Coastal Patrol and Surveillance Networks using AUVs and Delay-Tolerant Networking

Saiful Azad, Paolo Casari, Michele Zorzi

Department of Information Engineering, University of Padova, Italy — {azad, casarip, zorzi}@dei.unipd.it

***Abstract*—In this paper, we consider a coastal surveillance scenario, where Autonomous Underwater Vehicles (AUVs) patrol an area of interest and inspect surface ships or underwater assets passing through the area. A shore-based control center monitors the AUVs by means of delay-tolerant networking techniques. In particular, as the AUVs carry out their patrolling task, they may get in contact with one another and have a chance to exchange data about the inspected assets (identity, route followed, movement speed, etc.). Given that the area to be patrolled is usually quite large, these contacts are erratic and time-limited: this makes the AUVs and the sink a Delay-Tolerant Network (DTN). To make the communication between AUVs more effective during a contact, we propose a DTN protocol which splits the estimated contact duration between the nodes involved and enhance this protocol using an Automatic Repeat reQuest (ARQ) technique based on selective repeat for error control. Moreover, the structure of the signaling packets exchanged prior to data transmission is designed to help estimate the contact duration and thereby optimize the subsequent data packet exchange. Simulation results demonstrate that the proposed protocol outperforms other ARQ-based DTN routing protocols.**

I. Introduction and Context

The advancements in the design of AUVs over the last decade are encouraging coastal authorities to consider the use of these automatic devices in marine activities. In this paper, we focus on coastal surveillance using AUVs. As a typical scenario, we assume that a shore-based control center monitors the behavior of a network of AUVs that patrol an area of interest and inspect any asset (ship, boat or other elements) passing through the area. When an asset enters into the surveilled area, one or more AUVs start following it. While the asset moves, the responsibility of the follower AUVs is to detect some desired data of interest about the asset. To fix ideas, we assume that the AUVs track and store the trajectory of the asset they are following. The trajectory data is timestamped and reported to the control center ashore whenever there is a chance to do so. We assume that all communications are acoustic, hence the control center is also endowed with an acoustic transceiver placed off the shore.

Because the area to be patrolled may be large, the AUVs may be out of the range of the shore center most of the time. Hence, they may need to ask other nodes to relay their own data to the shore in an opportunistic, store-and-forward manner. Therefore, the group of AUVs can be considered as a Delay-Tolerant Network (DTN) [1]. In particular, whenever an AUV detects a contact with another AUV following a different asset, the two nodes can opportunistically exchange relevant trajectory information regarding their own asset, as well as other data they received from other nodes. The data is given a certain priority, so that the data with highest priority can be transmitted first. In the following, we assume that the most recently generated data also has the largest priority. Whenever a follower comes into the communication range of the shore center, it delivers both its own data and the data it is relaying for other nodes directly to the shore center (as long as the contact so allows); it will also keep track of the data transferred successfully, in order to avoid retransmitting the same information twice.

Like most DTNs, the network of AUVs considered in this paper is characterized by intermittent connectivity, both one-hop and end-to-end. Therefore, routing protocols which search and establish complete end-to-end routes before data transmission are not suitable [2], [3]; conversely, routing protocols which employ some form of "store-and-forward" approach are usually preferred [1], [4]. Since the contacts between the nodes are infrequent, and their duration is not known a priori, the performance of a DTN routing protocol primarily depends on the effective exchange of data upon the occurrence of contacts. In addition, these exchanges are vulnerable to errors caused by the underwater channel, which call for the use of some form of error control, e.g., via Automatic Repeat reQuest (ARQ).

In this paper, we propose a DTN protocol which makes the communication between AUVs more effective during a contact by employing an efficient packet exchange technique and an ARQ mechanism. The protocol is tailored for the coastal surveillance scenario described above, and favors the effective transmission of sensed data (in this case, the trajectory of the assets being inspected) from the AUVs to a control center ashore, possibly via store-and-forward communications among other AUVs in the network.

Our protocol estimates the contact duration so that both nodes in contact can employ a fair share of the contact time to transmit their own data. Data packets are assigned a priority (in accordance with the example above, the newest data packets generated are given the highest priority), in order to transmit the most important data first. In addition, we incorporated a modified version of the Underwater Selective Repeat (USR) [5] ARQ technique that provides an effective form of error control. Moreover, the structure of the signaling phase performed prior to data transmission is designed so as to assist the estimation of the contact duration. In turn, this allows the nodes to improve the effectiveness of the subsequent data packet exchange.

The rest of the paper is organized as follows. In SectionII we

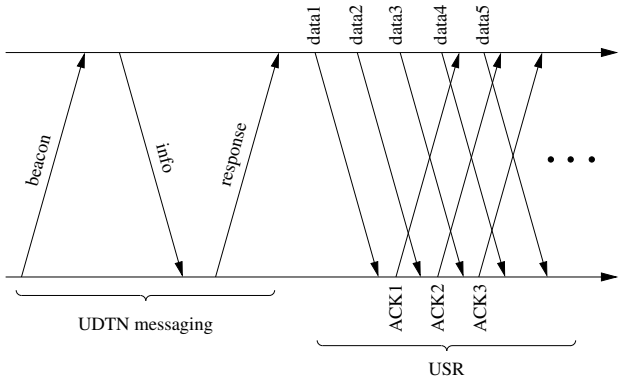

Figure 1. Signaling and data packet exchange pattern using the proposed UDTN technique with the USR protocol.

outline some related work on DTN routing; in Section III we describe our DTN routing protocol; in Section IV we outline the simulation scenario and the mobility model employed in the simulations; in Section V we describe the results of our simulations; in Section VI we draw some concluding remarks.

