Underwater Acoustic Sensors Data Collection in the Robotic Vessels as-a-Service Project

Alberto Signori
Dept. of Information Engineering
University of Padova
Padova, Italy
signoria@dei.unipd.it

Filippo Campagnaro
Dept. of Information Engineering
University of Padova
Padova, Italy
campagn1@dei.unipd.it

Davide Zordan
Dept. of Information Engineering
University of Padova
Padova, Italy
zordanda@dei.unipd.it

Federico Favaro
Dept. of Information Engineering
University of Padova
Padova, Italy
fedefava86@gmail.com

Michele Zorzi
Dept. of Information Engineering
University of Padova
Padova, Italy
zorzi@dei.unipd.it

***Abstract*—The Robotic Vessels as-a-Service (RoboVaaS) project aims to provide innovative services for the shipping activities and near-shore maritime operations such as ship-hull and quay walls inspection, anti-grounding service and environmental data collection. To this end, both underwater and above water communications will be employed. In this paper, we designed a polling based MAC layer protocol (UW-POLLING) for a data-muling scenario, in which a cluster-head node moves collecting data from sensor nodes deployed in the network and, optionally, forwards the data to a sink node directly connected to shore via an above water wireless link. We analyzed the protocol performance employing the DESERT Underwater Network Simulator by including the port of Hamburg bathymetry in the World Ocean Simulation System (WOSS).**

***Index Terms*—Underwater acoustic networks, polling protocol, Bellhop ray tracer, WOSS, DESERT Underwater.**

I. Introduction and related works

Underwater and above water marine assets may be required to exchange information in order to accomplish a common task or mission. Acoustic communication systems are typically employed for underwater data transfer, as acoustic transmission and signal processing techniques have been developed for considerable time now, and have reached a remarkable level of maturity [1], [2]. New acoustic low power sensor nodes can be deployed for a large amount of time before retrieval for maintenance [3], and the data acquired by these sensors can be collected from either autonomous surface [4] vehicles (ASVs) or autonomous underwater [5] vehicles (AUVs) equipped with both an acoustic modem and an above water wireless link (such as a point-to-point radio-link or a satellite link). The ASV exploits the latter to upload the collected data to shore, while the AUV can upload the data when resurfaced and an above water wireless link becomes available. Alternatively, in port scenarios, the vehicles can exploit the existing cellular UMTS/LTE network coverage.

The Robotic Vessels as-a-Service (RoboVaaS) project intends to exploit both the communication and the marine vehicle technologies described above to revolutionize shipping and near-shore operations, offering on-demand robotic-aided services (Figure 1). Specifically, these services include ship-hull inspection service, quay walls inspection service, anti-grounding service, and environmental and bathymetry data collection service.

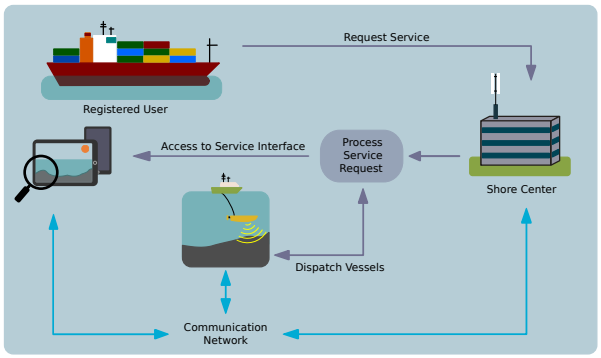

Fig. 1. RoboVaaS envisioned example scenario showing ship-hull inspection and anti-grounding services enabled through a fleet of ASVs and AUVs connected with acoustic underwater and radio communication. The figure also portrays inclusion of the control center and cloud integration.

In this paper, we focus on the study of the underwater environmental data collection service, performed by one or more autonomous vehicles equipped with an acoustic modem and moving along submerged acoustic sensor nodes (Figure 2). This study analyzes the feasibility of such service in the scenario of the port of Hamburg, characterized by shallow turbid fresh water, where the RoboVaaS concept will be demonstrated as part of the project. To this end, an underwater acoustic network composed by both static and moving nodes has been designed and simulated with the DESERT Underwater Network Simulation Framework [6] and the World Ocean Simulation System (WOSS) [7]. In our simulations, the moving nodes collect data from the sensors and upload it to

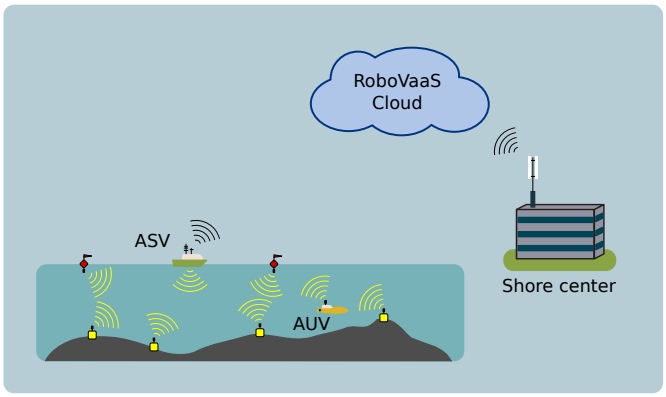

Fig. 2. RoboVaaS environmental data collection scenario, where both an ASV and an AUV collects data from static underwater sensor nodes.

the sink, a node directly connected to shore.

In order to overcome the challenges of the underwater acoustic channel [8], both MAC and routing protocols must be robust to link disruption. One of the main challenges in a mobile underwater network is to avoid packet collisions due to the long propagation delay. Protocol design has to address this problem, possibly exploiting the signal latency [9].

In [10] the authors demonstrated that a polling-based MAC layer (called UW-POLLING) suits well data-muling scenarios, by outperforming MAC designs based on random access, such as DACAP [11], CSMA and Aloha. In the polling protocol, a mobile node elected as cluster-head, triggers the channel and waits for the static nodes to answer with a probe (discovery phase). Before probing the channel, the static nodes wait for a backoff time randomly generated within a given interval. Each probe packet contains information on how much data the static node needs to transmit to the cluster-head. Then, the mobile node sends a poll packet, that assigns to each static node a time interval within which such node can transmit its own data packets.

