IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 28, 2021, accepted October 29, 2021, date of publication November 8, 2021,
date of current version November 18, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3126107

A Balanced Routing Protocol Based on Machine Learning for Underwater Sensor Networks

L. ALSALMAN AND E. ALOTAIBI
Computer Engineering Department, Kuwait University, Kuwait City 12037, Kuwait
Corresponding author: E. Alotaibi (e.alotaibi@ku.edu.kw)

This work was supported by Kuwait University under Project EO04/19.

ABSTRACT An underwater sensor network (UWSN) is a wireless network that is deployed in oceans, seas, and rivers for real-time exploration of environmental conditions. The network is used to measure temperature, pressure, water pollution, oxygen level, volcanic activity, floods, and water streams. Although radio frequency (RF) is widely utilized in wireless networks, it is incompatible with the UWSN environment; therefore, other communication mechanisms have been employed to manage the underwater wireless communication among sensors, such as acoustic channels, optical waves, or magnetic induction (MI). Unlike terrestrial wireless sensor networks, UWSNs are dynamic, and sensors move according to water activity. Therefore, the network topology changes rapidly. One of the most critical challenges in UWSNs is how to collect and route the sensed data from the distributed sensors to the sink node. Unfortunately, the direct application of efficient and well-established terrestrial routing protocols is not possible in UWSNs. In this work, a balanced routing protocol based on machine learning for underwater sensor networks (BRP-ML) is proposed that considers the UWSN environmental characteristics, such as power limitations and latency, while considering the void area issue. It is based on reinforcement learning (Q-learning), which aims to reduce the network latency and energy consumption of UWSNs. The communication technique in the proposed protocol is based on the MI technique, which has many advantages, such as steady and predictable channel response and low signal propagation delay. The simulation findings validated that BRP-ML reduced latency by 18% and increased energy efficiency by 16% compared to QELAR.

INDEX TERMS Underwater sensor network, routing protocol, reinforcement learning, network lifetime.

I. INTRODUCTION

Underwater sensor networks (UWSNs) have recently attracted industry and research community attention due to their broad application areas, such as resource discovery, disaster avoidance, auxiliary navigation, and military purposes. Actually, underwater acoustics is not new. It was first studied in the early 1800s, and the first practical application was in the early 1900s [1]. It was used on ships to increase navigational safety and receive bell signals. Then, between the 1920s and 1930s, researchers started understanding the basic concepts of underwater sound. By the Second World War, the United States had equipped its ships with communication systems to determine seabed depth and detect objects miles away [2].

Consequently, the necessity of using UWSNs has emerged. A UWSN is a wireless network that utilizes a set of sensors and autonomous underwater vehicles (AUVs) to collect and sense data underwater. These underwater sensors deliver the data to a surface sensor (sink node) directly or indirectly. Then, the data are transmitted from the sink nodes to an offshore monitoring center (base station) for analysis and study of the collected data. UWSNs are a hot research area because of their numerous applications. A practical underwater acquisition technique must be studied with the increasing demands for environmental marine surveillance, exploration of marine resources, scientific marine research, and marine protection. UWSNs have become a popular research field due to their significance to the community. These networks can be applied in different fields, such as the following:

1) Monitoring: UWSNs can be used to monitor the quality of water. [3] and [4] proposed implementing sensor networks to monitor water quality. Additionally, UWSNs are used for exploration purposes such as natural resources. The marine life environment can also be monitored by UWSNs.

The associate editor coordinating the review of this manuscript and approving it for publication was Resul Das.

2) Disaster prediction: UWSNs can be used to predict floods, volcanic activity, and earthquakes.

3) Military and surveillance applications: UWSNs can be used to secure port facilities, communicate with submarines, and detect mines [5].

4) Underwater localization: UWSNs can provides navigation assistance to vessels and explorers [6].

5) Underwater sports: UWSNs can be used for sensing parameters of swimmers such as velocity, which is utilized for further analysis.

Underwater environments are known to be harsh environments due to several factors, such as water movement, varying temperatures, and salinity [7]. Since establishing a connection through a UWSN is a challenging task, researchers have used other communication technologies for UWSNs, such as optical waves, acoustic waves, and electromagnetic (EM) waves. Each communication approach has different characteristics; therefore, choosing the appropriate technology is critical in creating an efficient UWSN [8]. Acoustic communication is the most popular technique; however, this method has several shortcomings, such as long transmission delays, strongly dependent channel behavior on the environment, and low data rates.