II. Related Work

Several DTN protocols have been proposed in the literature. One of the typical criteria to classify such protocols refers to whether or not packet replication is employed. Routing protocols which replicate packets are termed replication-based [1], [4], as opposed to forwarding-based protocols, which never replicate packets [6], [7]. As a general observation, replication-based protocols achieve better packet delivery ratio than forwarding-based protocols, at the price of greater overhead due to the multiple packet replicas injected into the network.

Epidemic routing [8] is a flooding-based routing scheme where a node continuously replicates and transmits packets to newly discovered contacts that do not already own a copy of the same packets. Being based on flooding, epidemic routing entails a large number of transmissions, and may not be suitable for underwater networks in general. Therefore, some routing protocols have been proposed to pursue the benefits of replication-based routing protocols while limiting the replication overhead. Spray and wait [4] is such a protocol, where a node can replicate up to a fixed number of copies of each packet. Two versions exist: the "vanilla" version allows only the source of a packet to replicate it, whereas the "binary" version allows both the source and any intermediate relay to do so. PROPHET [9] is another protocol which limits the packet replication by forwarding a packet only to those neighbors exhibiting a higher probability of reaching the packet destination in a short time. RAPID [1] is another such protocol, which computes the utility of contacts based on a routing metric (average delay, missed deadlines, or maximum delay) to be optimized, and replicates first those packets that exhibit the highest utility.

Most of the protocols described above are designed for terrestrial networks, where the transmission range of a modem is limited to around one hundred meters and the propagation of signals can be considered instantaneous; on the contrary, typical underwater modems support ranges of up to a few kilometers, and the underwater sound propagation delay is much higher.

A properly designed DTN routing protocol should leverage on the above properties to its own advantage. For example, this includes estimating the duration of contacts so as to understand how many packets can be exchanged during a contact, and fully exploiting the contact via efficient schemes for packet transmission and error control. In the following section, we will explain how the DTN routing protocol proposed in this paper achieves the two objectives above.

III. Underwater Delay Tolerant Network (UDTN) Protocol

The Underwater DTN (UDTN) routing protocol we propose in this paper is described in the next two subsections. Subsection III-A provides the details of the preliminary signaling phase employed to discover neighbors and organize the transfer of packets upon the occurrence of contacts, whereas Subsection III-B describes the ARQ technique employed for error control.

A. Preliminary UDTN messaging for neighbor discovery and contact setup

We start by illustrating the messaging scheme of UDTN, for which we also refer to Fig. 1.

Every node periodically sends a *beacon* packet (except the final destination of the data packets, or sink, which only acts as a receiver). After the transmission of the beacon, the sender allows a sufficient time to receive answers from other nodes within a given communication range. If no answer is heard, the node assumes that no other nodes are present nearby, and transmits a new beacon.

Call A the beacon sender. When another node (called B in the following) receives the beacon, it replies with an *info* packet containing its current position and velocity which allows the beacon sender to estimate the relative velocity of the nodes, hence the contact duration. The info packet includes an estimate of B's position and of its movement speed vector, as well as a summary of the contents of B's buffer. The summary is strictly related to the application to be supported (coastal surveillance, in this paper). As described in Section I, the network nodes are AUVs moving to track a certain asset. These AUVs store data packets containing samples of the position of the asset being followed. Given that the most recent packets have the highest priority, a good summary of B's buffer is composed of the most recent data packet available about every asset currently being followed in the network area.[1] This is accomplished by putting several (asset ID, timestamp) pairs in the *info* packet, one for each asset B stores information about.

[1]We stress that the summary includes information about every asset, not just the asset followed by B. In fact, B may be storing data packets received from other nodes, which hence provide information about other assets. These packets should be also be transmitted to A, in order to improve the chance that the sink receives them.

When A receives the *info* packet from B, it first estimates the duration of the contact with B. The estimation of the contact duration is given as follows. Call $\vec{P}_A$ and $\vec{P}_B$ the positions of nodes A and B, respectively. Also, call their speed vectors $\vec{V}_A$ and $\vec{V}_B$, where the modulus of the vectors is expressed in m/s. Therefore, the relative position of A and B is $\vec{P}_R = \vec{P}_A - \vec{P}_B$ and the relative velocity of those nodes is $\vec{V}_R = \vec{V}_A - \vec{V}_B$. Using $\vec{P}_R$ and $\vec{V}_R$, we can find the instantaneous distance $R(t)$ at time t as

$$R(t) = \sqrt{\|V_R\|^2 t^2 + 2(\vec{P}_R \cdot \vec{V}_R)t + \|P_R\|^2} \tag{1}$$

where $\cdot$ denotes the inner product, and $\| \ \|$ is the 2-norm. Note that in (1) we are assuming $\vec{P}_R$ and $\vec{V}_R$ to be constant and equal to the values read from the *info* packets, even though in principle they may vary over time. The approximate contact duration can be computed as the time required for the nodes to exit the communication range of each other. Therefore, if we replace $R(t)$ with the transmission range T_R of the modem in use (usually known from the data sheets provided by the modem vendors [10]–[12]), we can find the contact duration T_c as

$$\|V_R\|^2 T_c^2 + 2(\vec{P}_R \cdot \vec{V}_R)T_c + \|P_R\|^2 = T_R^2 \tag{2}$$