UW-POLLING assumes each mobile node to be the sink, moving along the network and collecting data from sensor nodes. Instead, APOLL, presented in [12], is a polling-based protocol in which a mobile node can act either as a moving sensor node or as a sink. In the former case, the AUV patrols an area and collects the data from its own sensors, eventually forwarding it to a sink. In the latter, the AUV collects the data from other sensor nodes in the network. In this paper the UW-POLLING MAC layer [10] has been enhanced, by introducing the capability of muling the data from sensors to a sink.

The rest of this paper is organized as follows: in Section II we provide a detailed description of the UW-POLLING protocol presenting how the nodes interact with each other, Section III presents the simulation scenarios, providing the port of Hamburg bathymetry and sound speed profile, and the parameters used for the simulations. In Section IV we report the simulation results and, finally, in Section V we draw our conclusions and describe ongoing works.

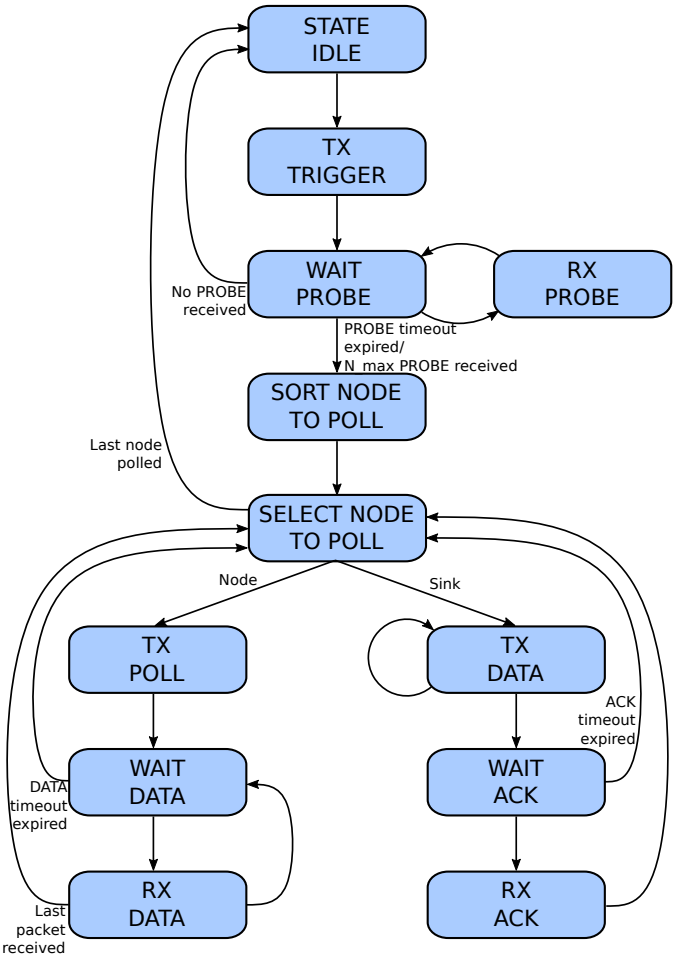

Fig. 3. State machine of the UW-POLLING protocol for an AUV.

II. UW-POLLING PROTOCOL

The MAC layer employed for retrieving the data from the underwater sensor network is based on the polling protocol, where a node acting as a cluster-head (the AUV) selects one by one which node can access the channel. Optionally, it can send the data to a sink node (the ASV) directly connected to shore. The MAC procedure is divided into two phases: the discovery phase and the polling phase. The AUV starts the discovery phase sending a TRIGGER packet (TrP), the nodes that receive this packet answer with a PROBE packet (PrP) to reveal themselves to the AUV, and then the AUV starts the polling phase selecting one by one which node is allowed to transmit its DATA packets by sending a POLL packet (PoP) to that node. If the selected node is actually a sink node, the AUV directly sends the DATA packets instead of sending a PoP. The whole procedure is described in Figure 3, Figure 4, Figure 5 and detailed below. At the beginning of the discovery phase the AUV sends a TrP to wake up the surrounding nodes. The TrP contains the Minimum Backoff Time ($T_{b_{min}}$) and the Maximum Backoff Time ($T_{b_{Max}}$) that will be employed by the nodes to generate a random backoff time used to access the channel. $T_{b_{min}}$ and $T_{b_{Max}}$ are stored

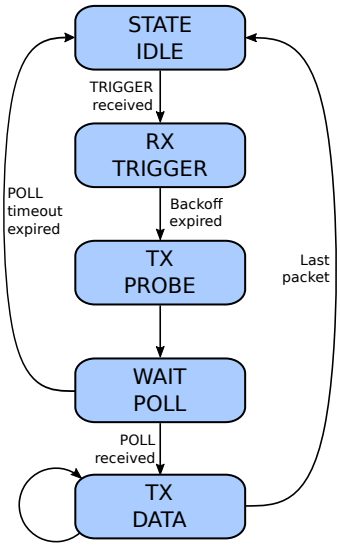

Fig. 4. State machine of the UW-POLLING protocol for a node.

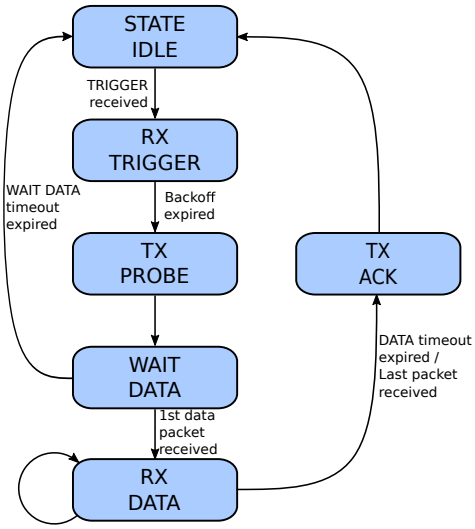

Fig. 5. State machine of the UW-POLLING protocol for a sink.

in the TRIGGER packet, so the AUV can adapt them to the network configuration. The AUV waits for the PROBE packets until either a timeout T_{probe} expires or a maximum number of PrPs (Pr_{Max}) are received. Each node i that received a TrP and has data to transmit sends back to the AUV a PROBE packet containing the number of packets it needs to transmit (Pks_i). The transmission of the PrP is contention based: each node transmits its PrP after a backoff period randomly chosen between $T_{b_{min}}$ and $T_{b_{Max}}$ with a uniform distribution. If the sink node receives the TRIGGER packet, it transmits a PrP as well, taking part in the contention.