Marine studies motivated scientists to research UWSNs. However, many challenges face this type of network. The speed of sound underwater is 1500 m/s, which is much slower than that of regular radio waves, which causes large propagation delays [9]. In addition, the underwater environment is dynamic, which complicates the network topology. The sensors are operated by batteries that are hard to recharge or replace. Thus, energy consumption must be managed to extend the network lifetime. Furthermore, there are other challenges that face UWSNs, such as routing, limited bandwidth, high error rate, and node localization [7]. Additionally, the void region issue is a primary concern that must be considered when designing a routing protocol. A void region is a region where nodes have drained their batteries or that lacks any nodes at all. These challenges make it difficult to apply typical terrestrial sensor network protocols in UWSNs.

In this paper, a balanced routing protocol based on machine learning for UWSNs (BRP-ML) is proposed, which works in four phases: the initialization phase, discovery phase, clustering phase, and data forwarding phase. Q-learning is used in the data forwarding phase, where routing is performed. Q-learning is a model-free reinforcement learning algorithm that approximates the optimal policy without knowing the model [10]. Moreover, within the clustering phase, another unsupervised learning algorithm is used to divide the nodes into clusters. Clustering extends the network lifetime and balances the energy consumption. A modified version of K-means is applied to the 3D network to cluster the nodes. K-means is a partitioning clustering algorithm. The magnetic induction (MI) technique is used as a communication method due to its benefits.

The rest of this paper is organized as follows: section II conducts a literature review of existing routing protocols designed for UWSNs. Section III provides a theoretical background of machine learning, including reinforcement learning, the Markov decision process (MDP), and Q-learning. In addition, it provides a review of underwater communication methods and energy models. Section IV describes the details of the BRP-ML methodology, including the clustering and routing processes, policy, and reward function. Section V discusses the simulation results and analysis performed to test the approach. It also includes the results of comparing our approach with other routing protocols. Section VI concludes the paper and suggests future work.

II. LITERATURE REVIEW

Recently, UWSNs have attracted the attention of researchers due to the widespread demand for understanding the underwater environment. In [11], a Q-learning-based routing protocol (QELAR) was suggested to extend the lifetime of acoustic UWSNs. In this protocol, each node exchanges its metadata with neighboring nodes. When a packet is sent, the node attaches the metadata with the packet. Then, other nodes can overhear the traffic, extract the metadata, and drop the packet if it is not the eligible forwarder. If the receiver node is the eligible forwarder, it will calculate the Q-value and select the next forwarder. The QELAR reward function depends on the residual energy only, which means that it will always choose nodes with the highest residual energy regardless of the delay cost. Therefore, if the number of nodes increases, longer paths with a greater number of hops will be created, which causes more delay and energy consumption. This protocol also provides a mechanism to detect transmission failure.

[12] proposed a machine learning (QDAR)-based routing algorithm to extend the lifetime of UWSNs while considering the delivery latency. Two types of packet structures are used in QDAR: the data-ready packet and the interest packet. The data-ready packet is transmitted from the source node to the sink node containing the node's necessary information. Moreover, the interest packet is transmitted from the sink node to the source node to determine the routing path. The algorithm consists of five phases. The first phase is the data-ready phase, where the node's data are collected to plan the routing path, which is where the data-ready packet is sent. The source node sends a broadcast to the sink node requesting a routing path. The second phase is the routing decision phase, where the QDAR algorithm determines the routing path. The third phase is the interest phase, where the sink node sends the interest packet to the source node following the output path of the QDAR algorithm. The fourth phase is the package forwarding phase, where data are sent from the source node to the sink node according to the set path. The final phase is the acknowledgment phase, where path reliability is checked. If the source node does not receive an acknowledgment (ACK) message, it returns to the first

phase. Otherwise, it loops between the fourth and fifth phases. Once the path is determined, the following packets coming from the same node follow the same path until the path fails. In the simulation, the latency was reduced compared with other routing protocols, but the network lifetime was reduced to achieve latency reduction. This unexpected behavior was caused by using a mechanism that allowed the source node to use the same route repeatedly until failure, which caused node exhaustion, leading to a reduction in the network lifetime.