Solving Equation (2), the approximate contact duration can be found as follows

$$T_c = \frac{-(\vec{P}_R \cdot \vec{V}_R) + \sqrt{(\vec{P}_R \cdot \vec{V}_R)^2 - \|V_R\|^2(\|P_R\|^2 - T_R^2)}}{\|V_R\|^2} \tag{3}$$

First, A checks that T_c is greater than a given threshold, equal to the time needed so that both A and B can transmit at least one data packet and receive its related ACK packet.[2] If no contact is expected to exceed this threshold, A goes back to the idle state and restarts periodic beacon transmissions after a backoff time.

After computing the approximate contact duration, A derives the share of the transmission time to be assigned to A and B as $T_s = \eta T_c/2$, where where $\eta < 1$ is a margin factor required to compensate for inaccuracies in the estimation of the contact duration, e.g., due to uncertainties in the position or speed estimates.[3]

At this point A is ready to send a *response* packet to B. In this packet, A includes the share of the transmission time to be assigned to B as well as a summary of its own buffer. If any other nodes transmitted *info* packets in response to A's beacon, they go back to the idle state when they hear the *response* packet for B. After transmitting the *response* packet, A waits for data packets from B. The number of packets to be transmitted can be estimated from the share of transmission time T_s as

$$\sigma = \left\lfloor \frac{T_s - \tau}{T_D + T_A + \varepsilon} \right\rfloor \tag{4}$$

where τ is the one-way propagation delay between A and B, T_D is the transmission time of a data packet, T_A is the transmission time of the acknowledgement packet, and ε is a guard time. We recall that the actual packet transmission will follow the rules of the USR protocol in Subsection III-B, and therefore we note that the value of σ should depend on W in Equation (6), which in turn depends on an updated estimate of the round-trip time (RTT). This estimate is not available during UDTN's signaling phase, hence a rough indication of the RTT is incorporated in ε.

After calculating σ, B retrieves from its buffer the packets to be transmitted . This is done in accordance with the summary of A's buffer, and in such a way that no packet is transmitted if A has no interest in it (referring again to our coastal surveillance application, no packet is transmitted to A about a given asset if A has more recent information about that asset). The packets are retrieved from the buffer so as to maximize the transfer of information to A regarding the status of the assets. Assuming that B is storing data about multiple assets, it will transmit one data packet per asset in a round-robin fashion, starting from the most recent packets (i.e., those with highest priority), until its transmission time is over.

The node delivers all data packets to the lower layer and waits for ACKs in accordance with the ARQ scheme implemented in UDTN. While the details of the scheme are given in Subsection III-B, here we wish to highlight an inherently cross-layer behavior in UDTN. Namely, the ARQ scheme always informs the routing protocol about the correct reception of the packets by the current destination (in this case, by A) via cross-layer messages. This helps the routing layer understand which messages have already been delivered to a node (in order not transmit the same packets twice), and allows the nodes to remove the packets from the queue if they have been correctly delivered to the final destination.

After B's turn to transmit is over, A starts its own transmissions. It may happen that B does not have σ packets to transmit. To avoid the loss of time allotted to that node, a flag is set in the header of the last packet transmitted. This flag notifies A that B's transmissions are prematurely over, so that A can start transmitting its own packets. If A also does not have σ packets to transmit, it sets the flag in the header like B and moves back to idle state. A special *proxy* packet with this flag set is sent by a node if it has no data to transmit at all. At the end of the two-way data transfer, both A and B move back to an idle state and start transmitting beacons periodically after a backoff time.

One further control procedure is applied during the data transmission process. Namely, if the node sending data packets does not receive ACKs for a given time, it will assume that the receiver has moved out of range, and it will hence stop transmissions. The same happens if the receiver does not receive any data for a given time.

[2]If A receives multiple *info* packets, it selects the node for which the contact duration is expected to be longest among those that exceed the threshold. The only exception to this rule is that the sink is always chosen if an *info* packet is received from it.

[3]The only exception is when A is in contact with the sink: in this case, A will be the only node to transmit, and $T_s = \eta T_c$.