After the transmission of a PROBE packet, the node starts a timeout T_{poll} within which it expects to receive a POLL packet from the AUV. If the PoP is not received within this timeout, the PrP is considered lost and the node returns to an IDLE STATE, waiting for another TrP. Similarly, the sink starts a timeout T_{data}^{SINK} within which it expects to receive the first data packet from the AUV.

The discovery phase ends when either the AUV has received Pr_{Max} packets or the timeout T_{probe} is expired.

If the AUV receives at least one PROBE packet the polling phase starts, otherwise the AUV moves to the IDLE STATE and then transmits another TrP. At the beginning of the polling phase the AUV creates the POLL list, that is the sorted list of the nodes that need to be polled. Node sorting depends on the adopted policy. To obtain a proportional fair scheduling, for each node i, AUV computes the weight $w_i = \frac{Pks_i}{P_{rx,i}}$, with $P_{rx,i}$ equal to the total number of packets already received from node i by the AUV, and sorts the nodes according to these weights in ascending order [13]. The sink is inserted in the list in the first position such that the sum of the packets in the AUV queue and the overall number of packets expected from the preceding node in the list is greater than or equal to the maximum number of packets the AUV can transmit to the sink in each round. If this condition is never reached, the sink is placed at the end of the list.

Once the POLL list is created, the AUV starts to poll the node according to the list order. The POLL packet contains the expected time the AUV should employ to poll all the nodes in the list. In such a way, all the nodes that receive a PoP can refine their own timeout T_{poll} according to the value inserted in the PoP. The sink can adapt the timeout T_{data}^{SINK} according to the value in the PoP as well. After sending the PoP to a node the AUV starts a timeout T_{data}^{AUV}, within which it expects to receive the DATA packets from the polled node i. The value T_{data}^{AUV} of this timeout is automatically set by the AUV, depending on the number of expected packets Pks_i and the estimated round-trip time (RTT) between the AUV and the polled node. When node i receives the POLL packet intended for itself, it starts to send back Pks_i packets to the AUV. At the end of the transmission the node moves to the IDLE STATE waiting for another TRIGGER packet. Once the AUV has received all the expected packets from the polled node it goes through the POLL list and sends the PoP to the next node. If not all the expected packets are correctly received, the AUV sends a new PoP to the next node in the list when the timeout T_{data}^{AUV} expires.

When the AUV finds the sink in the POLL list, it sends to it up to N_{Max} DATA packets previously received from the other nodes. Before sending the DATA packets, the AUV inserts in each packet a unique ID (UID) and the UID of the last packets that will be transmitted in that round. These values are used by the sink to send a cumulative acknowledge (ACK) when the last packet is received. If all the packets are correctly received the ACK contains the UID of the next expected packet. If some packets are lost a Selective Repeat (SR) Automatic Repeat Request (ARQ) mechanism is used, i.e., the ACK contains the UID of all the lost packets. The lost packet will be retransmitted when the AUV receives another PrP from the sink. When the first DATA packet is received by the sink, the T_{data}^{SINK} timeout is updated to a value within which the sink expects to receive all the packets from the

AUV. If the last packet is lost, the ACK is sent by the sink when T_{data}^{SINK} expires. If the ACK is not received by the AUV within a period T_{ACK} from the transmission of the last packet, the ACK is considered lost and the AUV starts to poll the next node in the list. In addition to this ACK, another ACK is inserted in the PrP sent by the sink. This second ACK employs the go-back-end (GBN) ARQ mechanism, i.e., the UID of the first lost packet, if any, is inserted in the PrP. If no packets are lost, the UID of the next expected packet is inserted in the PROBE packet. This second mechanism avoids the retransmission of the whole block of packets sent by the AUV in case the first ACK is lost.

When the AUV finishes polling all the nodes in the list the discovery phase starts again with the transmission of a new TRIGGER packet.

A. Timeout setting

An important part of the UW-POLLING protocol is the timeouts setting. In particular, T_{probe} has to be set to a value $T_{probe} \geq T_{b_{Max}} + RTT_{Max}$, where RTT_{Max} is the maximum RTT of a node in the network. In such a way all the PrPs are able to reach the AUV before T_{probe} expires.

When a node receives a PoP, the timeout T_{poll} is updated to the value inserted in the PoP. This value is computed each time a PoP is generated and is equal to:

$$T_{poll} = \sum_{i=1}^{N_{list}} \left(Pkts_i \cdot T_{data} + RTT_{est,i} + T_g \right) + N_{pkts}^{SINK} \cdot T_{data} + T_g \ , \quad (1)$$

where N_{list} is the number of remaining nodes in the POLL list (without considering the sink, if any), T_{data} is the time needed to transmit a DATA packet, $RTT_{est,i}$ is the estimated RTT between node i and the AUV, and T_g is a guard time, used to take into account the processing delay and errors in the RTT estimate. The last part of the equation computes the time needed to transmit all the data to the SINK, where N_{pkts}^{SINK} is the number of packets transmitted to the sink. This value is used by the sink to update T_{data}^{SINK} as well.

Once the first packet is received by the sink, the timeout T_{data}^{SINK} is updated again according to the time needed to receive the data. Since the UID of the last expected packet is inserted in the packet, the sink can retrieve the amount of data it will receive and estimate the time needed to receive it. The time needed to receive the data is $T_{rx} = N_{pkts}^{SINK} \cdot T_{data}$ and the timeout is set to:

$$T_{data}^{SINK} = T_{rx} + 0.5 \cdot T_{rx} \ . \quad (2)$$

The timeout T_{data}^{AUV} is computed as the estimated time needed to receive the expected data from the polled node, that is:

$$T_{data}^{AUV} = Pkts_i \cdot T_{data} + RTT_{est,i} + T_g \ . \quad (3)$$

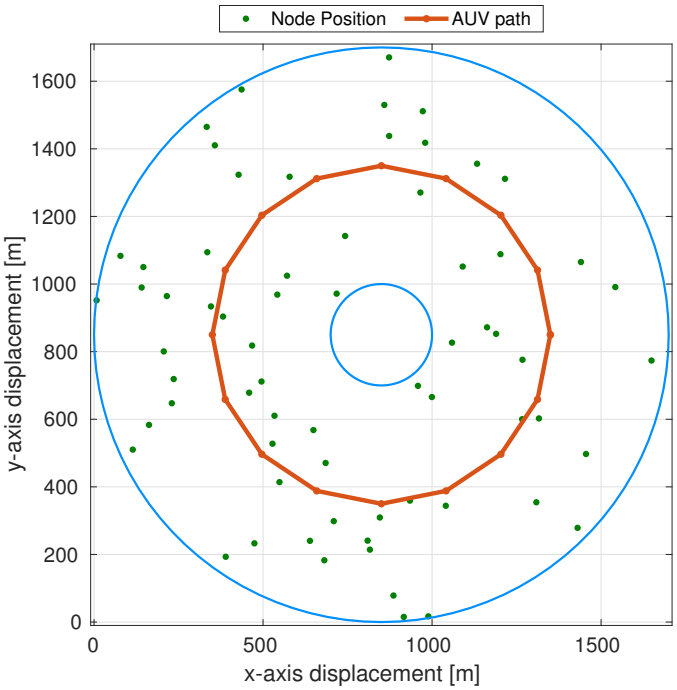

Fig. 6. Scenario 1: example of network scenario with a node density $N = 10$. The solid red line represents the path of the AUV. The blue lines define the area in which nodes are randomly placed.