In [13], the study aimed to increase the network lifetime and decrease the transmission delay by using MI. According to transmission time and energy consumption, reinforcement learning (Q-learning) is applied to create multihop paths. Only one node can send data at a time for its cluster head. There are two algorithms used to apply the protocol. The first proposed algorithm goes through the initialization phase to initialize some variables. Then, it enters a loop where it computes an H-table for each node that determines the next possible hops. If the node has accessible hops, it will compute the Q-table values and the reward based on the distance-based path (Dhop) and energy-based path (Ehop). Finally, it updates the Q-table and the Q-value for the next state, while the second proposed algorithm is used to find an optimal path. The suggested algorithm outperformed the other algorithms in terms of energy consumption, throughput, and lifetime performance. In contrast, the transmission delay was not very low. A higher throughput was expected since the data rate was higher with MI communication. The results of the simulations were compared with acoustic-based protocols, which tend to have low propagation speed and high latency. Then, the transmission delay was higher, as expected.

[14] considered an underwater optical network that suffers from a low delivery ratio and high energy consumption. It suggested using a multiagent reinforcement learning protocol (MARL) to consider more information exchange among nodes. It was assumed that a single-agent approach would focus only on its state, causing the residual energy to be unevenly distributed in the network. Furthermore, since the network used optical communication, the data link was more vulnerable than acoustic links. Thus, the link quality was considered while designing this protocol. The protocol works as follows: first, it initializes the routing table and sends a broadcast packet periodically every communication routing step to determine the states of the neighboring nodes and update the routing table. Then, it calculates the reward function based on the link quality of the possible next hop and all nodes' residual energy. After that, it updates the Q-value and V-value and then tries to choose the next hop with the highest V-value. In the simulation, MARL used a small number of relay nodes, which reduced the convergence time. Although the proposed protocol offered less broadcast time and stronger adaptability to water dynamics, it chose a fast route that caused more energy consumption.

[15] focused on extending the network lifetime and balancing the residual energy of nodes while solving the void node problem. In this protocol, each node is considered an agent. Moreover, the node's behavior must be optimized according to the reward function. The suggested reward function attempts to increase the network lifetime and balance the consumption of energy. To solve the void node problem, this study used a mechanism called the adjacent node technique (ADN) to choose a trained and optimal node that is near the source node. Each node that faces the void problem should select another optimal path that maximizes and satisfies its reward function. After using Q-learning, the performance is enhanced in terms of the energy tax. The energy consumption has been reduced even with a large network radius. However, the network lifetime has decreased when using the ADN mechanism.

In [16], the authors proposed Q-learning combined with a deep neural network (DQELR) to consider multiple metrics, such as network lifetime, node mobility, globally optimal paths, energy consumption, and end-to-end latency, in underwater acoustic sensor networks. It adopted two kinds of neural network training on routing decisions: off-policy routing and on-policy routing. The off-policy (offline) training is executed before the network is deployed underwater, where the network topology and node state are known and saved into an experience pool. Then, after the network is deployed underwater, on-policy (online) training starts. The experience pool of the off-policy training is used in the on-policy training to help make better decisions. Each node must store neighbor information and its Q-values with every neighbor, depth, residual energy, and the parameter variation (w) of the neural network. DQELR uses an asynchronous strategy to update w, meaning that when a new loss value is found, w is not updated instantly. The old value of w is stored and accumulated with the loss gradient and updated after a particular time, which archives the neural network noncorrelation input requirement. The multilayer perceptron model used in the suggested protocol consists of an input layer, an output layer, and three hidden layers. The hyperbolic tangent (tanh) function is implemented as the activation function. In the simulation, DQELR achieved a high energy efficiency and network lifetime. In terms of end-to-end latency, it was higher than other protocols. Additionally, its packet delivery ratio was higher than those of other protocols for a low packet generation rate (λ). By increasing the value of λ to 0.05, the packet delivery rate of DQELR decreases compared to those of other protocols. In the proposed protocol, each node can send packets, which causes more frequent packet collisions. Consequently, this improves the network efficiency only for low values of λ.

In [17], a Q-learning-aided ant colony routing protocol (QLACO) was proposed, which differs from the other work because it used ant colony optimization (ACO) with reinforcement learning. The path is selected based on the reward function and the critical ants. The architecture of the network is composed of several surface sinks, many sensor nodes, and several AUVs. The AUV travels and gathers the data from the sensors. Routes are discovered by artificial ants, and then the Q-table is updated. There are two main concepts

introduced in this phase: forward ants (FANTs) and backward ants (BANTs). Each node periodically maintains a Q-table by sending FANTs and broadcast messages. The next hop of packets chosen by FANTs should satisfy the highest Q-value. FANTs collect destination node information before reaching the node location. As a result, the sink has an overview of the network to determine the optimal path. Finally, FANTs change to BANTs when reaching the destination node and return to the source node. The data of all nodes located on the return path are collected to calculate the Q-value by BANTs. Then, the reward function is calculated based on residual energy, time delay, and transmission delay. QLACO was compared with two approaches based on measuring time delay, delivery ratio, and energy consumption in the simulation. The overall performance of QLACO was better than that of the QELAR and the depth-based protocol (DBR). However, the QLACO did not mention how the AUVs fit in the algorithm and how the tasks are distributed among them.