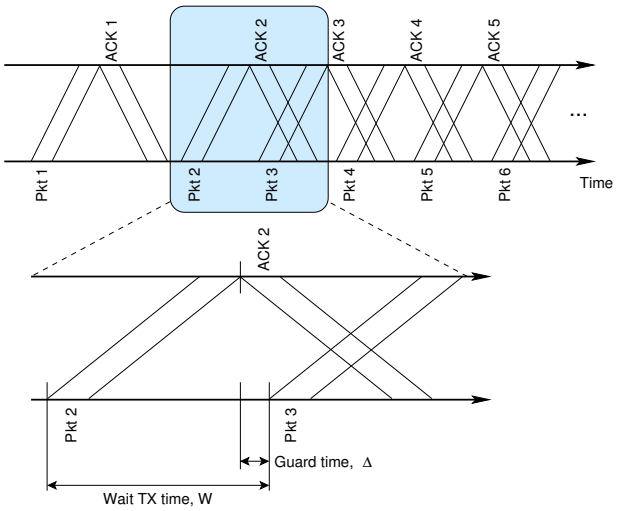

Figure 2. Packet exchange scheme employed in the USR protocol (reproduced from [5]).

B. Modified Underwater Selective Repeat (USR)

USR is an ARQ scheme based on the selective repeat (SR) technique [5]. USR allows underwater nodes to exploit the usually long underwater propagation delays by performing multiple data transmissions within one round-trip time. The transmissions are spaced so that the data transmitter is never deaf to incoming ACKs related to previously transmitted packets. USR typically behaves according to the SR technique, but reverts to plain Stop&Wait (S&W) ARQ i) when there is only one packet to transmit in the transmitter's buffer, ii) when the distance between the source and the destination does not allow the sender to transmit multiple packets in the same round-trip time (RTT) while still respecting the timings of the protocol, and iii) for transmitting the initial data packet in a sequence, which is used to determine whether to employ the S&W or the SR technique. When a node receives the ACK for its initial packet, it calculates the RTT between itself and the receiver. If the RTT is long enough to accommodate more than one packet transmission, the node computes the number of packets that fit within the RTT and starts transmitting so that the reception of ACKs will be interlaced in time with the packet transmissions. This technique is shown in Fig. 2 [5].

Using again A and B to denote the communicating nodes, the number of packets (or "window size") M that B can transmit to A before having to wait for an ACK can be found as

$$M = \max\left(1, \left\lfloor \frac{k\,\tau_{AB}}{T_D + T_A + \Delta} \right\rfloor\right) \tag{5}$$

where T_D and T_A are the transmission time of a data packet and of an ACK packet, respectively, τ_{AB} is one-way propagation delay between A and B, and $0 \leq k \leq 2$ is a tunable parameter, which specifies the portion of τ_{AB} to be considered in the calculation. If $k = 0$, USR always employs S&W ARQ (window size equal to 1), whereas for $k = 2$ the whole RTT will be considered when computing the window size. In (5), Δ is a guard time, which is required to compensate for mobility-induced changes of the RTT.

Since the window size M depends on the RTT, every source-destination pair computes a different window sizes, in general: the USR protocol automatically adapts to the window size thanks to the preliminary S&W round and to the corresponding derivation of M in (5). When $M > 1$, B injects more than one packet in the channel before waiting for an ACK from the receiver A. In order to avoid being deaf to ACK receptions, B waits for a fixed waiting time W before sending the next packet. W is calculated in such a way that if A transmits an ACK right after the reception of a data packet from B is completed, B will receive the ACK in the middle of the waiting period W. By imposing these constraints, we have

$$W = \frac{2(T_A + 2\tau_{AB} - (M-1)T_D)}{2M - 1}. \tag{6}$$

It should be noted that if the nodes are mobile, as is the case in an underwater DTN, the estimation of the RTT should take this aspect into account so that, e.g., packet transmissions do not superimpose to ACK receptions as nodes move. To achieve this, it is sufficient to assume, as a worst case, that the nodes are moving towards each other, i.e., that the RTT is progressively reduced as time goes by. In particular, assume that a node A met with node B at time t_1, and inferred the propagation delay τ_{AB} between the two nodes using the timing of USR's control messages. Assuming that the speed of sound underwater is constant and equal to c, the distance $d_{AB} = c\,\tau_{AB}$ among the nodes can also be computed. To take mobility into account, at any later time t_2, A can compute the distance between the nodes as

$$d'_{AB} = d_{AB} - N_M(t_2 - t_1)v, \tag{7}$$

where v is the maximum speed of the mobile nodes, and N_M can be 0 (if the sender and receiver are both static), 1 (if only one of the two nodes moves), or 2 (if both move). The updated propagation delay $\tau'_{AB} = d'_{AB}/c$ can also be obtained and substituted in (5) in order to compute M. A can repeat the estimation of d'_{AB} at all data packet transmissions performed using USR, and thereby adapt to RTTs that vary over time by adapting the number of packets transmitted within one RTT.

Unlike the original USR technique, the modified version we employ in this paper is aware of the duration of the contact between two nodes employing UDTN, and of the share of this duration that is allotted to each node for transmissions. This knowledge is employed to avoid retransmitting packets indefinitely until all packets are correctly received (as this would likely exceed the time share allotted to a node). On the contrary, all data packets chosen according to the node buffer summaries contained in the *info* and *response* messages are transmitted first, and retransmissions take place only if there is sufficient time left. If one packet is not received correctly after this process, the same packet is retransmitted again at the next meeting between the nodes. A further modification to USR includes the cross-layer message sent to the UDTN routing protocol to notify about the correct delivery of data to

the node in contact. The UDTN can then update the status of the packets in the buffer: e.g., if the receiver was the sink, the packets correctly delivered to it are removed from the buffer.