III. Simulation scenario and system configuration

In our scenario, the underwater acoustic network is composed by one mobile node (AUV) collecting data simultaneously from submerged nodes. The AUV can either act as a sink, or act as a mule and forward data to a different sink node, statically placed in a fixed position. In detail, an AUV patrols a large area containing several static underwater acoustic sensor nodes, either anchored to the sea bottom or deployed from buoys. Each static node acts as an autonomous data collection entity, while the AUV collects the data from the static nodes. In addition, the AUV uploads the collected data to a node that is directly connected to shore with a radio link, and acts as the sink for the network.

In order to tune the maximum backoff time, we performed a simulation in a uniform and random nodes deployment, with one AUV performing the same circular path 5 times as depicted in Figure 6 (Scenario 1). The performance has been analyzed by varying the AUV speed (v), the node density (N), calculated as the average number of nodes deployed within the coverage area (with a radius of 350 meters), and the maximum backoff time $T_{b_{Max}}$. In this scenario we considered the AUV acting also as the sink node. In this case the bellhop ray-tracer has not been used, as the objective of this simulation is to select the backoff timeout rather than evaluate the protocol in a realistic environment. To perform this second task, we consider a more realistic scenario, where 11 nodes have been deployed along the Elba river, in the proximity of the port of Hamburg, as reported in Figure 7 (Scenario 2). In this scenario we considered also a sink node different from the AUV and

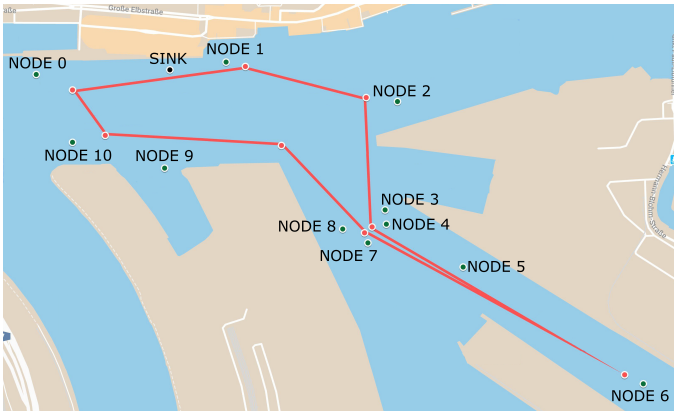

Fig. 7. Scenario 2: Hamburg port. The solid red line represents the path of the AUV.

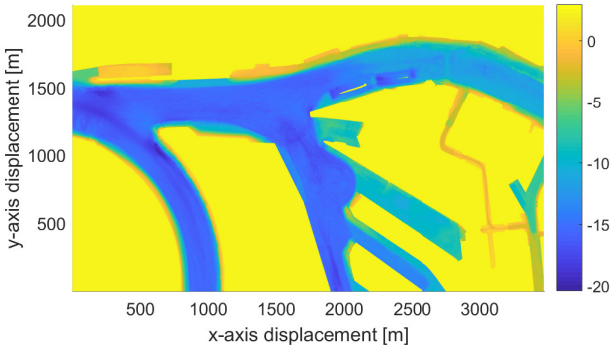

Fig. 8. Hamburg port bathymetry in the selected area.

statically placed in a fixed position. In this case we employed the Bellhop Ray tracer, by including the river bathymetry in our simulator. Section III-A presents a detailed explanation of how the acoustic channel of the port of Hamburg is modeled in our simulations, and Section III-B the simulation settings.

A. Simulation of the port of Hamburg acoustic channel

The port of Hamburg bathymetry and sound speed profile data have been included in the DESERT simulations through the WOSS framework to run the Bellhop Ray-Tracer and obtain an accurate acoustic propagation model. The Hamburg Port Authority (HPA) provided us with the bathymetry map of all the port area, and we selected a section of 3.5x2.1 km, with resolution of one sample per meter, exporting it into a csv text file. This file has been included in the WOSS framework [7], by creating the module `BathyHamburgPortDb`, for including the textual bathymetry database of the port of Hamburg.

B. Simulation settings

We assume to use a high frequency acoustic modem, with carrier frequency of 150 kHz and bandwidth of 60 kHz (like the EvoLogics S2C HS model [14]), as those frequencies are not heavily affected by the harbor shipping activity, at the price of a shorter transmission range [15] . The acoustic

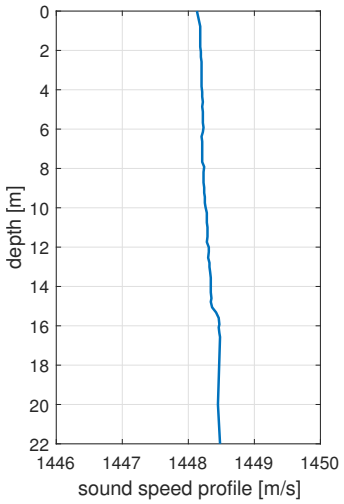

Fig. 9. Sound speed profile of the Hamburg port in the selected area.

source level has been set to 156 dB re 1 μPa @1m, and the data rate to 7 kbit/s. The length of the DATA packets has been set to $L = 125$ Bytes. For all the simulations we set the T_{probe} timeout equal to $T_{probe} = T_{b_{Max}} + 2$ s, the guard time $T_g = 1$ s and $T_{b_{min}} = 0$.

IV. Results

In this section we report the results obtained simulating the protocols in the two scenarios described above. In particular, in Section IV-A we analyzed how the backoff time affects the network performance of the first scenario. In Section IV-B we reported the results obtained in the scenario of the port of Hamburg.