Although the discussed algorithms achieve good performance, there remains room for improvement. Moreover, to overcome the limitations and find the appropriate balance between energy consumption and delay, this study proposes an efficient machine learning-based routing protocol. This study considers the void region issue while enhancing the performance of 3D UWSNs by using MI underwater communications, which is a new promising technology. The suggested protocol uses two machine learning algorithms that are utilized for clustering and routing. An unsupervised machine learning algorithm is applied for clustering, and reinforcement learning is applied for routing. The protocol is partitioned into four phases, which makes it adaptive and flexible. The performance analysis is conducted through simulations and experiments. The simulation results show that BRP-ML can achieve high delivery rates, shorter delays, and a longer network lifetime.

III. BACKGROUND

The UWSN architecture basically consists of sensor nodes (underwater or at the water surface), sink nodes, and AUVs if they exist. There can be one or more sink nodes. If the network has multiple sinks, then sensing nodes have alternative paths along which they can send data packets. The sensor node architecture contains a managing energy unit and power supply, CPU, communication module that applies the used communication method, sensing module, data storage used to store the sensed data, and depth control component, which is a measuring system [18].

A. COMMUNICATION METHOD

MI is based on Faraday's law of induction using two wired coils to interchange data [19]. The modulated sinusoidal current in the transmitting coil (TC) initially generates a magnetic field that changes with time (time-varying) in space, from which the sinusoidal current in the receiving coil (RC) is then induced accordingly. The data are subsequently recovered by demodulating the mediated current. There is no need to equip a power source with MI communications at the receiver. To transmit the data, the magnetic field must be altered according to the waveforms where the data are delivered. The coils' radiation resistance is far less than that of the electric dipole. In other words, only a small amount of energy is radiated across the channel. Therefore, in MI waves, multipath fading is not a concern. In addition, the MI magnetic permeability above water is the same as that underwater and is not affected by water quality, which often changes with the region, time, and depth. Thus, the MI channel behavior is more predictable and steadier than the previously mentioned techniques [20].

Because MI waves travel much faster than acoustic waves, MI waves significantly enhance the underwater communication delay efficiency and provide timely data transmission. A shorter delay improves the design and deployment of underwater communication protocols such as localization, routing, and medium access control (MAC). In addition, the synchronization of the physical layer between wireless devices is reliable and more accessible due to the stable channel response and the slight delay of the MI waves. Moreover, MI waves work by using unseen and unheard waves [21]. Consequently, it facilitates energy-saving and secure communication between wireless devices that can serve military and civil objectives and other applications.

Therefore, MI can be utilized for many underwater applications, such as long-term underwater monitoring, military purposes, disaster detection, and gas or oil leakage detection. Table 1 illustrates a summary comparing optical, acoustic, EM, and MI technologies based on several characteristics.

To accomplish energy-efficient underwater communication, an accurate MI channel model must be built. The energy consumption model contains two main parts: transmitting and receiving power. Since MI communication is used, the MI transmitter is modeled as the primary coil and the MI receiver as the secondary coil of a transformer. The primary coil works at a low frequency and aims to increase the field strength and enhance the magnetic moment. The power used in the primary coil loop is equivalent to the transmitting power, and the power used in the load impedance is equivalent to the receiving power [22]. The transmitting power Pt for a single hop is the real part of the complex number from (1):

$$P_t(r) = Re\{\frac{U_s^2}{Z_t + Z_t'}\} \tag{1}$$

The notations used in the equations of this section are defined in Table 2. The receiving power for a single hop depends mainly on the data processing power plus the receive power, where the energy consumption E_{recive} per receive is expressed as [23]

$$E_{recive} = \frac{L(P_r + P_{trans})}{\alpha_m B(l)} \tag{2}$$

The delay for each packet in a single hop is [23]

$$T_{hop} = \frac{L}{\alpha_m B(l)} + \frac{D_{nm}}{S} \tag{3}$$

TABLE 1. Underwater communication technology comparison.

