. In particular, we explore how nodes can acquire knowledge on the quality of the links to neighboring nodes through each of their communication devices, and how this knowledge can be used for selecting reliable multi-link, multi-modal routes to the data final destination. We define a *model-based reinforcement learning* framework through which senders are able to select the best forwarding relay for their data packets jointly with the best communication device to reach that relay. We name the resulting forwarding method *MARLIN*, for *Multi-modAl Reinforcement Learning-based RoutINg*. MARLIN is flexibly defined to consider recent channel quality over each communication device, as well as information on routing reliability and delivery times from neighboring nodes, thus addressing network wide performance via local information exchange. It can also be configured to support multiple soft Quality-of-Service (QoS) classes,

through which applications can seek reliable routes to their data final destination, privileging packet delivery ratio, or routes that provide faster packet delivery at the expense of moderate packet loss.

The performance of MARLIN is evaluated through simulations with SUNSET, an extension to the simulator ns-2 that models a wide variety of details of the underwater channel and environment realistically [12]. We consider compelling underwater scenarios, with different network size, varying traffic and data packet sizes, which represent the variety of settings suitable to a large number of applications. Nodes are endowed with two underwater acoustic modems with very different characteristics. We investigate metrics such as the packet delivery ratio (PDR), the end-to-end latency, and the energy spent to deliver data. We also consider protocol fairness and the overhead due to data packet collisions. In these settings we compare MARLIN to the following state-of-the-art protocols: (i) CARP, a cross-layer solution designed to be reliable, channel aware and energy efficient [13], and (ii) QELAR, a machine learning-based protocol designed for minimizing and balancing node energy consumption [14]. We choose CARP and QELAR as they have been shown to outperform previous solutions for underwater routing and machine learning-based routing, respectively. Finally, to demonstrate that multi-modality alone does not lead to superior performance, i.e., to validate the need of our smart framework for route determination, we benchmark the performance of MARLIN to that of MFlood, a multi-modal version of the common flooding protocol enhanced to reduce collisions and increase reliability, where a packet is broadcast on one of the modems selected randomly.

Our results show that MARLIN obtains remarkable packet delivery ratio, outperforming the other three protocols in all scenarios. In particular, in challenging settings—networks with 40 nodes, high traffic and large packets—MARLIN obtains a PDR that is twice as much that of the second best performing protocol. It also achieves remarkable fairness, delivering packets from all nodes in the networks, even those further from the destination. Because of the clever definition of the cost optimization function of its core framework, despite delivering a higher number of packets MARLIN is the fastest of all considered protocols, with improvements over the second fastest of up to 58%. Finally, energy consumption is also well kept at bay, showing performance at par with that of CARP, which uses channel reservation and save energy on packet retransmission. The results in this paper confirm that the trend of using multi-modal communication for link reliability extends to network-wide performance provided that route selection is driven by a smart choice of best relays and communication devices. This makes MARLIN a solution for future underwater networking, achieving performance levels needed by key underwater applications.

The rest of the paper is organized as follows. Section II introduces notation and preliminaries on model-based reinforcement learning, the core of MARLIN routing. Section III defines MARLIN in details. Section IV reports results from our comparative performance evaluation of MARLIN, CARP, QELAR and MFlood. In Section V we survey previous works on multi-modal communications and on underwater reinforcement learning-based routing. Section VI concludes the paper.

II. Preliminaries and Notation

This section is intended to introduce notation and concepts that are preliminary to the protocol description. We also provide a brief introduction to model-based reinforcement learning, whose methods constitute the core of MARLIN.

Multi-modal scenario. We consider a multi-hop underwater wireless sensor network (UWSN) made up of N static nodes whose sensors produce data packets p that need to be routed to the network data collection point (the *sink*). Each node is equipped with multiple wireless communication devices (modems) operating on different frequencies, at different bandwidths, obtaining different bitrates, and with various communication ranges and power consumptions. Nodes are generically indicated as i and j. A specific modem is denoted by m, and $\mathcal{M}$ indicates the set of all modems available at each node. With $\mathcal{M}_{idle}$ we indicate the subset of the modems of a node that are idle at a certain point in time, i.e., that are ready to be used for transmission.

Quality-of-Service classes. MARLIN can be configured to support different QoS classes. In particular, in this work we consider a *reliability class* r, and a *low latency class* l. If it is configured to support class r, MARLIN seeks to deliver the highest number of packets to the sink. Otherwise, if it is configured to support class l, it does its best to be fast, at the expense of tolerating some packet loss.

Use of implicit ACKs. In order to reduce overhead—and latency, and energy consumption—each packet is acknowledged implicitly, leveraging the broadcast nature of the wireless channel. Specifically, after transmitting a packet, the sender starts listening to the channel on any of its modems. If it overhears the packet being retransmitted by the chosen relay within a given time, it considers the packet transmitted successfully. If it does not, the packet is considered lost. (Only the sink sends explicit ACKs back to its senders, as it does not forward the packet further). The node behavior after packet loss depends on the QoS class of the packet that has been transmitted, as described in details below.

A brief primer on reinforcement learning. Reinforcement learning concerns how some *agents* take *actions* in a given environment so as to optimize some notion of cumulative cost [15]. To this purpose, agents *learn* and optimize *policies* online through direct experience rather than compute them a priori. Given the set $\mathcal{S}$ of possible states of an agent, and the set $A(s)$ of the actions available at each state, a policy is a function π that associates each state $s \in \mathcal{S}$ with the action $a \in A(s)$ that the agent should take towards cost minimization. In the context of our work, agents correspond to underwater nodes handling packets, while the policy corresponds to the forwarding strategy.