A. Scenario 1: backoff time results

An important parameter to be set in the UW-POLLING protocol is $T_{b_{Max}}$, i.e., the Maximum Backoff Time a node can select to send a PROBE packet. The best $T_{b_{Max}}$ depends on the node density (N) and the AUV speed (v): the former determines how many nodes take part in channel contention, and the latter how long a node will be in the transmission range of the AUV.

In the first scenario the results have been obtained averaging 20 independent simulations. In this scenario we considered that each node has always 5 packets to transmit to the AUV when it is polled. We analyzed the network performance in terms of throughput (Thr) and PROBE packet delivery ratio (PDR_{probe}) varying the maximum backoff time $T_{b_{Max}}$ from 0.01 to 60 s. The throughput is computed as:

$$Thr = \frac{N_{rx}^{AUV} \cdot L}{T_{sim}}, \tag{4}$$

where N_{rx}^{AUV} is the overall number of packets received by the AUV during the simulation, L is the length of the packets,

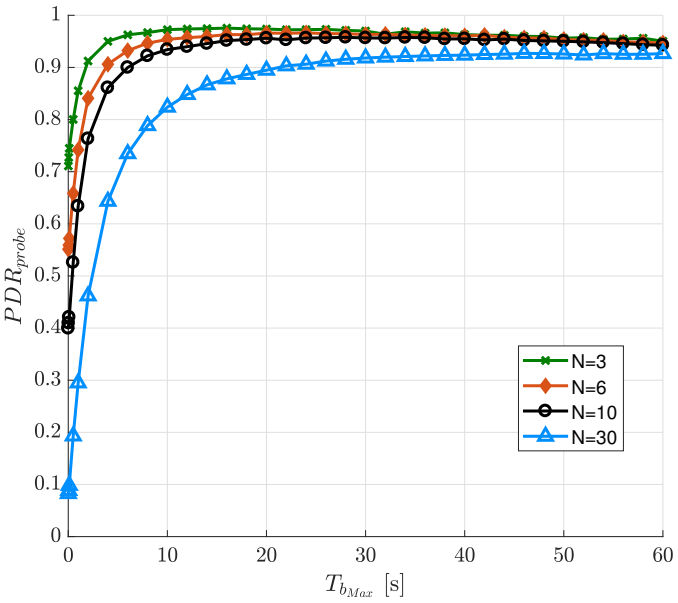

Fig. 10. PROBE packet delivery ratio, varying $T_{b_{Max}}$, for different node density N and fixed AUV speed $v = 2$ m/s.

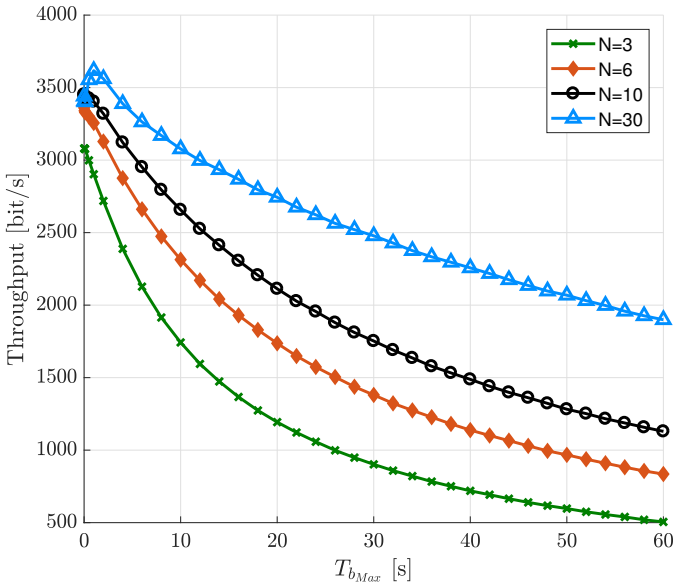

Fig. 11. Overall throughput of the network, varying $T_{b_{Max}}$, for different node density N and fixed AUV speed $v = 2$ m/s.

and T_{sim} is the duration of the simulation. The PDR_{probe} is computed as:

$$PDR_{probe} = \frac{\sum_{i=1}^{N_{nodes}} N_{tx,i}^{PrP}}{N_{rx}^{PrP}} , \quad (5)$$

where N_{nodes} is the number of nodes in the network, $N_{tx,i}^{PrP}$ is the number of PrPs transmitted by node i and N_{rx}^{PrP} is the overall number of PROBE packets received by the AUV. First of all, we run simulations varying the node density N from 3 to 30, considering a fixed AUV speed $v = 2$ m/s. Then, to investigate the impact of the AUV speed on the network performance, we run different simulations with a fixed node density $N = 10$, but varying the AUV speed from 0.1 to 4 m/s.

Figure 10 depicts the PROBE PDR obtained with different values of the node density N. Considering a node density $N = 3$, we can observe that for $T_{b_{Max}} \geq 2$ s the PDR is always bigger than 0.9. The maximum PDR, equal to 0.97, is obtained when $T_{b_{Max}} = 16$ s. For $T_{b_{Max}} > 16$ s, the PDR slightly decreases, due to the AUV mobility. Indeed, if the time between the TRIGGER reception and the PROBE transmission is too long, the AUV could go out of range, therefore the TRIGGER is correctly received but the PROBE is not. Moreover, even with $T_{b_{Max}} = 0.01$ s, that is comparable with the transmission time of a PROBE packet ($T_{tx}^{probe} = 16$ ms), the packet delivery ratio has a value of 0.7. This is due to the low propagation speed of acoustic waves, therefore 2 nodes can simultaneously transmit without colliding at the receiver if their distance from the receiver (AUV) differs by at least 24 m, which is the distance covered by the acoustic waves in T_{tx}^{probe}. A similar behavior has also been addressed in [16]. Similarly, with a node density $N = 6$, when $T_{b_{Max}} \geq 4$ s the PDR is always bigger than 0.9, increases up to a maximum of 0.95 when $T_{b_{Max}} = 20$ s and then slightly decreases for higher values of $T_{b_{Max}}$. Considering $N = 10$, we observe that the PROBE PDR is bigger than 0.9 when $T_{b_{Max}} \geq 8$. In this case the maximum value of PDR is equal to 0.95 and is reached for $T_{b_{Max}} = 28$ s. The network with $N = 30$ nodes per coverage area has the worst performance in terms of PROBE PDR. Indeed the PDR becomes bigger than 0.9 only when $T_{b_{Max}} \geq 22$ s. In this scenario, with $T_{b_{Max}} = 0.01$ s the PDR drops below 0.1.

