Passive Gateway Election Mechanisms for Swarms of Drones in Aquatic Sensing Environments

João Patrício*, Miguel Luís*†, Susana Sargento*‡
*Instituto de Telecomunicações, 3810-193 Aveiro, Portugal
†ISEL - Instituto Superior de Engenharia de Lisboa, 1959-007 Lisboa, Portugal
‡DETI / University of Aveiro, 3810-193 Aveiro, Portugal

***Abstract*—Monitoring an aquatic sensing environment with multi-swarms of drones requires a reliable communication between the drones for both navigation and data gathering tasks. This work proposes mechanisms for cluster formation in multi-swarms of drones, allowing the aquatic drones to select, in a distributed way, the gateways of each cluster that will be responsible for forwarding collected data towards the gateway on land. Two gateway election methods were designed and implemented, one dealing with the energy of aquatic drones, and the other considering different metrics such as link quality, centrality and energy. The proposed methods were evaluated across several cases and scenarios, where clusters changed in a dynamic way due to mobility and energy constraints. The obtained results show that the election of gateways with a method based on several metrics, together with an appropriate control strategy, provides a better outcome of the network behaviour throughout the aquatic monitoring tasks.**

***Index Terms*—Clustering, Data Gathering, Link Quality, Betweenness Centrality, Delay Tolerant Networks, Gateway Election**

I. Introduction

Monitoring an aquatic environment using Unmanned Surface Vehicles (USVs) overcomes several previous constrains, such as human effort, access to the information, cost, low efficiency and many others. USVs can autonomously monitor an aquatic environment with both environment and aquatic sensors, making available the information to a monitoring platform through wireless connections from the water to the land.

USVs can form clusters as they move in the water, so that they can self-organize to exchange information and send data to the land. In this cluster environment, USVs send their collected data to a cluster head (CH), also referred as gateway (GW), that gathers this data and further sends it to the Base Station (BS) on land. Due to the network's mobility, the loss of connection to the GW might happen when a USV goes on a mission alone or travels between clusters. Therefore, a GW must be elected within each cluster and, as the network of each cluster changes, the election process must occur over time. Such process may consider a wide range of features and characteristics, such as energy, centrality and many others.

This paper proposes strategies capable of making each USV self-adapt to its cluster, delivering the collected information to the BS through the elected GW, keeping the lowest cost and overhead possible. It proposes GW election mechanisms that take into consideration metrics such as energy of the USVs and the link quality between USVs. The GW election mechanisms proposed allow to extend the lifetime of the network while delivering the data through high quality links. The control strategy showed to be crucial for the network behaviour, since it provided better stability to the network reducing the number of GW elections and preventing the wrong elections of GWs within the cluster.

The remainder of this paper is organized as follows. Section II presents a brief overview of GW election mechanisms in the literature. Section III presents the proposed GW election methods and control strategy. Section IV presents the results of the developed work through several scenarios and use cases. At last, Section V presents the conclusions and directions for future work.

II. Related Work

Network nodes have limited power supply, so the energy is usually a central point when designing clustering protocols. One of the initial protocols focused on energy is the LEACH [1] protocol. LEACH dynamically elects the CH in a round-robin fashion that allows each node in the cluster to become CH in varying rounds. This approach increases the network lifetime, in comparison with direct communication, by balancing the energy consumption of each node. Younis and Fahmy proposed the HEED [2] protocol, which is based on two clustering parameters: residual energy and node degree. This parametric combination leads to a load balancing feature and a more energy efficient protocol. Unfortunately, it presents a few drawbacks, such as additional broadcast packet overhead caused by the CH selection procedures, hotspots problems due to the more workload on CHs, specially the ones near the BS [3], and extraneous CH formation due to uncovered nodes [4].

Approaches were made based on the Fuzzy Logic, a Computational Intelligence technique that can be used in applications with uncertainties [5]. In [6] the authors presented a fuzzy-based protocol, RE-TOPSIS, that uses LEACH logic for the first CH election, and then the fuzzy logic on 6 different factors. The CH election is made by first discovering the neighbors in the network, then each node calculates the rank of their neighbors and itself based on the considered factors. The neighbor with the highest rank announces itself in the network as a CH through broadcast. Then, the neighbors answer with *request joints* to the CH. After a threshold value, a new CH election takes place.