In order for an agent to establish how good it is to be in a given state, a value function $V^{\pi}(s)$ is defined as the expected

infinite-horizon discounted cost starting from s as initial state and using a given policy π as follows:

$$V^{\pi}(s) = E_s^{\pi}\left\{\sum_{t=0}^{\infty}\gamma^t c(s_t, a_t)\middle| s_0 = s\right\}, \quad (1)$$

where $0 \leq \gamma < 1$ is the discount factor, s_t and a_t are the system state and the action taken at time t, respectively, and $c(s_t, a_t)$ is the expected cost associated to state s_t and decision a_t. For each state $s \in \mathcal{S}$, the optimal policy π^* minimizing the value functions satisfies the Bellman optimality equation:

$$V^{\pi^*}(s) = \min_{a \in A(s)}\left\{c(s,a) + \gamma\sum_{s' \in \mathcal{S}} P^a_{s\to s'} V^{\pi^*}(s')\right\}, \quad (2)$$

where $P^a_{s\to s'}$ represents the transition probability from state s to state s' after action a has been taken. Equations (2) highlight that the policy π^* that minimizes the cost depends on the immediate cost of taking the action a from state s and on the expected discounted cost from the next state s' onward.

In order to measure the costs of taking different actions a from state s to different states s' we define the function Q as the expected infinite horizon discounted cost of taking an action a in state s and then following the policy π:

$$Q^{\pi}(s,a) = c(s,a) + \gamma\sum_{s' \in \mathcal{S}} P^a_{s\to s'} V^{\pi}(s'),\ \forall s \in \mathcal{S}. \quad (3)$$

For each state $s \in \mathcal{S}$ the Bellman equation and the function Q allow us to greedily compute the optimal policy as $\pi^*(s) = \arg\min_{a\in A(s)} Q^{\pi^*}(s,a)$.

Solving the Belman equations (2) depends on knowing the cost $c(s,a)$ and the transition probabilities $P^a_{s\to s'}$. In scenarios where these parameters are not known a priori, models can be provided for their estimation, so that functions V and Q can be learned online, by interacting with the agent environment. This interaction usually takes the form of *exploiting* the knowledge acquired so far by the agent, and of *exploring* the agent environment to gain new knowledge. Learning techniques following this methodology are called *model-based* [15].

III. The MARLIN Protocol

In this section we define MARLIN. The protocol is introduced as configured for QoS class r. Section III-C describes the details of MARLIN when configured for QoS class l.

The operations of MARLIN are quite simple: When a node i has a packet p to forward, it chooses the most suitable relay among its neighbors together with a suitable modem. Then node i transmits p to that relay using the chosen modem. After transmission node i awaits to overhear the forwarding of p by the relay (implicit ACK). If that does not happen within a predefined time, the node retransmits p till success, for at most $K-1$ additional times, $K \geq 1$. If all retransmission attempts fail, the packet is dropped.

This simplicity of operations is supported by quite a sophisticated reinforcement learning-based framework for the selection of the relay and of the modem. In this framework, each node i acts as an agent that, for each packet p, determines the best among a set of forwarding decisions (action a, i.e., a relay and a modem). The optimal routing policy π^* is determined by learning the function $V_i^{\pi^*}(s) = \min_a Q_i^{\pi^*}(s,a)$ for each state s of node i handling p. This function V plays the role of routing table providing the forwarding decision a that minimizes the cost of sending p to the sink. In the following we describe the details of how function $V_i(s)$ is computed over time by learning the values of $Q_i(s,a)$ following a model-based reinforcement learning approach.[1] We start by defining the model for routing packets in MARLIN, i.e., the state space, the actions, the state transition dynamics and the cost function. Then we describe how nodes in MARLIN learn how to route by determining optimal forwarding decisions.

A. A model for forwarding packets

States. A node i handling a packet p is associated with a state s defined as the number of times $k \leq K-1$ that it has transmitted p unsuccessfully. The state space $\mathcal{S}$ is therefore the following: $\mathcal{S} = \{0, \ldots, K-1\} \cup \{rcv, drop\}$, where rcv identifies successful packet transmission, and $drop$ identifies the case when the maximum number K of transmissions has been exceeded and the packet is discarded.

Actions. Actions concern forwarding decisions, i.e., the joint selection of a relay among the sender neighbors and of the modem to that relay. Let us denote with $\mathcal{N}_i^m$ the set of nodes that a node i can reach using modem m. Using different modems results in different sets, although a neighbor can be reached by multiple modems.

For each node i, state s and set of modems $M \subseteq \mathcal{M}$, the set $A_i^M(s)$ of available actions is:

$$A_i^M(s) = \{a = \langle j, m\rangle \mid m \in M,\ j \in \mathcal{N}_i^m\}, \quad (4)$$

where $a = \langle j, m\rangle$ is the action of forwarding a packet to neighbor j using modem m. Since no action can take place when $s \in \{rcv, drop\}$, it is $A_i^M(rcv) = A_i^M(drop) = \emptyset$.

Transitions. The transition from one state to another depends on the current state s and on the performed action $a = \langle j, m\rangle$. Let us denote with $P_{i,j}^m$ the probability of correct packet transmission on the link from node i to node j using modem m. When the transmission of p succeeds after k unsuccessful attempts, node i transitions from state $s = k$ to state $s' = rcv$. The transition probability is the following:

$$P^{\langle j,m\rangle}_{i,s\to rcv} = \begin{cases} P^m_{i,j} P^{m'}_{j,i} & \text{if } 0 \leq k < K-1 \\ P^m_{i,j} & \text{if } k = K-1. \end{cases} \quad (5)$$

When p can be (re)transmitted, i.e, when the number k of retransmission is $< K-1$, successful transmission depends on the following probabilities: (i) The probability $P_{i,j}^m$ that the packet is received by node j on modem m. This probability is computed by node j and broadcast in the header of its packets. (ii) The probability $P_{j,i}^{m'}$ that the packet p, forwarded by node j on best modem m', is successfully overheard by node i. This

[1] From now on we will omit the superscript π^* for ease of notation.

probability is computed by node i based on overhearing node j transmissions on modem m'.[2] When the number of possible retransmissions has reached its last value K, there is no need for i to listen for an ACK, and the state transition depends only on the link from i to j.

When the transmission of p fails, we have two possible transitions. If $k < K-1$ we just increase the number of retransmissions, and the next state is $s' = k+1$. Otherwise, if $k = K-1$, the packet p is dropped and the next state is $s' = drop$. In both cases, the transition probability can be defined simply as $P^{\langle j,m\rangle}_{i,s\to s'} = 1 - P^{\langle j,m\rangle}_{i,s\to rcv}$.