Figure 11 reports the overall throughput of the network, obtained with different values of the node density N. With $N = 30$ the value of $T_{b_{Max}}$ that maximize the throughput is a trade-off between the number of PROBE packets correctly received by the AUV and the time spent by the AUV waiting for the PROBE packets after the transmission of a TRIGGER. The maximum throughput, equal to 3.6 kbit/s, is reached with $T_{b_{Max}} = 1$ s. If $T_{b_{Max}}$ is lower than 1 s the number of received PROBE packets is too small, and therefore only a few nodes are allowed to transmit their DATA packets. With $T_{bMax} > 1$ s the time waited by the AUV to receive the PROBE packets is predominant and therefore the throughput decreases. With lower values of the node density, i.e., $N = 3$, $N = 6$ and $N = 10$, the time spent by the AUV waiting for the PROBE packets is predominant, and therefore the throughput decreases as $T_{b_{Max}}$ increases.

We analyzed the performance of the network in terms of fairness as well. In particular, we compute Jain's Fairness Index (JFI) as [17]:

$$JFI = \frac{\left(\sum_{i=1}^{N_{nodes}} P_{rx,i}\right)^2}{N_{nodes} \sum_{i=1}^{N_{nodes}} P_{rx,i}^2} , \quad (6)$$

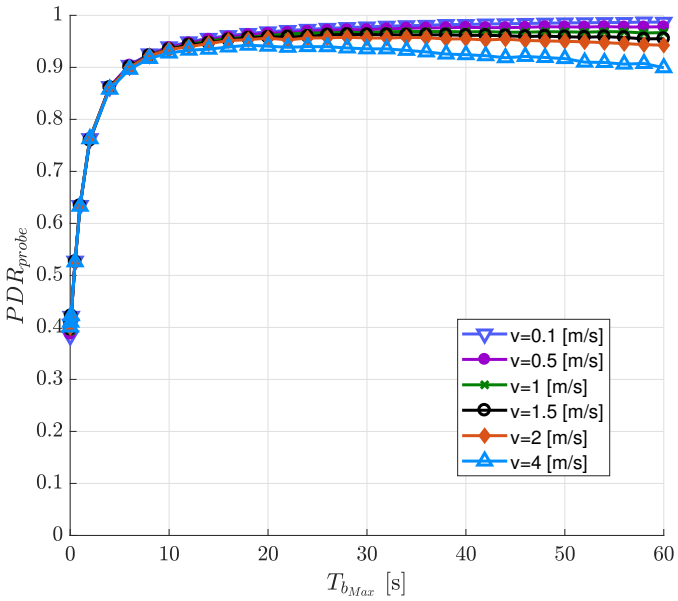

Fig. 12. PROBE packet delivery ratio, varying $T_{b_{Max}}$, for different AUV speed v and fixed node density $N = 10$.

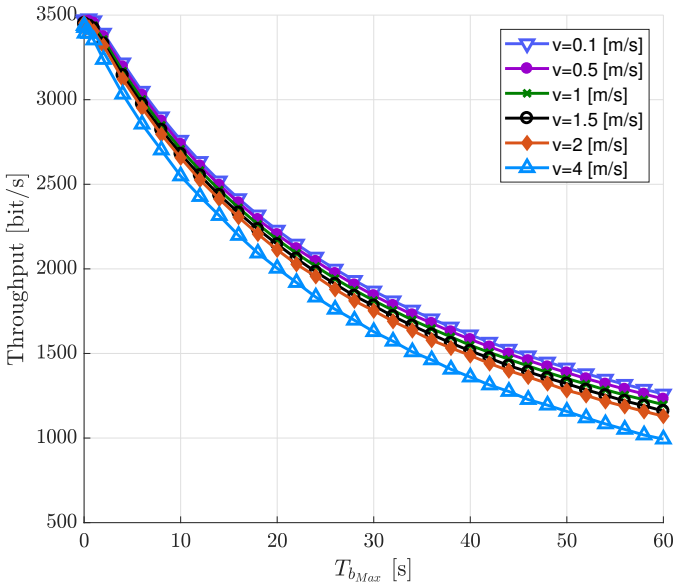

Fig. 13. Overall throughput of the network, varying $T_{b_{Max}}$, for different AUV speed v and fixed node density $N = 10$.

where $P_{rx,i}$ is the number of DATA packets received by the AUV from node i. The simulation results show that, for all the considered node densities N, JFI is bigger than 0.9 for all values of $T_{b_{Max}} > 1$ s.

Figure 12 reports the PROBE PDR as a function of $T_{b_{Max}}$ and for different values of the AUV speed v. For $T_{b_{Max}} \leq 10$ s the PDR turns out to be equivalent for all the considered v. For bigger values of $T_{b_{Max}}$ the lower the AUV speed, the bigger the PDR. In particular, with $v = 0.1$ m/s the PDR never decreases when the maximum backoff time increases. This is because with this speed the AUV moves a few meters between the TRIGGER transmission and the PROBE reception, and therefore the triggered node will likely never go out of range. Indeed, considering the worst case with a backoff time equal to 60 s, the distance traveled by the AUV between the TRIGGER transmission and the PROBE reception is about 6 m. Differently, with an AUV speed $v = 4$ m/s, the PDR slightly decreases as $T_{b_{Max}}$ increases, because the probability than a node goes out of range between the reception of a TRIGGER and the transmission of the PROBE is high.

Figure 13 depicts the overall throughput as a function of $T_{b_{Max}}$ for different values of the AUV speed v. Similarly to Figure 11 the throughput decreases with $T_{b_{Max}} > 0.5$ s; in addition, it slightly decreases as the AUV speed increases.

B. Scenario 2: Hamburg port results

In this subsection we report the results of the simulations in the Hamburg port scenario. In this scenario, first we perform different simulations without employing the Bellhop ray tracer, i.e., without using the WOSS framework, letting the AUV perform both 2 and 10 laps of the path depicted in Figure 7. In this case the channel model employed for the simulations is the one described in [18]. Then, we run a simulation with the Bellhop ray tracer, employing the port of Hamburg Bathymetry imported in the WOSS framework, to compare the previous results in a more realistic channel. In this case we let the AUV perform 2 laps of the same path.