978-1-7281-6587-5/20/$31.00 ©2020 IEEE

Many optimizations of the previously presented protocols have been developed over the years, but most of them tend to focus on the energy consumption and centrality, not taking into account the quality of the connection between the nodes [2], [7]. The majority fail to be robust in scenarios with high mobility and require high processing due to their complexity [8], or have a considerable network overhead and do not consider mobile scenarios as [9].

Our approach considers a mechanism for Cluster Head/Gateway election on a dynamic cluster, done on a passive way with low overhead, and combining multiple metrics that are relevant for aquatic monitoring environments in scenarios with high mobility. Moreover, we consider the possibility to have several dynamic clusters simultaneously, with nodes joining and leaving the clusters in a dynamic approach.

III. Gateway Election

This section describes the proposed approaches in terms of GW election and network control strategy.

A. Clustering

The USVs are usually deployed as a team into an aquatic environment. During the monitoring task, the initial team (or the initial cluster) may be divided in smaller clusters. Clustering the network increases the lifetime and quality of the network [7]. A cluster is a group of nodes that can reach each other through one or more hops, and have elected a cluster head, the GW, to which they send the collected informationThe GW gathers this information and sends it to the BS through a long range technology or a range enough to reach the BS.

B. Neighborhood Awareness

Keeping the awareness of the surrounding nodes within the cluster for routing decisions leads to a more efficient utilization of the cluster's network [10]. The neighborhood graph, along with an energy table containing the current energy of every node in the network, are key for the good operation of the routing process, as many decisions will come based on the information provided by them. Each node will receive the network graph and energy table in every *beacon* transmitted by each node (USV), that is sent periodically as a neighbor announcement. Therefore, this knowledge is obtained through a passive way and can be used for a passive GW election.

C. Gateway Rank Calculation

The passive GW election mechanism takes advantage of the neighborhood adjacency graph and the energy table already spread within the cluster for the rank calculation, allowing each node to have the knowledge of the current network status. Therefore, the nodes can simultaneously elect the same GW every time a new GW is to be elected.

Based on the neighbors' graph weight and energy table stored on each node, where i represents a node, the rank calculation is given by

$$Rank(i) = (1 - C_B(i)) \times LQ_{av} + C_B(i) \times Ener(i), \quad (1)$$

where LQ_{av} is the average link quality between a node and the other nodes in the cluster, which is normalized between 0 and 100. From the weight of each vertex v of the neighbors' graph, the link quality estimation between two nodes is given by $LQ(v) = 100 - Weight(v)$. N is the total number of nodes in the cluster, and LQ_{av} is given by

$$LQ_{av}(i) = \frac{\sum_{n=0}^{N} LQ(n)}{N}. \quad (2)$$

$Ener(i)$ is obtained from the neighbors' energy table, being a percentage value with a minimum of 0 and a maximum of 100, containing the normalized value of energy on the node i. $C_B(i)$ is the betweenness centrality, presented by Ulrik Brandes [11], that quantifies the number of times a node acts as a bridge along the shortest path between two other nodes. The betweenness centrality of a node i is given by

$$C_B(i) = \sum_{s \neq v \neq t \in V} \frac{\sigma_{st}(i)}{\sigma_{st}}, \quad (3)$$

where σ_{st} is the total number of shortest paths from node s to node t, and $\sigma_{st}(i)$ is the number of paths through i. Its output has a minimum value of 0 and a maximum of 1.

This rank equation provides a result derived from the energy and link quality connection, where the weight of each factor in the equation is defined by the betweenness centrality. Therefore, when a node has a higher betweenness centrality, the energy will weight more on the equation. This is due to this node's centrality, as it will probably forward more messages between other nodes and a GW, when compared with a more isolated node, resulting on a higher energy expenditure. On the other hand, if a node as a lower centrality, it has a higher probability of having a poorer link quality connection within the network. Here, the link quality of the connection should have more impact on the decision and on the ranking metric.

D. Gateway Election Process

The GW of a cluster might change overtime, specially under a mobile environment. Therefore, a cluster must be able to elect and re-elect a GW.