Costs. In a reinforcement learning approach, the inner logic of a protocol resides in the structure of the cost function c_i to be optimized. MARLIN focuses on minimizing the *network-wide time needed by node i to deliver packet p to the sink*, i.e., to its final destination. In order to express the whole routing time, we associate each state-action pair (s,a) with a cost reflective of the time needed to forward the packet by one hop, to a selected relay, and of the time needed to forward it from that relay to the sink. Equally important, we increase reliability by associating a high penalty time to transitions to the *drop* state. As we seek to minimize time, this discourages packet loss. More formally:

$$c_i(s,a) = \begin{cases} t_i(s,a) + d_i(s,a) & \text{if } 0 \le k < K-1 \\ t_i(s,a) + d_i(s,a) + l_i(s,a) & \text{if } k = K-1, \end{cases} \tag{6}$$

where $t_i(s,a)$ is the time needed for p to be delivered to neighbor j, $d_i(s,a)$ is the time from node j to the sink, and $l_i(s,a)$ is a large value that we use to encourage nodes to deliver a packet (this penalty make sense only when the packet has no other changes to be retransmitted, i.e., $k = K-1$). The cost $t_i(s,a)$ is defined as follows:

$$t_i(s,a) = t_m + p_{i,j} + \begin{cases} w_i & k = 0 \\ b_i & \text{otherwise,} \end{cases} \tag{7}$$

where t_m is the time needed to transmit p on modem m, $p_{i,j}$ is the propagation time between the two nodes, w_i is the average time spent by packet p in the queue of node i (before the first transmission of the packet), and b_i is the time spent waiting for the implicit ACK (subsequent retransmissions). The $d_i(s,a)$ component of the cost is given by:

$$d_i(s,a) = V_j(0) P^m_{i,j}, \tag{8}$$

where $V_j(0)$ is the value of the function V associated to the state $s = 0$ of relay node j. This cost is, by definition, a measure of the minimum time needed to reach the sink starting from node j. (It is available to node i as it is broadcast by node j in the header of its packets.) The cost $V_j(0)$ is multiplied by the probability $P^m_{i,j}$ as node j will forward the packet only in case it correctly receives it from node i. Finally, in case a packet has been unsuccessfully retransmitted for $k = K-1$ times, we associate to the action $a = \langle j,m\rangle$ of the last retransmission the penalty time $l_i(s,a)$ aimed at discouraging node i to drop the packet. We can think of dropped packets as packets that are delivered to the sink arbitrarily late in time. As such, the cost $l_i(s,a)$ associated to "delivering the packet that late" is defined as:

$$l_i(s,a) = L(1 - P^m_{i,j}), \tag{9}$$

where $(1 - P^m_{i,j})$ is the probability of dropping the packet, and L is set to a value greater than the highest cost of delivering the packet to the sink through any of the neighbors of node i (i.e., $L > \max_{j \in \bigcup_m \mathcal{N}^m_i} V_j(0)$). In other words, as node i approaches the maximum number K of transmission attempts, its actions tend to favor the reliability of the transmission to the next hop.

B. Learning to route

To compute optimal forwarding decisions, every time a packet p is ready to be transmitted, node i executes an algorithm for learning the value of the function Q_i and update its function V_i. Based on this value node i determines the optimal forwarding action $a = \langle j,m\rangle$ for p, i.e., the best relay j and the best modem m to reach it. Each node starts with no knowledge of its surrounding environment. Interacting with its neighbors, it iteratively acquires and updates its knowledge over time. In particular, function Q_i is approximated relying on current estimations of the transition probabilities $P^a_{i,s\to s'}$, and on the estimated value of the functions $V_j(0)$ from neighboring nodes j, needed to estimate the cost $c_i(s,a)$.[3] Algorithm COMPUTER&M describes the learning process of node i and the corresponding determination of the best *relay* and *modem* (R&M) for packet p.

Algorithm COMPUTER&M(k, $\mathcal{M}_{idle}$)

1: **for all** $(s \in \mathcal{S})$ **do**
2: **for all** $a \in A^{\mathcal{M}}_i(s)$ **do**
3: $Q_i(s,a) = c_i(s,a) + \gamma \sum_{s' \in \mathcal{S}} P^a_{i,s\to s'} V_i(s')$
4: **end for**
5: $V_i(s) = \min_{a \in A^{\mathcal{M}}_i(s)} Q_i(s,a)$ *#Update*
6: **end for**
7: $\tau = random\_number(0,1)$
8: **if** $\tau < 1 - \epsilon$ **then** *#Exploitation*
9: $\langle j,m\rangle = \arg\min_{a \in A^{\mathcal{M}_{idle}(k)}_i} Q_i(k,a)$
10: **else** *#Exploration*
11: $m =$ random modem in $\mathcal{M}_{idle}$
12: $\langle j,m\rangle = \arg\min_{a \in A^{\{m\}(k)}_i} Q_i(k,a)$
13: **end if**
14: **return** a = $\langle j,m\rangle$ *#Forwarding decision*

The algorithm takes as input the current state k and the set of idle modems $\mathcal{M}_{idle}$. When packet p is ready for transmission, node i updates the model and computes the new value function (line 1 to 6). Once the model has been updated, a forwarding action can be selected. To balance between exploitation of the acquired knowledge and exploration of new one, we use the ϵ-greedy heuristic, because of its well-known robustness and

[2] The determination of best modems and probabilities is described in details in Section III-B.

[3] From now on, all values of the transition probabilities, and of functions V, Q and c are to be intended estimates changing over time.

effectiveness [15]. Specifically, each time, the best neighbor on the best idle modem is selected with probability $1-\epsilon$ (line 9), exploiting the knowledge just updated. Conversely, with probability ϵ we explore new solutions by selecting a random modem among those available (line 11), and we forward the packet to the best relay we can reach using that modem (line 12). Even if exploration may produce a suboptimal choice of modem and relay, the broadcast nature of the transmission to this relay allows nodes to acquire key statistics about their neighbors.