IV. Simulation scenario and settings

The performance of the UDTN protocol is evaluated using the ns2-Miracle framework [13], along with the World Ocean Simulation System (WOSS) package [14], which provides ns2-Miracle with an interface to the Bellhop ray tracing tool [15]. The location chosen for simulation is an area of $8000\,\mathrm{m} \times 4000\,\mathrm{m}$ in the Mediterranean Sea, whose upper-left corner is placed at $43.0625°N$, $9.3095°E$. Network operations are assumed to take place in July: the corresponding environmental properties are retrieved from the oceanographic databases employed by WOSS.

All nodes transmit using a Binary Phase Shift Keying (BPSK) modulation, at a bit rate of $4800\,\mathrm{bps}$. The size of the *data* packet is fixed to $L_D = 125\,\mathrm{Bytes}$, the ACK size is $L_A = 11\,\mathrm{Bytes}$, the *beacon* size is $L_B = 1\,\mathrm{Byte}$, the minimum length of an *info* packet is $L_I = 58\,\mathrm{Bytes}$ and the minimum length of a *response* packet is $L_R = 11\,\mathrm{Bytes}$. The simulations have been performed for various data generation rates per node, λ, from 0.18 to 6 packets per minute per node. We considered two different numbers of assets to be inspected (hence of AUVs in the DTN), 3 and 6. The nodes (both the assets and the inspecting AUVs) are initially deployed at random within the area. After this, the assets start moving freely around the area according to a Gauss-Markov mobility model [16]; the follower AUVs start moving to approach and follow the assets in accordance with the rules of the mobility model described in Subsection IV-A. A sink is placed near the shore, and represents the transceiver connected to the shore-based control center.

We compare the performance of UDTN against a modified version of the Spray-And-Wait (SAW) [17] protocol. As the original version of SAW [4] assumes reliable packet transfer (which is quite unrealistic for underwater networks) we extended SAW to incorporate a S&W ARQ technique for error control. This is required also for a more fair comparison against the UDTN protocol, which implements ARQ via USR. The messaging pattern of the SAW protocol has also been modified to favor the exchange of fresh information among the nodes. To this end, all nodes periodically transmit beacon messages. Upon receiving a beacon, a node sends a query packet with 40 message ids related to packets that are still to be delivered to the sink. After receiving the query packet, the beacon sender transmits a response packet mentioning which ones of these 40 packets it would like to receive. The requested packets are then sent using a S&W technique. Note that, unlike in UDTN, the communication is not bidirectional, i.e., only the nodes that receive a beacon can transmit. In addition, no adaptation is performed to compensate for time-varying RTTs.

The transmission range we assume in the following is $T_R = 2000\ \mathrm{m}$. In addition, we set $\eta = 0.5$, $k = 2$ and $N_M = 2$. The results are averaged over 100 simulation runs. Each node stops generating packets after $86400\,\mathrm{s}$ or $24\,\mathrm{hours}$ and the simulation ended at $100000\,\mathrm{s}$ or $27.78\,\mathrm{hours}$.

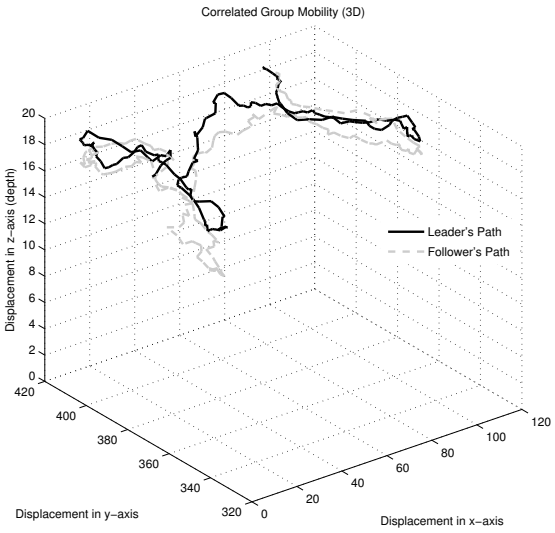

Figure 3. Example of group mobility realization. A "leader node" (representing an asset to be inspected) is moving according to a Gauss-Markov model (black line), and a second node (representing an "inspector AUV") is following the leader (grey line).

A. Details on the mobility model

In order to simulate the trajectories of mobile assets within the network area, we employ the leader-follower paradigm (or "group mobility model") proposed in [18]. The model in [18] was designed for 2D terrestrial radio networks: for this paper, we have extended the movement to take place in a 3D space and to realistically represent an AUV movement (e.g., by forbidding depth changes to be arbitrarily fast).