For the election of a cluster's GW, two mechanisms are proposed, which will chose a GW to be elected. Then, a control strategy will decide if the proposed GW is, effectively, elected. The first mechanism is based on the energy level of each node, and is similar to the most common GW election methods. It will be used as a benchmark. The second mechanism is based on the ranking metrics previously presented, and represented by (1), which uses not only energy, but also the link quality of connections between the node and its neighbors. It also takes into account the betweenness centrality of the nodes in the cluster. Beyond the two GW election mechanisms, a control strategy is also introduced: it is responsible to control if or when a new GW should be chosen. The control strategy verifies the mechanisms' outputs and decides if their proposed GW should get elected or not.

Algorithm 1: Energy-based mechanism

```
for id = first node to id = last node of the cluster do
    Cb(id) ← get_central_betweenness(id) ;  ▷ C_B from (3)
    Ener(id) = EnergyTable(id) ;
best_id = −1;
highest_energy = 0;
for id = first node to id=last node of the cluster do
    if Ener(id) > highest_energy then
        best_id=id;
    else if Ener(id) == Ener(best_id) and Cb(id) ≠ Cb(best_id) then
        best_id = node id with the highest betweenness centrality;
    else
        best_id = node with the lowest id;
```

Algorithm 2: Rank-based Mechanism

```
for id = first node to id = last node of the cluster do
    cb(id) ← get_central_betweenness(id) ;  ▷ C_B from (3)
    av_lqe(id)= ← get_average_lqe(id) ;  ▷ LQ_av from (2)
    ener(id) = EnergyTable(id) ;
    rank(id) ← get_rank(id, cb(id), av_lqe(id), ener(id))
    ▷ Rank(i) from (1)
best_id = −1;
highest_rank = 0;
for id = first node to id=last node of the cluster do
    if rank(id) > highest_rank then
        best_id = id;
    else if Rank(id) == Rank(best_id) and cb(id) ≠ cb(best_id) then
        best_id = node id with the highest betweenness;
    else
        best_id = node with the lowest id;
```

1) Energy-based Mechanism: The first mechanism goes through all nodes in the cluster and calculates, for each node, its betweenness centrality and energy (Algorithm 1).

Then, it compares its energy (that represents the rank in this mechanism) until it finds the node with the highest energy value. If two nodes have the same energy value, the tiebreaker is done by choosing the node that has the biggest betweenness centrality. If they are tied again, the node with the lowest ID is the chosen one.

This mechanism allows the elected GW to vary frequently, keeping a uniform value of battery within the cluster. It will result in a higher delay, as it will be electing mostly with the same frequency all nodes over time, regardless their poor quality of connection to the neighborhood.

2) Rank-based Mechanism: This mechanism is based on the ranking given by (1). As shown in Algorithm 2, the rationale behind this mechanism is to start by computing the betweenness centrality for each node in the cluster. Then, the average link quality of each node in the cluster to the others is determined, following the rank of every node. Next, it finds the best rank of each node in the cluster. If two nodes end up having the same rank value, the tiebreaker is done by selecting the node with the biggest betweenness centrality. If they are once again tied, it is chosen the node with the lowest ID.

This mechanism allows the elected GW to vary less frequently: a major difference of battery levels within the cluster is expected, specially from nodes with higher centrality. A lower delivery delay is expected, as it will be electing nodes with better link connection to the neighborhood.

3) Control Strategy: A periodic GW check up event is scheduled to occur every *gwCheckPeriod* seconds. The schedule is made after the election of a new GW, or after a previous check that has not scheduled any other event. The control strategy aims to control the inequality and frequency of the GW proposed for election by the mechanisms; a similar control strategy is proposed to be executed when there is no elected GW in the cluster.

In a base approach, if a new GW is proposed, then it just checks its address and directly elects it without any additional control, scheduling at the end the periodic GW check event.

On the control strategy (Algorithm 3), when a new GW is proposed to be elected, it is rechecked a defined number of times that this node must be proposed as GW in a row ($control\_count$). This is done by saving the new proposed GW and scheduling a recheck up event in $control\_time$ seconds to verify if it is proposed again. After suppressing the $control\_count$ value, if the rank of the proposed GW is at least 10% better than the rank of the current GW, then the proposed new GW is the elected GW and the periodic GW check event is set. This control prevents an uncontrolled change of the GW overtime. It also avoids situations where nodes elect different GWs, as this control strategy makes the node recalculate the proposed GW a few times before concluding that it is the best choice. This recheck also gives time for the node to receive new information about the neighborhood, decreasing the chance of wrong GW elections by the nodes.