The execution of Algorithm COMPUTER&M relies on the knowledge of the transition probabilities $P^a_{i,s\to s'}$ and on the packet forwarding cost $c_i(s,a)$, which in turn depends on the value function $V_j(0)$ of each neighbor j of node i. In the rest of this section we describe how these probabilities and function values are determined.

The estimation of the transition probabilities is based on the estimation of the link probabilities $P^m_{i,j}$ (Equation (5)). Nodes estimate link quality upon receiving a packet. In particular, a receiver j keeps count of the number of packets $n^m_{i,j}$ received from each neighbor i on modem m, regardless of whether node j is the packet intended destination. The incoming link probability is estimated as $P^m_{i,j} = n^m_{i,j}/n^m_i$, where n^m_i is the total number of packet sent by node i, an information that node i broadcasts in the header of its packets. These estimates are then broadcast by node j into its packet headers, to be overheard by its neighbors. In order to keep track of the varying link conditions, the counts n^m_i and $n^m_{i,j}$ are computed over a sliding window. If node i fails to overhear transmissions from a neighbor j within a given time it automatically update its own link transmission probability. In particular, node i "degrades" $P^m_{i,j}$ to $(n^m_i/(n^m_i+1))P^m_{i,j}$. If node i does not hear packets from node j for a given time, it removes node j from the list of its neighbors until node j is heard again. The determination of $c_i(s,a)$ is based on information available locally at node i and on values broadcast by node j in the header of its packets.

C. Configuring MARLIN for QoS class l

MARLIN can be configured to support faster packet forwarding to the sink. This is obtained by avoiding multiple retransmissions of a packet, i.e., by setting the retransmission threshold K to 1, and by adopting a collaborative transmission strategy. More precisely, when a sender transmits a packet p, it selects a neighbor (the *main relay*) and a modem, as before, but it also adds to the packet header a *prioritized list of backup relays*. These are the nodes that, upon realizing that the main relay is not forwarding p, will forward it themselves. Upon receiving the packet, the backup relays set a backup timer and store the packet in their queues. The timer is set to a time that is inversely proportional to the node position in the prioritized list of backup relays: The higher its position, the shorter the time. When the timer goes off, the node checks if it has overheard the packet transmission by higher priority nodes. If so, it discards the packet; otherwise it transmits it. For example, the first backup relay forwards p only if it does not overhear its transmission by the main relay; the second backup relay transmits p only if it does not overhear its transmission from either the main relay or the first backup node, and so on. By not retransmitting a packet for which there is no implicit ACK, the original sender keeps transmitting other packets, whose queuing delay is thus shorter. This is a key feature of MARLIN, enabling overall faster packet delivery to the sink.

The learning Algorithm COMPUTER&M described above can be easily extended to output also the prioritized list of backup relays. Specifically, let us assume that Algorithm COMPUTER&M outputs node j as the main relay to be reached using modem m. Let us also assume that node j uses modem m' to forward the packet p. The first backup relay h is selected by node i using the following rule:

$$\langle h, m\rangle = \underset{a\in A_i^{\{m\}}(0)\setminus\{\langle j,m\rangle\}}{\arg\min} \frac{Q_i(0,a)}{P^{m'}_{j,h}}. \tag{10}$$

This rule aims at selecting a node that is not only a good forwarder (low $Q_i(0,a)$) but also a node that is "well connected" to the main relay j, i.e., with high chances of overhearing j forwarding p (high $P^{m'}_{j,h}$). In so doing, node i tries to avoid redundant retransmissions that would increase network traffic and waste energy. Applying rule (10) with h in place of j produces the second best backup relay, and so on.

The length of the list is set to a number $\ell_i \le |\mathcal{N}^m_i|$, which may vary dynamically, depending on the network load. In fact, we keep adding backup relays to the list until either the probability that none of them receives the packet is lower than a given threshold P_{lost} or the size of the list ℓ_i is reached.

IV. PERFORMANCE EVALUATION

We demonstrate the effectiveness of our reinforcement learning-based approach to multi-modality through a simulation-based comparative performance evaluation of our protocol and three other protocols for underwater routing. In particular, we compare the performance of MARLIN, configured for reliability (MARLIN-r in the following) and for low latencies (MARLIN-l), to that of the following protocols: (i) CARP, a state-of-the-art cross-layer solution designed to be reliable, channel aware and energy efficient. CARP is characterized by a channel reservation phase through control packets (named PING and PONG) that also carry routing information [13]. (ii) QELAR, a machine learning-based protocol designed for minimizing and balancing node energy consumption [14]. (iii) MFlood, a multi-modal enhanced version of the common flooding protocol designed to reduce collisions and increase reliability, where a packet is broadcast on an idle modem selected randomly. All routing protocols have been implemented in SUNSET [12], an extension of ns-2 connected to the Bellhop ray tracing tool via the WOSS interface [16]. The environmental data input to Bellhop refer to an area located in the Norwegian fjord off the coast of Trondheim, with the coordinate $(0,0,0)$ of the surface located at $63°, 29', 1.0752''N$ and $10°, 32', 46.6728''E$. Sound speed profiles, bathymetry profiles and information on the type of bottom sediments of the selected area are obtained from the

World Ocean Database, from the General Bathymetric Chart of the Oceans (GEBCO), and from the National Geophysical Data Center's Deck41 data-base, respectively [17].

A. Simulation scenarios and settings

We consider three UWSNs with 6, 20 and 40 nodes randomly and uniformly placed in rectangular regions with surface of 1 km^2, 2 km^2 and 4 km^2, respectively. This allows us to investigate networks with size ranging from that of current deployments (6 nodes) to that of larger (20) and desirable (40) networks. In all scenarios nodes are deployed at different depths, ranging from 10 to 240 m, while the sink is located at one of the corners of the deployment area, 10 m below surface. Topology construction ensures that each node has at least one route to the sink.