According to this model, a leader L_i moves either randomly or by following a pre-defined path, and each follower $F_{L_i,j}$ tunes its movement so that it approaches the route of the leader. In our scenario, the leader node represents any surface or underwater asset that enters the coastal area under surveillance, and thereby has to be inspected; followers represent instead the AUVs that approach these assets for performing the inspection. (In the following, we will use the above terms interchangeably.) Note that a follower can follow only one leader, whereas a leader may have several followers. From now on, we will assume that every leader has only one follower, i.e., an asset entering the patrolled area is inspected by only one AUV.

The movement of the followers consists of two components: i) a movement that attracts the follower towards the leader, and ii) a random movement. The attraction is obtained in a way that is similar to the attraction of electrical charges. Therefore, the mobility model has basically three parameters: the charge of the leader, C_L, the charge of the follower, C_F, and a tunable parameter α which is used to tune the intensity of the attraction field. In particular, a negative value of α attracts the

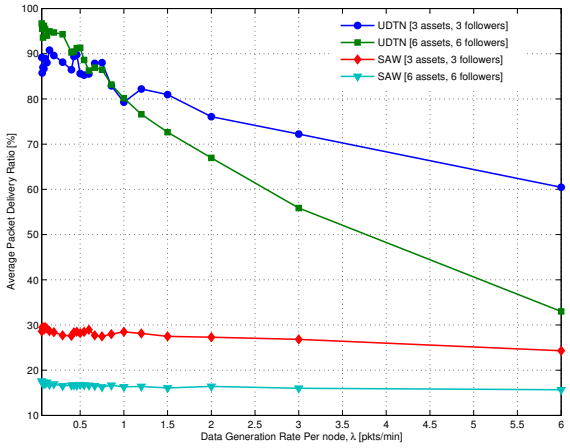

Figure 4. Packet delivery ratio as a function of the data generation rate per node for UDTN and SAW.

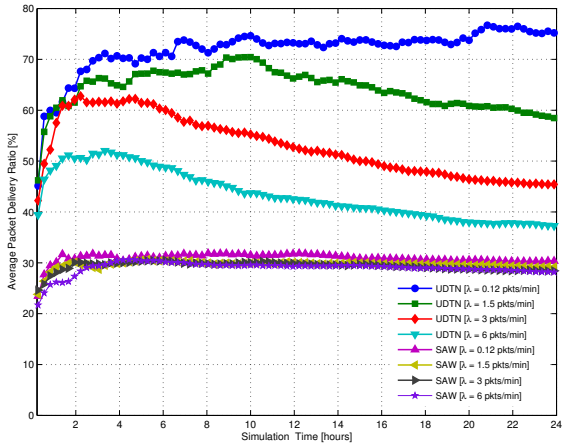

Figure 5. Packet delivery ratio for one network node as a function of the simulation time for UDTN and SAW for a randomly selected node in a scenario with 3 assets and 3 followers, for several values of λ in packets per minute per node.

follower towards the leader, whereas a positive value pushes the follower away. By considering the above parameters one can get the following expression for the attraction velocity $\vec{v}_a$ at time t_k [18]:

$$\vec{v}_a(t_k) = \beta(t_k)\frac{C_L C_F}{d^\alpha}\vec{u}_a \tag{8}$$

where $\vec{u}_a$ is the unit vector directed from the follower towards the leader and $\beta(t_k)$ is the modulus of the attraction speed at a distance $d = 1\ m$, which is calculated as [18]

$$\beta(t_k) = (1-\zeta_a)\beta(t_{k-1}) + \zeta_a s_{a,k} \tag{9}$$

where $0 \leq \zeta_a \leq 1$ is a tunable variable, and $s_{a,k}$ is a Gaussian random variable with mean s_m and standard deviation s_v. Finally, the random velocity $\vec{v}_r(t)$ of the follower's movement is obtained through a Gauss-Markov mobility model [16]. Hence, the speed vector of a given follower at time t can be calculated as

$$\vec{v}(t) = \vec{v}_r(t) + \vec{v}_a(t) \tag{10}$$

Fig. 3 shows an example of leader-follower mobility pattern in a 3D area, obtained using the technique described above, with parameters $C_L = 2$, $C_F = 2$, $\zeta_a = 0.8$, $s_m = 0.02$ and $s_v = 0.005$. These parameters have also been used throughout the simulation campaign.

V. Simulation Results

As a first comparison between UDTN and SAW, we show in Fig. 4 the Packet Delivery Ratio (PDR) of the UDTN and SAW protocols, defined as the ratio of the number of packets delivered to the sink to the number of packets generated in the network. The PDR is shown as a function of the data generation rate per node, λ, for a first configuration with 3 assets and 3 followers, and a second configuration with 6 assets and 6 followers.[4] We also recall that a follower is always assigned to one and only one asset. The first observation is that UDTN outperforms SAW for all chosen values of λ. When the data generation rate is low, the 6-node network experiences a higher packet delivery ratio due to the denser deployment of the nodes, which increases the chance of contact. In turn, this favors the delivery of data to the sink, possibly across multiple intermediate store-and-forward steps. However, the performance of the 6-node network decreases due to greater interference and contention as the packet generation rate per node is increased. For higher values of λ, in fact, the 3-node network experiences a higher packet delivery ratio, which also decreases more smoothly due to the lower interference affecting the communications.