There is an additional control strategy that the GW election process goes through when there is no GW elected. Not having an elected GW can occur at the beginning, when the nodes are deployed without a predefined GW, or when the cluster suffers a change on its elements, that is, a node or more leaves or enters the cluster. Such action forces a deletion of the current GW on each node, so it triggers a new GW calculation and election, taking into account the new state of the cluster. This control strategy rechecks if the chosen GW is proposed more than $control\_countP$ times in a row, scheduling this recheck

Algorithm 3: Control Strategy

```
count_new_ElectGw = 0;
if exists a current GW and the proposed GW ≠ current GW then
    if new GW == GW proposed on the last check then
        count_new_ElectGw + +;
    else
        count_new_ElectGw = 0;
    if count_new_ElectGw > control_count then
        count_new_ElectGw = 0;
        if rank of new GW > 0.9×rank of current GW then
            elect new GW as current GW;
            schedule next periodic GW check up event;
    else
        save new GW as GW proposed by this check up;
        schedule a new GW recheck up event in
         control_time seconds;
else
    schedule next periodic GW check up event in
     gwCheckPeriod seconds;
```

in periods of $control\_timeP$ seconds. After overcoming these steps, it elects the proposed GW. These control parameters are defined in the configuration of the mechanism, and also prevent the election of different GWs in the same cluster by its nodes.

E. Gateway Election Process Flow

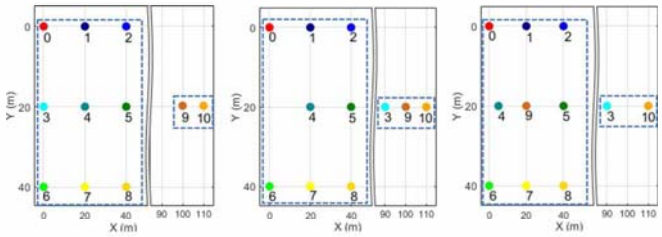
The process of electing a GW starts by checking if the node is alone in the cluster or if it has neighbors. If it is alone, it rechecks this state five times in a row, separated by *gwCheckPeriod* seconds each, until it elects itself as a GW. However, if it has neighbors, it calculates the rank for each node. Then, accordingly to the cluster's state, it enters on the control strategy. After a new GW is elected, the periodic GW check event is schedule in *gwCheckPeriod* seconds.

The control counters, times and $gwCheckPeriod$ are parameters of the control strategy and are defined at the beginning. They can be changed for an optimization of the proposed mechanisms and control strategy.

IV. Network Evaluation

This section evaluates the GW election mechanisms in two distinct scenarios (A and B), with different types of mobility and recreating potential aquatic sensing scenarios.

The topology of scenario A is illustrated in Figure 1: 11 USVs are divided in two clusters represented by the blue dotted squares. Initially, there are 9 USVs in one cluster and 2 in another one (Figure 1a). In the first cluster, the 9 USVs are separated from each other 20 meters, and in the second cluster, the USV 9 is 10 meters away from USV 10. After 20 seconds, the USV 3 moves in direction to the second cluster and reaches it at 40 seconds, as illustrated in Figure 1b. Then, at 70 seconds, USV 4 moves to the left within the cluster, and the USV 9 gets out of its cluster and joins the first cluster. Both USVs stop their movement at 80 seconds of simulation run time, as illustrated in Figure 1c. The BS is located 500m into land from where the USVs were deployed.



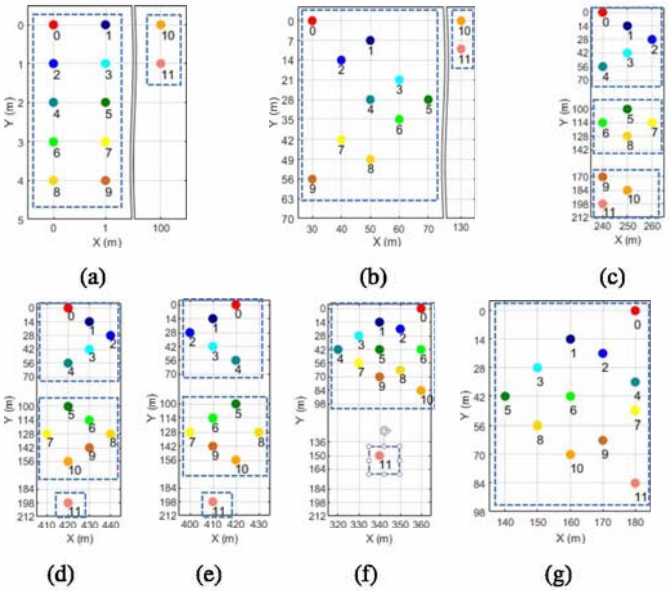
(a) $T_0 = 0\ s.$ (b) $T_1 = 40\ s.$ (c) $T_2 = 80\ s.$

Fig. 1: Aquatic scenario A.