We consider underwater nodes equipped with two acoustic modems with different characteristics. The first modem carrier frequency is set to 25.6 kHz for a bandwidth of 4 kHz, resulting in a bitrate of 4000 b/s. The second modem carrier frequency is higher, 63 kHz, for a bandwidth of 30 kHz and a bitrate of 9000 b/s. We assume a BPSK modulation for both modems. For the selected value of the bandwidth and of the carrier frequency the transmission power of the two modems is set to 2.8 W and 5.5 W. Their reception power is set to 0.5 W. These values are consistent with those of commercial modems by Teledyne Benthos [9] and Evologics [11].

Traffic is generated according to a Poisson process with aggregate (network-wide) rate of λ packets per second, where λ ranges in $\{0.02, 0.09, 0.16\}$, corresponding to low, medium and high traffic, respectively. Once a packet is generated, it is associated with a source selected randomly among all nodes (but the sink). The destination of all packets is the sink. The data packet payload size (in bytes) varies in the set $\{250, 500, 1000\}$. The total size of a data packet is given by the selected payloads plus the headers added by the different layers. The physical header overhead changes according to the data rate but is dominated by a 10 ms synchronization preamble. At the MAC layer, the header size depends on the protocol. QELAR and MFlood use CSMA, whose header contains the sender and the destination addresses, and the packet type. Its length is 3 B. QELAR also needs 6 B extra for information on the residual energy and for the state space of the node. Being a cross layer protocol, CARP implements its own MAC, and the header of its MAC packets also carries routing information. As such, the size of its PING and PONG control packets is 10 B and 6 B, respectively. Its ACK and HELLO packets are 6 B long. The CARP MAC data packet header is 4 B long. Finally, as MARLIN carries a number of information in the packet header, including the value function, $P_{i,j}$ estimates, list of backup relays, etc., its size varies with the network size. In our implementation the MARLIN header size were 7 B, 15 B and 30 B, depending on the network size. In our experiment, we have set the parameter ϵ of Algorithm COMPUTER&M to 0.1, as typical [15]. The number of retransmissions K used by MARLIN-r, QELAR and CARP is set to to 4 (low traffic), 3 (medium traffic) and to 2 (high traffic). In the implementation of MARLIN-l, the value of K is set to 1, the maximum length ℓ_i of the prioritized list of backup relays of node i is set to 4 (low traffic), 3 (medium traffic) and to 2 (high traffic), and P_{lost} is set to 0.05.

B. Simulation metrics

Routing performance is assessed through the investigation of the following metrics.

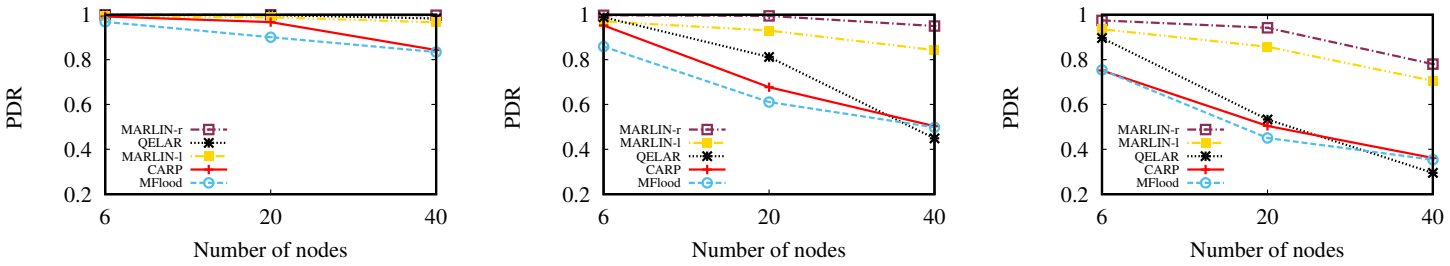
- *Packet delivery ratio* (PDR), defined as the fraction of packets correctly received by the sink with respect to those generated by the nodes.
- *End-to-end latency*, defined as the time between packet generation and the time of its correct delivery to the sink.
- *Energy per bit*, defined as the energy consumed by the network to correctly deliver a bit of data to the sink.

C. Performance results

In this section we illustrate the results from simulations in the described scenarios. For QELAR and CARP we show figures obtained by using the modem that produces their best performance. We consider a packet size of 1000 B, as those for a packet size of 250 B and of 500 B show similar trends and no further insights. All results are obtained by averaging over data from 150 simulation runs, which achieves a statistical confidence of 95% within a 5% precision.

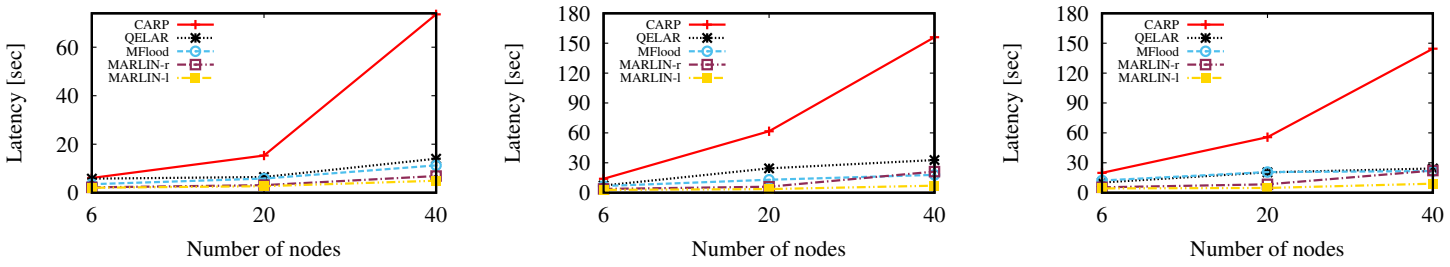
1) Packet delivery ratio: Fig. 1 shows the PDR for different network sizes and traffic rates. Independently of the QoS class, MARLIN obtains the highest PDR in all scenarios. Its advantage over the other protocols increases with the traffic load, indicating higher scalability. At the highest traffic rate, MARLIN delivers even more than twice the packets delivered by all other protocols (Fig. 1c). The reason does not stem solely from the use of multiple modems. In fact, the performance of MFlood, which is well below that of MARLIN in all scenarios, shows clearly that just spreading packets randomly onto multiple devices is no game changer, and obtains results that are only slightly better of common flooding over a single modem (not shown here). QELAR obtains its best performance by using different modems in different scenarios. For instance, at low traffic, its best PDR is obtained using one modem, while at medium and high traffic QELAR delivers more packets using the other one. This suggests that there is no winning choice for all scenarios. The fact that MARLIN is able to choose the best modem on a per-link basis, switching to the device that provides the best forwarding conditions, obtains values of PDR that are up to 160% higher than those produced by QELAR. Finally, we observe that the PDR performance of CARP suffers from difficulties in gaining channel access, especially at high traffic. This results in a PDR that is even 54% worse than that of MARLIN.