Similar observations hold for SAW, where however the PDR of the 3-node network is always greater than the PDR of the 6-node network. In any event, SAW is consistently outperformed by UDTN. There are several reasons for this. Namely, UDTN allows both nodes to transmit during a contact, whereas in SAW only one node can transmit. Furthermore, the S&W ARQ scheme in SAW introduces delays between subsequent packet transmissions, whereas the modified USR technique employed with UDTN exploits the long RTTs typical of underwater networks to improve the throughput of the data exchange [5]. Furthermore, the packet transmission between communicating nodes in UDTN is adapted based on the computation of T_s, as described in Subsection III-A, whereas no adaptation is performed in SAW.

The evolution of the PDR at different time epochs during the simulation is shown in Fig. 5 for one network node, and for different values of the packet generation rate per node. The PDR is initially low in all cases, as the few contacts that still occur are due to the random deployment of the nodes. The PDR starts increasing as time goes by and subsequent contacts allow the nodes to exchange packets and to deliver

[4]From now on, we will refer to these configurations as the 3-node and the 6-node networks, respectively: this also reflects the fact that only the followers communicate.

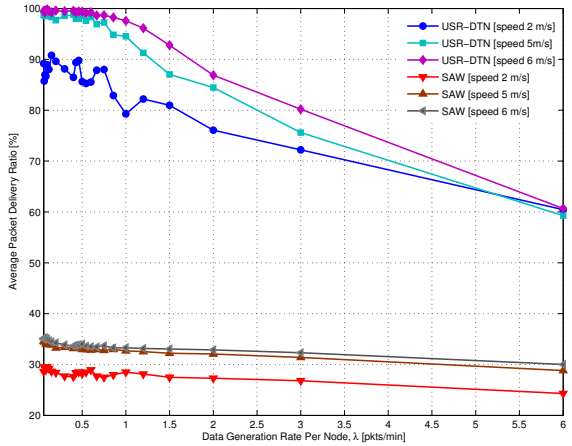

Figure 6. Packet delivery ratio as a function of the packet generation rate per node, λ in packets per minute per node, for several values of the asset and follower speed, in a scenario with 3 assets and 3 followers.

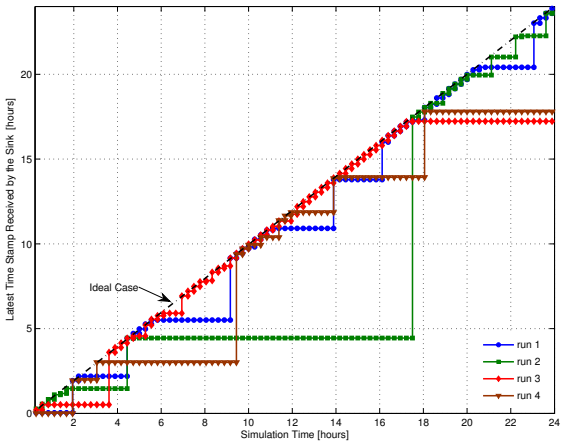

Figure 7. Latest packet received by the sink as a function of the simulation time for UDTN in a scenario with 3 assets and 3 followers, and for $\lambda = 6$ packets per minute per node.

them to the sink using store-and-forward. For UDTN, in the presence of packet generation rates of 1.5 packets per minute per node and above, however, the number of generated packets exceeds the capability of the network to transport the packets to the sink, so that the node buffers build up: eventually, the nodes will drop old packets to free space for newly generated ones. As a result, the PDR decreases towards the end of the simulation. Notably, the PDR of SAW is consistently lower, at about 30% in all configurations, and is constant throughout the simulation. This is due to the lower number of transmissions that take place in SAW, due to both the absence of bidirectional communications within a contact and the use of S&W. Fig. 5 can also be used to understand the maximum packet generation rate per node that is sustainable in the network. Namely, if the curves keep increasing or are stable with the simulation time, we can infer that the network can correctly convey all generated traffic to the sink. In this 3-node network case, the maximum sustainable packet generation rate is between 0.12 and 1.5 packets per minute per node.

Fig. 6 shows the PDR of UDTN and SAW in a 3-node network for three values of the speed of the assets and of the follower AUVs. Since a higher speed translates into more frequent contacts, the PDR increases with increasing node speed for both protocols. In particular, for lower values of λ, the PDR of UDTN reaches a PDR close to 100% for speeds of 5 and 6 m/s, which is around 10–15% higher than that the PDR at 2 m/s. Some improvement in the PDR is also observed for intermediate values of λ. For higher values, the shorter duration of the contacts induced by the higher speeds (and the consequent lower number of exchanged packets per contact) dominates, making the PDR equivalent to that of the lowest speed case.