Scenario B mimics a more real aquatic monitoring task in different areas. Its topology is represented in Figure 2 through several time-sketches. The dotted line surrounds the USVs of each cluster. This scenario is made of 12 USVs that are initially grouped in two clusters, one with 10 USVs, and another with 2 USVs deployed 100 meters away, as shown in Figure 2a). All USVs move with an average speed of 1 m/s. After deployment at 30 seconds, the USVs are formed as shown in Figure 2b), and after two minutes they change to the formation represented in Figure 2c). The formation keeps changing overtime as illustrated by the remaining sub-figures of Figure 2. The BS is located $400m$ into land from where the USVs were deployed initially.

The evaluation process took place using a modified version of OMNeT++[1] along with the INET framework, a well known discrete event network emulator. The modifications were

[1]https://omnetpp.org



(a) (b) (c)

(d) (e) (f) (g)

Fig. 2: Aquatic scenario B: a) $T_0 = 0s$, b) $T_1 = 30s$, c) $T_2 = 240s$, d) $T_3 = 420s$, e) $T_4 = 520s$, f) $T_5 = 600s$, g) $T_6 = 780s$.

needed in order to create a Delay Tolerant Network environment embracing the modifications previously presented. As a result, an Aquatic Surface Delay Tolerant Networks Simulator (AquaticDTNS) was created. All USVs were equipped with a Wi-Fi wireless interface with the range set to $40m$, for USV-USV communication, along with a cellular interface for communication with the BS on land to dispatch the monitoring information when elected as GW. The initial energy to be used by the communication module was set to $100J$.

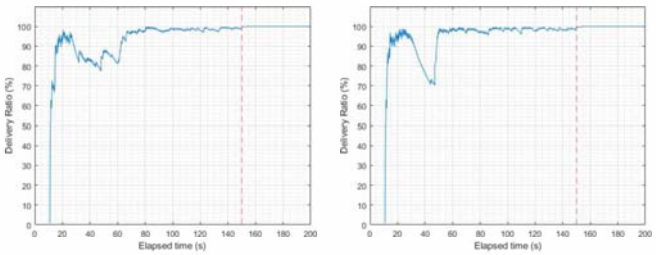
A. *Evaluating different election and control strategies*

Scenario A is used to evaluate the two proposed GW election mechanisms and control strategy, where the first mechanism is referred to as $M1$ and the second to $M2$. The non-existence of control strategy is referred to $C1$, and the control strategy to $C2$. The simulation total time is $200s$, where every USV generates data messages during the first $150s$. The the control strategies were defined with $control\_count = 2$, $control\_time = 2$, $control\_countP = 5$ and $control\_timeP = 1$.

The non-existence of a control strategy results on a high number of GW election, as shown in Table I. The results about the delivery ratio over time, illustrated in Figure 3, show an unstable delivery ratio on the periods with movement. In Figure 4 we can notice that there are GWs elected simultaneously within the same cluster, or even the lack of a GW elected due to the chaotic change of GWs. Such behavior also results on a high number of total data hops and redundant overhead, as the data messages are transmitted on the wrong direction unnecessary.

TABLE I: Results for scenario A.

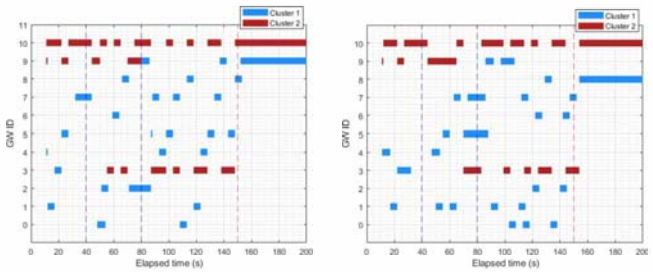
	M1C1	M2C1	M1C2	M2C2
Simulation events	783167	764348	753482	742298
Redundant overhead [pkts]	210	137	33	14
Mean delay [s]	2.6143	3.2645	1.4379	1.4205
Total data hops	4497	2949	2085	1606
GW elections	44	39	12	7
Battery difference [%]	4	24	12	35



(a) Mechanism 1. (b) Mechanism 2.