2) End-to-end latency: Fig. 2 shows results concerning the packet end-to-end latency. In spite of a packet delivery ratio significantly higher than that of other protocols, MARLIN exhibits always the lowest latencies. This is because the routing model explicitly take latency and robustness into account in its cost function (Section III-B). This clearly points out



(a) Low traffic: $\lambda = 0.02$ (b) Medium traffic: $\lambda = 0.09$ (c) High traffic: $\lambda = 0.16$

Fig. 1: Packet Delivery Ratio.



(a) Low traffic: $\lambda = 0.02$ (b) Medium traffic: $\lambda = 0.09$ (c) High traffic: $\lambda = 0.16$

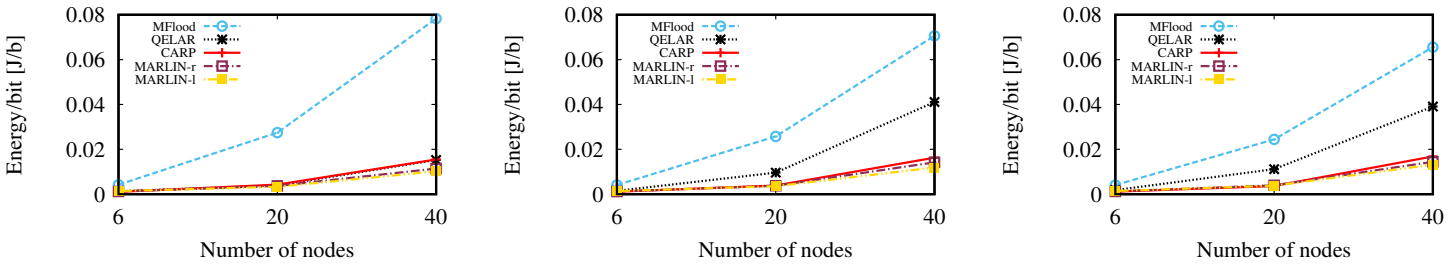
Fig. 2: End-to-end latency.

the effectiveness of MARLIN in forwarding packets on the best available routes. Not surprisingly, CARP experiences the highest latencies because of the handshake phase needed prior to packet transmission. The latency gap between our protocol and the others decreases with traffic as MARLIN is able to deliver much more packets, and from the furthest nodes. The comparison between the performance of MARLIN-l and MARLIN-r shows the effectiveness of the design of MARLIN when configured to obtain lower latencies (Section III-C). MARLIN-l delivers packets up to 60% faster than MARLIN-r, sacrificing a mere 10% drop of the PDR.

3) Energy per bit: Fig. 3 concerns results on the energy needed to deliver a bit of data. The worst performance is shown by MFlood, which spends more than any other protocols independently of the scenario. By choosing relays and modems smartly, MARLIN always exhibits excellent performance. For instance, it consumes up to 71% less energy per bit than QELAR, as it delivers packets by using shorter routes and experiencing a lower number of retransmissions. CARP is as efficient as MARLIN in routing packets to the sink because of its channel reservation phase, which results in few packet collisions. However, the energy spent for control packets raises its energy per bit up to 36% more than that spent by MARLIN.

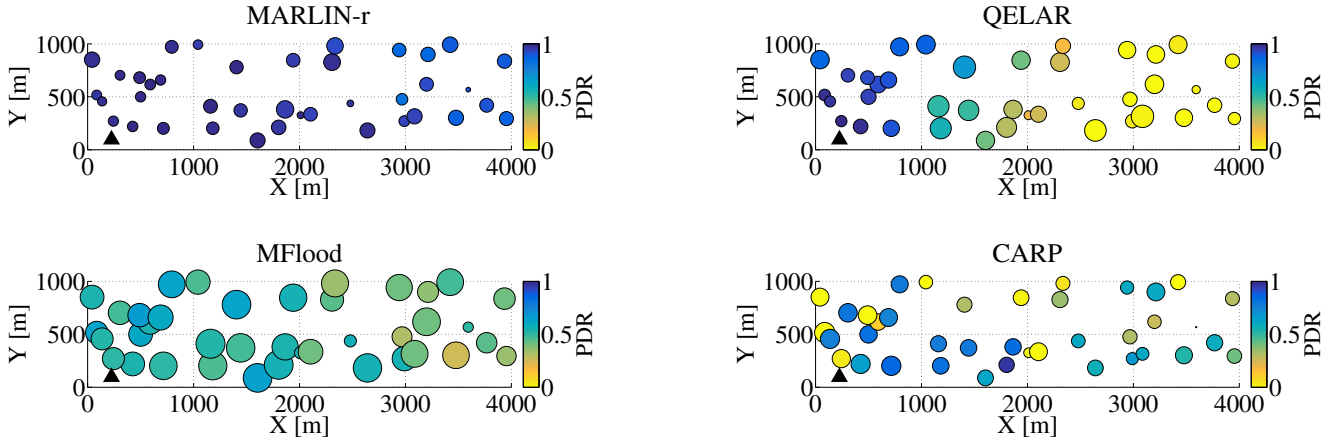
We end this section by providing further evidence of the effectiveness of the learned exploitation of multi-modality. Fig. 4 shows the four topologies of a network with 40 nodes running the protocols MARLIN-r, MFlood, QELAR and CARP. The sink is depicted as a black triangle. Each other node is depicted as a circle whose radius is proportional to the number of packet collisions experienced at that node (in this pictures, smaller is better). The color of the node indicates its PDR i.e., the fraction of its packets that are successfully delivered to the sink: The darker the color the higher the PDR.