Finally, we recall that in the coastal surveillance application considered in this paper, one of the objectives is to deliver timely information to the sink. In particular, we want to measure how often the sink is updated by the network with new data about the assets being followed. The evolution of the latest timestamp of the data delivered to the sink as a function of the simulation time is shown in Fig. 7. We set $\lambda = 3$ packets per minute per node

From the figure we can observe that the sink is typically up to date with new information. If long periods of "starvation" occur, it is a mobility issue (no node is in contact with the sink for a long time), not a problem of the UDTN protocol. On the contrary, the capability of UDTN to convey data to the sink via opportunistic transmissions, perhaps through multiple store-and-forward steps, effectively keeps the sink up to date when nodes are in range, as seen from the cases when the curves in Fig. 7 closely follow the ideal case line, and from the packet delivery ratio values in Fig. 5.

VI. Conclusion

In this paper, we proposed UDTN, an underwater DTN routing protocol suitable for monitoring the activity of mobile coastal surveillance networks. In particular, we considered a scenario where several AUVs patrol a given area, and move to inspect any assets entering the area under surveillance. They exploit movements to opportunistically exchange packets when two nodes are within the communication range of each other, with the ultimate objective to deliver data to a shore-based sink in a timely manner. UDTN entails some simple criteria to make the communication between the network nodes effective and fair; in addition, it is embedded with an ARQ technique based on selective repeat, that makes error recovery efficient.

We evaluated UDTN against a DTN routing protocol called Spray-And Wait (SAW), modified to also perform error control. Our results demonstrate that UDTN performs significantly better than SAW, by achieving better PDR and timely delivery of data to the sink for various values of the scenario parameters.

ACKNOWLEDGEMENT

This work has been supported in part by the Italian Institute of Technology within the Project SEED framework (NAUTILUS project), and by Whitehead Alenia Sistemi Subacquei, Livorno, Italy, and the European Defence Agency, under the RACUN project.

The authors would like to thank Prof. Simin Nadjm-Tehrani for sharing an ns2 version of the Spray-and-Wait protocol.

REFERENCES

[1] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "DTN routing as a resource allocation problem," in *Proc. of ACM SIGCOMM*, Kyoto, Japan, Aug. 2007, pp. 373–384.

[2] N. Nicolaou, A. See, P. Xie, J.-H. Cui, and D. Maggiorini, "Improving the robustness of location-beased routing for underweater sensor networks," in *Proc. of IEEE Oceans - Europe*, 2007, pp. 1–6.

[3] J. M. Jornet, M. Stojanovic, and M. Zorzi, "Focused beam routing protocol for underwater acoustic," in *Proc. of ACM WUWNeT*, New York, USA, 2008, pp. 75–88.

[4] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and Wait: an efficient routing scheme for intermittently connected mobile networks," in *Proc. of ACM WDTN*, Philadelphia, PA, Aug. 2005, pp. 252–259.

[5] S. Azad, P. Casari, and M. Zorzi, "On ARQ strategies over random access protocols in underwater acoustic networks," in *Proc. of IEEE/OES Oceans*, Santander, Spain, May 2011.

[6] S. Jain, K. Fall, and R. Patra, "Routing in a delay-tolerant network," in *Proc. of ACM SIGCOMM*, Portland, OR, 2004, pp. 145–157.

[7] D. Henriksson, T. F. Abdelzaher, and R. K. Ganti, "A caching-based approach to routing in delay-tolerant networks," in *Proc. of ICCCN*, Honolulu, HI, Aug. 2007, pp. 69–74.

[8] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Department of Computer Science, Duke University, Tech. Rep. CS-2000-06, Apr. 2000.

[9] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *ACM Mobile Comput. and Commun. Review*, vol. 7, no. 3, pp. 19–20, Jul. 2003.

[10] "Aquacomm: Underwater wireless modem." [Online]. Available: http://www.dspcomm.com/products_aquacomm.html

[11] "Underwater acoustic modem models." [Online]. Available: http://www.link-quest.com/html/models1.html

[12] "S2C R 48/78 Underwater Acoustic Modem." [Online]. Available: http://www.evologics.de/en/products/acoustics/s2cr_48_78.html

[13] N. Baldo, F. Maguolo, M. Miozzo, M. Rossi, and M. Zorzi, "NS2-MIRACLE: a modular framework for multi-technology and cross-layer support in network simulator 2," in *Proc. of ACM NSTools*, Nantes, France, Oct. 2007.

[14] F. Guerra, P. Casari, and M. Zorzi, "World Ocean Simulation System (WOSS): a simulation tool for underwater networks with realistic propagation modeling," in *Proc. of ACM WUWNet 2009*, Berkeley, CA, Nov. 2009.

[15] M. Porter *et al.*, "Bellhop code." [Online]. Available: http://oalib.hlsresearch.com/Rays/index.html

[16] B. Liang and Z. J. Haas, "Predictive distance-based mobility management for PCS networks," in *Proc. of IEEE INFOCOM*, New York, NY, Mar. 1999, pp. 1377–1384.

[17] E. Kuiper, "Details on packet level design for two delay-tolerant routing portocols." [Online]. Available: liu.diva-portal.org/smash/get/diva2:359919/FULLTEXT01

[18] M. Rossi, L. Badia, N. Bui, and M. Zorzi, "On group mobility patterns and their exploitation to logically aggregate terminals in wireless networks," in *IEEE VTC Fall*, Dallas, TX, US, September 2005.