Fig. 3: Delivery ratio in scenario A with no control strategy.

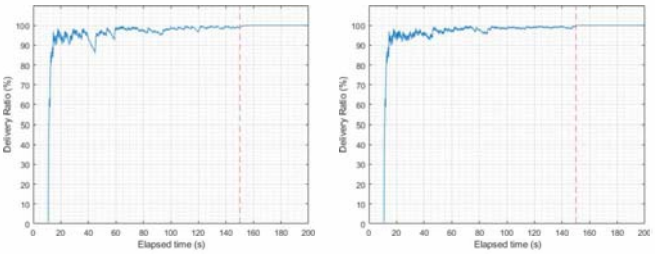
With the control strategy, the overall behaviour is improved. In Figures 5 and 6, it can be observed that the first mechanism still presents a slight worst delivery ratio and a higher variation of the elected GW during time, as expected. The results in



(a) Mechanism 1. (b) Mechanism 2.

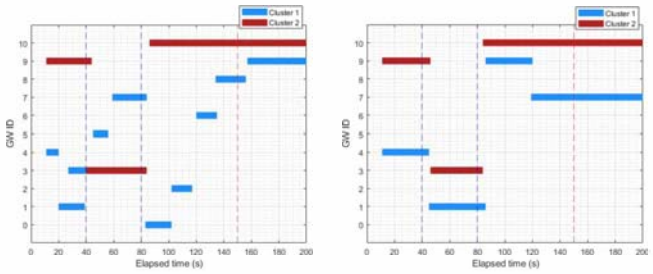
Fig. 4: Gateways elected in scenario A with no control strategy.

Table I show that, even though both mechanisms present a similar mean delay, the first mechanism has a higher number of total data hops. This was expected, since the first mechanism does not take into account the highest betweenness centrality and quality of the connection, ending up electing USVs that are further away than the majority of USVs.



(a) Mechanism 1. (b) Mechanism 2.

Fig. 5: Delivery ratio in scenario A with control strategy.



(a) Mechanism 1. (b) Mechanism 2.

Fig. 6: Gateways elected in scenario A with control strategy.

In Figure 6a we can also observe that there are two elected GWs in cluster 1 from around the 30 to the 40 seconds, and between the new election of GWs there is a gap of a few seconds without an elected GW. Such occurs because the USV 3 moves around that period of time, changing its centrality that is used to break the tie for the same battery of other USVs, and this information can take longer to arrive to other USVs than the time that the control strategy takes to elect a new GW. The variation of the parameters related with the operation mode of the control strategy is evaluated in the next section.

Overall, the control strategy has a much better performance than the non-existence of a GW election control, providing a more stable election of the GW and, consequently, a more stable network. Between the two mechanisms, the second one presents an overall better performance due to its additional parameters that lead to a better proposal of GWs election.

B. Evaluating the impact of each control parameter

In this section, we evaluate the performance of the second mechanism - rank based - along with the control strategy when there is not a previous GW election, described in Section III, for different values of control variables. From the several parameters, we assess the impact of *control_count* and *control_countP*, creating 4 different cases as presented in Table II.

TABLE II: Different parameters for mechanism 2 with control strategy.

	Case 1	Case 2	Case 3	Case 4
control_count	1	6	10	15
control_time [s]	2	2	2	2
control_countP	3	5	20	60
control_timeP [s]	1	1	1	1

Table III presents the results for each case when evaluated in Scenario B. Each case was tested in 10 simulations and the average value is presented along with the 95% confidence interval. We can observe that low control values result in a higher number of GW elections, creating situations with more than one GW in the cluster. Such situation adds confusion within the network and results in a higher number of hops travelled by a packet, as data messages are travelling back and forward in the network until a common GW is elected. Having more data messages going through the network also increases the redundant overhead, as the loss of acknowledgment packets also increases. For higher values of *control_count* and *control_countP*, there is a higher delivery ratio at the end of the simulation, but also a higher delay and a higher unevenness among the USV's battery. An overall better behaviour was observed with the parameters used in case 2, where results are fair and have a far better confidence interval than the other cases, which indicate a more stable behaviour.