We notice that the nodes running MARLIN-r are all fairly small, and that all nodes, including those far away from the sink, are colored in the shades of the darkest color (Fig. 4a, top). In other words, MARLIN-r is always successful in selecting relays and modems that obtain high PDR, even from node further away from the sink, suffering from few collisions. Similarly to MARLIN, MFlood also uses multiple modems. However, the random selection of the modem does not yield high PDR and the high number of packets transmitted incurs many collisions, with detrimental effects also on latency and energy consumption. As depicted in Fig. 4a (bottom), nodes running MFlood are quite large, and their color is on the lighter side throughout the network. The figure concerning QELAR (Fig. 4b, top) reveals an evident problem with fairness, as the protocol is able to deliver very few packets from nodes away from the sink, incurring a high number of collisions. The small size of the nodes confirms that collisions are not a problem for CARP, because of its channel reservation mechanism (Fig. 4b, bottom). However, nodes experience problems in reserving the channel, independently of their distance from the sink. In fact, because of the sink-induced traffic funnel effect, channel contention is higher in the region of the sink, which results in some nodes with lower PDR even in that region.



(a) Low traffic: $\lambda = 0.02$ (b) Medium traffic: $\lambda = 0.09$ (c) High traffic: $\lambda = 0.16$

Fig. 3: Energy per bit.



(a) MARLIN-r and MFlood (b) QELAR and CARP

Fig. 4: A joint snapshot of the PDR and collisions per node in networks with 40 nodes and medium traffic.

V. Related Work

Multi-modal communications have emerged as a means to enhance UWSN reliability and performance in a variety of scenarios. Existing works concern combining acoustic communication for the long haul, more robust, low data rate exchanges, with short-range, high data rate optical packet transfer [3], [4], [5], [6], [7], [8]. While these works show that using multiple communicating devices overcomes engaging challenges of underwater data transfer, they do not concern data routing, as we do in this paper. The only previous work on multi-modal routing is the MUlti-level Routing protocol for Acoustic-Optical UWSNs (MURAO), by Hu and Fei [6]. MURAO concerns partitioning the network nodes into two layers. Lower layer nodes are responsible for multi-hop data forwarding over optical channels. Nodes in the upper layer use long range/low bandwidth acoustic communication to coordinate the routing of the lower level nodes. Actual data routing within the two layers is performed by QELAR, a machine learning-based routing protocol for single-modem UWSNs (described below). MURAO requires nodes to be deployed densely enough to obtain a connected topology over the optical links. Given the short range of these links, MURAO can be costly and even impracticable for applications requiring coverage of large areas.

While multi-modal routing is still an unexplored topic, routing protocols for UWSNs with single-mode acoustic modems have been proposed for over a decade now, and include remarkably effective solutions, including [13], [18], [19], [20], [21], [22] and those surveyed by Ayaz et al. [23] and by Li et al. [24]. A solution that stands out in terms of enhanced performance is the Channel-aware Routing Protocol (CARP) by Basagni et al., which exploits link quality information for data forwarding [13]. Nodes are selected as relays based on their link quality, hop count and residual energy. CARP utilizes a channel reservation mechanism à la RTS/CTS for channel access and for selecting packet relays (cross layer design). For this reason, while achieving reliability and suffering from few packet collisions, it incurs remarkable latencies.

The use of reinforcement learning techniques for UWSN routing has been explored in [6], [14], [25], [26]. Solutions presented in [25], [26] concern the specific scenario of networks with intermittent connectivity. The QELAR protocol by Hu and Fei has been introduced for routing in scenarios similar to those considered in this paper. QELAR is based on

a model-based Q-learning approach aimed at maximizing the residual energy among nodes [14]. The learning cost function accounts for the residual energy of each node as well as for the energy distribution among neighboring nodes, and relays are chosen depending on the energy they can save. This makes QELAR a solution that compares well with previous protocols, especially in terms of network lifetime. MURAO, cited earlier in this section, uses QELAR as intra-layer routing, both optical and acoustic [6].

VI. Conclusions

This paper concerns UWSNs with nodes with multi-modal communication capabilities. We present a reinforcement learning-based framework that instructs senders to select the best forwarding relay for its data and the best communication device to reach that relay. The resulting forwarding method, named MARLIN, can be configured to seek reliable routes to their final destination, or to provide faster packet delivery. Through a SUNSET-based performance study we show that MARLIN always outperforms state-of-the-art underwater forwarding protocols by delivering more packets (up to twice as much than anybody else), remarkably reducing packet collisions, showing fairness throughout the network, maintaining latencies that are up to 58% lower that those of other solutions, while saving considerable energy. Our results clearly show that the smart use of multi-modal communications takes underwater networking to levels of reliability and low latencies long demanded by the majority of key underwater applications.

Acknowledgments

The work was performed under the sponsorship of the EU FP 7 ICT project SUNRISE "Sensing, monitoring and actuating on the UNderwater world through a federated Research InfraStructure Extending the Future Internet." Stefano Basagni was supported in part by grant NSF CNS 1428567 and through a GENI SAVI travel grant.

References

[1] F. S. Rende, A. D. Irving, A. Lagudi, F. Bruno, S. Scalise, P. Cappa, M. Montefalcone, T. Bacci, M. Penna, B. Trabucco, R. Di Mento, and A. M. Cicero, "Pilot application of 3D underwater imaging techniques for mapping posidonia oceanica (L.) delile meadows," *ISPRS—International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XL-5, no. W5, pp. 177–181, 2015.

[2] D. Scaradozzi, L. Sorbi, and F. Zoppini, "DiRAMa facilitates data gathering and analysis at sea," *Sea Technology Magazine*, vol. 55, no. 6, pp. 19–22, June 2014.