In these simulations we use a maximum backoff time $T_{b_{Max}} = 10$ s and the AUV moves at the speed of $v = 2$ m/s. We analyzed the network performance in terms of DATA packet delivery ratio and throughput, observed as a function of the average generation period of the DATA packets (T_{app}).The average generation period is equal for all the sensor nodes in the network.

Figure 14 shows the DATA PDR computed at both MAC and APP layers. The PDR computed at the MAC layer is close to 1 for $T_{app} \geq 10$ s. For $T_{app} < 10$ s, the PDR decreases because the AUV does not have enough time to forward all the collected DATA packets to the sink node. The simulations performed with the WOSS framework and Bellhop ray-tracer confirm the PDR trend. Considering the cases in which 2 laps are performed, both with and without WOSS, we can observe that the PDR computed at the APP layer never gets close to 1. With $T_{app} \geq 22$ s the PDR is about 0.9. This is due to the fact that the nodes continue to generate packets also in the last lap. Indeed, when the AUV passes by a node it collects all the data generated so far, but the data generated after the last contact will never be collected by the AUV as the simulation is over. In addition, with 10 s $\leq T_{app} < 22$ s, some packets are not sent to the sink, because the nodes close to the sink are not able to transmit all their DATA packets to AUV. This happens because when the AUV gets in range with the sink it has too much DATA packets to send and spends most of the time forwarding data to the sink. For this reason the other nodes in the sink range have fewer opportunities to be polled. To overcome this issue, the protocol should be enhanced, letting

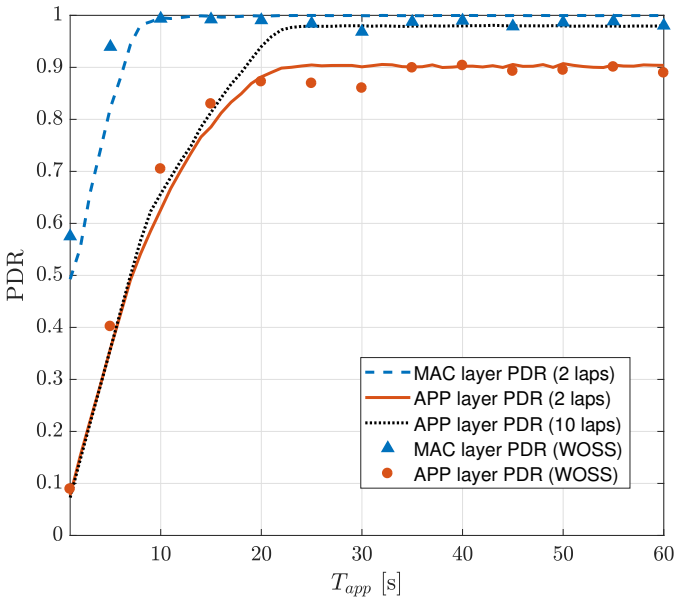

Fig. 14. APP and MAC layer packet delivery ratio. PDR is obtained either employing or not the WOSS Framework.

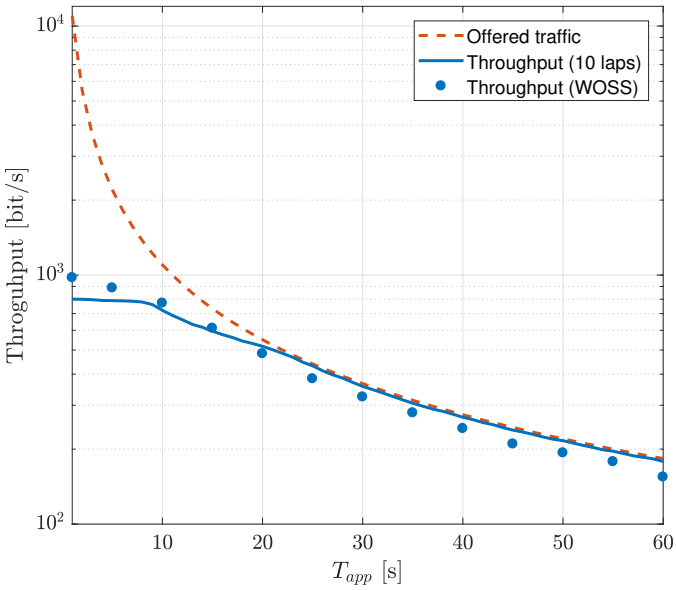

Fig. 15. Throughput obtained with the AUV performing 10 laps (solid blue line), with the AUV performing 2 laps and employing WOSS framework (blue circle) and compared with the generated traffic (dashed red line).

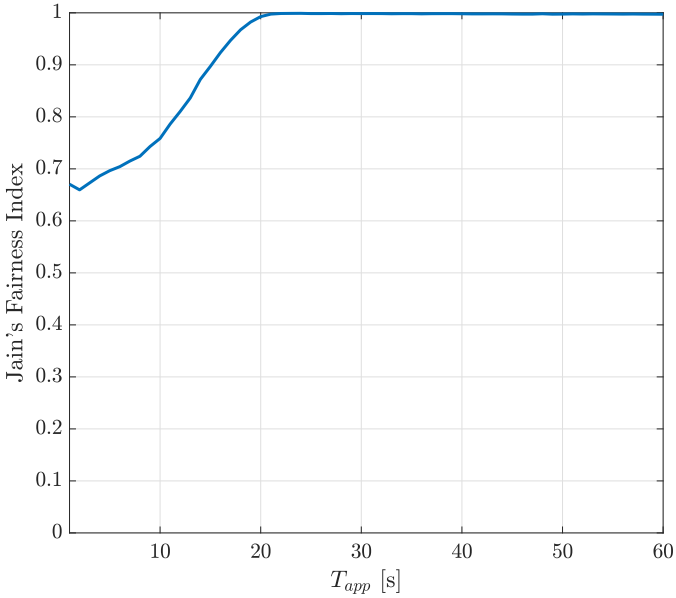

Fig. 16. Jain's fairness index obtained with the AUV performing 10 laps.

the sink collect itself the DATA packets from the surrounding nodes. Like for the MAC PDR, with $T_{app} < 10$ s the PDR goes further down because the AUV is not able to forward all the DATA packets to the AUV. We also analyzed the PDR computed at the APP layer, with the AUV performing 10 laps. In this case, with $T_{app} \geq 22$ s the PDR goes closer to 1 than in the previous cases. This is because the number of packets generated in the last lap and not collected by the AUV is independent of the number of laps, but their relevance in terms of PDR becomes lower when the overall number of transmitted packets increases, as in the case when the AUV performs 10 laps.[1]