TABLE III: Results for scenario B, method 2 and third control strategy in different cases.

	Case 1	Case 2	Case 3	Case 4
End delivery ratio [%]	95.11 ±3.40	97.90 ±2.81	99.73 ±0.31	99.92 ±0.15
Redundant overhead [pkts]	270 ±80	109 ±32	134 ±23	459 ±376
Mean delay [s]	5.29 ±1.54	3.57 ±0.34	5.84 ±0.81	16.60 ±4.98
Total data hops	17717 ±6289	11906 ±862	14407 ±2648	11588 ±1809
GW elections	22 ±1	18 ±1	14 ±1	11 ±1
Battery difference [%]	50 ±4	65 ±9	62 ±8	75 ±9

V. Conclusions

This paper proposed a novel approach for gateway election in swarms of drones in aquatic sensing environments. Different election mechanisms, based on different metrics such as betweenness centrality and energy consumption, were designed and complemented with control strategies in order to control the GW election process. Based on simulation results, the first mechanism showed that the USVs ended the monitoring task with a similar battery level, while with the second mechanism, they presented uneven levels of battery. However, the second mechanism, which considers both energy and link quality, presented a better overall performance, as it achieved a lower mean delay and a better ratio on the packet delivery, with a lower number of hops by the data packets. The control strategy has proved to be crucial for the network behaviour, providing more stability, reducing the total number of GW elections, and preventing the wrong elections of GWs within the cluster. In the end, different control values were evaluated, showing how they impact in the performance of the GW election scheme, and consequently, on the network behavior.

Future work will address the real deployment of USVs in a real monitoring aquatic environment.

Acknowledgment

This work was funded by the European Regional Development Fund (FEDER), through the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 and National Financial Support Public (FCT/OE), project MobiWise (POCI-01-0145-FEDER-016426), and through the Programa Integrado de IC&DT Centro2020, project SmartBioR (Centro-01-0145-FEDER-000018).

References

[1] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, vol. 2, Jan 2000, p. 10.

[2] O. Younis and S. Fahmy, "Distributed clustering in ad-hoc sensor networks: a hybrid, energy-efficient approach," in *IEEE INFOCOM 2004*, vol. 1, March 2004, p. 640.

[3] D. Wei, Y. Jin, S. Vural, K. Moessner, and R. Tafazolli, "An energy-efficient clustering solution for wireless sensor networks," *IEEE Transactions on Wireless Communications*, vol. 10, no. 11, pp. 3973–3983, November 2011.

[4] N. Aslam, W. Phillips, W. Robertson, and S. Sivakumar, "A multi-criterion optimization technique for energy efficient cluster formation in wireless sensor networks," *Information Fusion*, vol. 12, no. 3, pp. 202–212, 2011, special Issue on Information Fusion in Future Generation Communication Environments.

[5] M. Negnevitsky, *Artificial Intelligence: A Guide to Intelligent Systems*. Addison-Wesley, 2005.

[6] S. Murugaanandam and V. Ganapathy, "Reliability-based cluster head selection methodology using fuzzy logic for performance improvement in wsns," *IEEE Access*, vol. 7, pp. 87 357–87 368, 2019.

[7] N. Aierken, R. Gagliardi, L. Mostarda, and Z. Ullah, "RUHEED-Rotated Unequal Clustering Algorithm for Wireless Sensor Networks," in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*, March 2015, pp. 170–174.

[8] P. S. Mehra, M. N. Doja, and B. Alam, "Fuzzy based enhanced cluster head selection (fbecs) for wsn," *Journal of King Saud University - Science*, vol. 32, no. 1, pp. 390–401, 2020.

[9] P. Gupta and A. Sharma, "Clustering-based heterogeneous optimized-HEED protocols for WSNs," *Soft Computing*, vol. 24, p. 1737–1761, February 2020.

[10] D. Sousa, M. Luís, S. Sargento, and A. Pereira, "An Aquatic Mobile Sensing USV Swarm with a Link Quality-Based Delay Tolerant Network," *Sensors*, vol. 18, no. 10, 2018.

[11] U. Brandes, "A faster algorithm for betweenness centrality," *The Journal of Mathematical Sociology*, vol. 25, no. 2, pp. 163–177, 2001.