[3] F. Campagnaro, F. Guerra, P. Casari, R. Diamant, and M. Zorzi, "Implementation of a multi-modal acoustic-optic underwater network protocol stack," in *Proceedings of MTS/IEEE OCEANS 2016*, Shanghai, China, April 10–13 2016, pp. 1–6.

[4] F. Campagnaro, F. Favaro, F. Guerra, S. V. Calzado, M. Zorzi, and P. Casari, "Simulation of multimodal optical and acoustic communications in underwater networks," in *Proceedings of the MTS/IEEE OCEANS 2015*, Genova, Italy, May 18–21 2015, pp. 1–6.

[5] S. Basagni, L. Bölöni, P. Gjanci, C. Petrioli, C. A. Phillips, and D. Turgut, "Maximizing the value of sensed information in underwater wireless sensor networks via an autonomous underwater vehicle," in *Proceedings of IEEE Infocom 2014*, Toronto, Canada, April 27–May 2 2014, pp. 988–996.

[6] T. Hu and Y. Fei, "MURAO: A multi-level routing protocol for acoustic-optical hybrid underwater wireless sensor networks," in *Proceedings of SECON 2012*, Seul, Korea, June 18–21 2012, pp. 218–226.

[7] N. Farr, A. Bowen, J. Ware, C. Pontbriand, and M. Tivey, "An integrated, underwater optical/acoustic communications system," in *Proceedings of the MTS/IEEE OCEANS 2010*, Sydney, Australia, May 24–27 2010, pp. 1–6.

[8] N. Farr, J. Ware, C. Pontbriand, T. Hammar, and M. Tivey, "Optical communication system expands CORK seafloor observatory's bandwidth," in *Proceedings of the MTS/IEEE OCEANS 2010*, Seattle, WA, September 20–23 2010, pp. 1–6.

[9] "The Teledyne Benthos SMART product SM-975," http://teledynebenthos.com/product/smart_products/sm-975.

[10] E. Demirors, G. Sklivanitis, T. Melodia, S. N. Batalama, and D. A. Pados, "Software-defined underwater acoustic networks: Toward a high-rate real-time reconfigurable modem," *IEEE Communications Magazine*, vol. 53, no. 11, pp. 64–71, November 2015.

[11] Evologics, "Evologics S2C acoustic modems." [Online]. Available: https://www.evologics.de/files/DataSheets/EvoLogics_S2CR_Modems_a4_WEB.pdf

[12] C. Petrioli, R. Petroccia, and D. Spaccini, "SUNSET version 2.0: Enhanced framework for simulation, emulation and real-life testing of underwater wireless sensor networks," in *Proceedings of ACM WUWNet 2013*, Kaohsiung, Taiwan, November 11–13 2013, pp. 1–8.

[13] S. Basagni, C. Petrioli, R. Petroccia, and D. Spaccini, "CARP: A channel-aware routing protocol for underwater acoustic wireless networks," *Ad Hoc Networks*, vol. 34, pp. 92–104, November 27 2015.

[14] T. Hu and Y. Fei, "QELAR: A machine-learning-based adaptive routing protocol for energy-efficient and lifetime-extended underwater sensor networks," *IEEE Transactions on Mobile Computing*, vol. 9, no. 6, pp. 796–809, June 2010.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[16] F. Guerra, P. Casari, and M. Zorzi, "World Ocean Simulation System (WOSS): A simulation tool for underwater networks with realistic propagation modeling," in *Proceedings of ACM WUWNet 2009*, Berkeley, CA, November 3 2009, pp. 1–8.

[17] "WOD, GEBCO, and Deck41." [Online]. Available: http://www.nodc.noaa.gov/OC5/WOA05/pr_woa05.html,http://www.gebco.net,http://www.ngdc.noaa.gov/mgg/geology/deck41.html.

[18] Y. Noh, U. Lee, P. Wang, B. S. C. Choi, and M. Gerla, "VAPR: Void-aware pressure routing for underwater sensor networks," *IEEE Transactions on Mobile Computing*, vol. 12, no. 5, pp. 895–908, 2013.

[19] G. Toso, R. Masiero, P. Casari, O. Kebkal, M. Komar, and M. Zorzi, "Field experiments for dynamic source routing: S2C EvoLogics modems run the SUN protocol using the DESERT underwater libraries," in *Proceedings of MTS/IEEE OCEANS 2012*, 2012, pp. 1–10.

[20] X. Xiao, X. P. Ji, G. Yang, and Y. P. Cong, "LE-VBF: Lifetime-extended vector-based forwarding routing," in *Proceedings of CSSS 2012*, Nanjing, China, August 2012, pp. 1201–1203.

[21] D. Shin, D. Hwang, and D. Kim, "DFR: An efficient directional flooding-based routing protocol in underwater sensor networks," *Wireless Communications and Mobile Computing*, vol. 12, no. 17, pp. 1517–1527, December 2012.

[22] D. Pompili and I. F. Akyildiz, "A multimedia cross-layer protocol for underwater acoustic sensor networks," *IEEE Transactions on Wireless Communications*, vol. 9, no. 9, pp. 1536–1276, September 2010.

[23] M. Ayaz, I. Baig, A. Abdullah, and I. Faye, "A survey on routing techniques in underwater wireless sensor networks," *Journal of Network and Computer Applications*, vol. 34, no. 6, pp. 1908–1927, November 2011.

[24] N. Li, J.-F. Martnez, J. M. Meneses Chaus, and M. Eckert, "A survey on underwater acoustic sensor network routing protocols," *Sensors*, vol. 16, no. 3, pp. 1–28, March 22 2016.

[25] R. Plate and C. Wakayama, "Utilizing kinematics and selective sweeping in reinforcement learning-based routing algorithms for underwater networks," *Ad Hoc Networks*, vol. 34, pp. 105–120, 2015.

[26] T. Hu and Y. Fei, "An adaptive routing protocol based on connectivity prediction for underwater disruption tolerant networks," in *Proceedings of IEEE Globecom 2013*, Atlanta, GA, December 9–13 2013, pp. 65–71.