Figure 15 depicts the throughput in the case when the AUV performs 10 laps and in the case it performs 2 laps using the WOSS framework along with the Bellhop ray-tracer. Moreover, we plot also the overall offered traffic of the network. Considering the scenario with 10 laps, the maximum throughput is around 790 bit/s with $T_{app} < 10$ s. This is because the network is not able to handle more than this amount of generated traffic and the exceeded packets are not transmitted, remaining either in the AUV or in the nodes queues. With $T_{app} \geq 10$ s and $T_{app} < 22$ s, the throughput is lower than the generated traffic, because, as described above in the PDR analysis, the nodes close to the sink are not able to transmit all their DATA packets to the AUV. With $T_{app} > 20$ s the network is able to handle all the generated traffic by the nodes. We also report the results obtained employing the WOSS framework along with the Bellhop ray-tracer and we can observe that the trend is similar to the one obtained in the scenario without the use of the ray tracer.

Figure 16 reports the fairness results. The drop of JFI for $T_{app} < 20$ s confirms the fact that the nodes close to the sink are not able to transmit all their DATA packets to the AUV.

V. Conclusions and Future Works

In this paper we presented an enhancement of the UW-POLLING MAC protocol. We introduced the capability of a mobile node to collect data from sensor nodes and then

[1]We expect a similar behavior also with WOSS and the Bellhop ray-tracer, however we could not verify this statement as the WOSS simulation was not feasible with our computation capabilities: the simulation with two laps required the full computational power of a new generation core i7 with 16 GB of RAM, and lasted almost 2 days, more laps would have required more time and, most important, more RAM, which was not available.

forward them to a sink node directly connected to shore with a radio link. First, we analyzed how the maximum backoff time affects the network performance, and then we simulated the protocols in the scenario of the port of Hamburg. To this purpose we performed the simulations introducing the Hamburg port bathymetry in the WOSS framework, to employ the Bellhop ray-tracer. We analyzed the maximum offered traffic that can be supported employing UW-POLLING protocol in this scenario.

Future works will include the possibility to use multiple sink nodes at the same time, to automatically adapt the backoff time according to the number of nodes in the AUV range and to enhance the policy used to POLL the nodes in order to improve the fairness also with a large offered traffic.

ACKNOWLEDGMENTS

This work has been supported in part by the Italian Ministry of Education, University and Research (MIUR), and ERA-NET Cofound MarTERA (contract 728053). The authors are grateful to the Hamburg Port Authority (HPA) for providing the port bathymetry and sound speed profile, and to Federico Guerra for the support on using the World Oceans Simulation System (WOSS).

REFERENCES

[1] M. Chitre, S. Shahabudeen, and M. Stojanovic, "Underwater acoustic communications and networking: Recent advances and future challenges," *Marine Tech. Soc. Journal*, vol. 42, no. 1, pp. 103–116, spring 2008.

[2] C. Renner and A. J. Golkowski, "Acoustic Modem for Micro AUVs: Design and Practical Evaluation," in *Proc. of the 11th ACM International Conf. on Underwater Networks & Systems (WUWNet)*, Shanghai, China, Oct. 2016.

[3] "Evologics S2C beacon," accessed: Jan. 2019. [Online]. Available: https://www.evologics.de/en/products/acoustics/s2c_beacon.html

[4] "Sonobot autonomous hydrographic survey vehicle," accessed: Jan. 2019. [Online]. Available: https://www.evologics.de/en/products/sonobot/index.html

[5] "Folaga data sheet," accessed Jan. 2019. [Online]. Available: https://www.graaltech.com/folaga-features

[6] F. Campagnaro *et al.*, "The DESERT underwater framework v2: Improved capabilities and extension tools," in *Proc. Ucomms*, Lerici, Italy, Sep. 2016.

[7] "The World Ocean Simulation System - *WOSS*," Last time accessed: May 2019. [Online]. Available: http://telecom.dei.unipd.it/ns/woss/

[8] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," in *WUWnet*, 2006.

[9] R. Diamant *et al.*, "Leveraging the Near–Far Effect for Improved Spatial-Reuse Scheduling in Underwater Acoustic Networks," *IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS*, vol. 16, no. 3, pp. 1480–1493, 2017.

[10] F. Favaro, P. Casari, F. Guerra, and M. Zorzi, "Data upload from a static underwater network to an AUV: Polling or random access?" in *Proc. MTS/IEEE Oceans*, Yeosu, Republic of Korea, May 2012.

[11] B. Peleato and M. Stojanovic, "Distance aware collision avoidance protocol for ad hoc underwater acoustic sensor networks," *IEEE Commun. Lett.*, vol. 11, no. 12, pp. 1025–1027, Dec. 2007.

[12] W. Liu, J. Weaver, L. Weaver, T. Whelan, R. Bagrodia, P. A. Forero, and J. Chavez, "APOLL: Adaptive polling for reconfigurable underwater data collection systems," in *Proc. MTS/IEEE OCEANS*, Kobe, Japan, May 2018.

[13] G. Miao, J. Zander, K. W. Sung, and S. B. Slimane, *Fundamentals of mobile data networks*. Cambridge University Press, 2016.

[14] "Evologics S2C M HS modem," accessed: Sep. 2018. [Online]. Available: http://www.evologics.de/en/products/acoustics/s2cm_hs.html

[15] E. Coccolo, F. Campagnaro, A. Signori, F. Favaro, and M. Zorzi, "Implementation of AUV and ship noise for link quality evaluation in the desert underwater framework," in *Proc. ACM WUWNet*, Shenzhen, China, Dec. 2018.

[16] W. Liu, J. Weaver, T. Whelan, R. Bagrodia, P. A. Forero, J. Chavez, and M. Capella, "Adaptive polling for underwater data collection systems," in *Proc. MTS/IEEE OCEANS*, Charleston, USA, Oct. 2018, pp. 1–7.

[17] J.-Y. Le Boudec, *Performance evaluation of computer and communication systems*. Epfl Press, 2011.

[18] M. Stojanovic, "On the relationship between capacity and distance in an underwater acoustic communication channel," *ACM SIGMOBILE Mobile Computing and Communications Review (MC2R)*, vol. 11, pp. 34–43, 2007.