	Acoustic	Optical	EM	MI
Range of communication	Up to 1 km	10-100 m	Maximum 10 m	10-100 m
Bandwidth	~ kHz	10-150 MHz	~ MHz	~ MHz
Propagation speed	1500 m/s	3.33 x 10^7 m/s	3.33 x 10^7 m/s	3.33 x 10^7 m/s
Channel dependency	Major (Doppler, Multipath, temperature, ambient noise, salinity, pressure)	Major (conductivity, scattering of light, ambient noise, line-of-sight)	Minor (multipath, conductivity)	Minor (conductivity, permeability)
Latency	High	Low	Moderate	Low
Antenna size	~ 10 cm	~ 10 cm	~ 50 cm	~ 15 cm
Operation Stealthily	No	Yes	No	Yes
Marine life impact	Yes	No	No	No
Human activity/waves tolerance	No	Yes	Yes	Yes
Noise immunity	No	No	Yes	No

Since our method follows a multihop path, the distance between the source node n and the destination node (sink node) is the sum of distances in between ($D_{ns} = \sum_{i=1}^{k} D_{nmi}$). Then, the total delay is the sum of multiple single-hop delays $T(D_{ns})$ and the queuing delay [13]:

$$T_{total} = \frac{L}{\alpha_m B(l)} + \frac{D_{ns}}{S} + C * h \tag{4}$$

Table 2 shows the definitions of the notations used in the previous equation. Note that the equations mentioned in this section are used in the simulations for energy consumption calculations.

B. INTELLIGENT ROUTING PROTOCOLS

Routing in UWSNs is a challenging issue. There are special methods of routing applied only in terrestrial wireless networks. These routing methods are based on reinforcement learning, ACO, fuzzy logic, genetic algorithms, or neural networks [24].

1) REINFORCEMENT LEARNING-BASED METHOD

This is the most popular method applied for UWSNs, as discussed in the previous section. Reinforcement learning is implemented easily and adaptable to topology variations. This is the main reason why it is the most common method used for distributed problems.

TABLE 2. Power equations notations.

Notation	Definition
r	Transmission range
n	Sending node
m	Receiving node
s	Sink node
E_{recive}	Energy consumed due to transmission per hop
$P_t(r)$	Transmitting power for a single hop
P_r	Receiving power
α_m	Modulation bandwidth efficiency
$B(l)$	Available bandwidth
P_{trans}	Translating power
k	Number of hops to the sink node
U_s	Transmitter battery voltage
Z_t	Transmitter self-impedance
Z_t'	Transmitter influence on the receiver
D_{nm}	Distance between nodes n and m
L	Packet size
S	MI speed
T_{hop}	Single hop delay
T_{total}	Total path delay
D_{ns}	Distance between the sending node n and the sink node
C	Number of data chunks
h	Duration needed to receive a data chunk

2) ACO-BASED METHOD

The ACO algorithm imitates an ant's behavior, where each ant leaves a trace after it walks that makes the coming ants follow the most visited path [17]. This method is a common routing solution for solving wireless sensor networks. However, balancing convergence and avoiding prematurity must be considered when applying this method.

3) FUZZY LOGIC-BASED METHOD

In traditional logic, an element value can be represented by 1 or 0, where 1 is true and 0 is false. In fuzzy logic, an element can be partially true or false by a certain value between 0 and 1 [25]. This method can be applied to routing optimization and achieve multiple criteria simultaneously. However, it can produce nonoptimal solutions and cannot easily adapt to topology variations.

4) GENETIC ALGORITHM-BASED METHOD

The genetic algorithm is an approach used to solve constrained and nonconstrained optimization problems using natural selection [26]. This method uses a fitness function to obtain a value that represents the solution efficiency. It can work with multiple objective optimization problems, but it is computationally expensive and requires many resources.

5) NEURAL NETWORK-BASED METHOD

A neural network is a set of algorithms that attempts to detect a relationship between data based on a procedure that mimics how the human brain works. Neural networks consist of three

types of layers: input, hidden, and output layers. Flexibility and scalability are the main features of neural networks when applied to UWSNs. In addition, it can work with multicriteria objectives. However, it is known to be a computationally expensive and time-consuming approach [16].

C. REINFORCEMENT LEARNING

Machine learning is a system that learns from data and produces a model that predicts outcomes over time [27]. Machine learning is a subfield of artificial intelligence. Reinforcement learning is a machine learning model training method to perform actions and choices. The difference between reinforcement learning algorithms and classic dynamic programming is that reinforcement learning targets large MDPs and does not assume the exact MDP mathematical model. Figure 1 demonstrates the classic reinforcement learning framework.

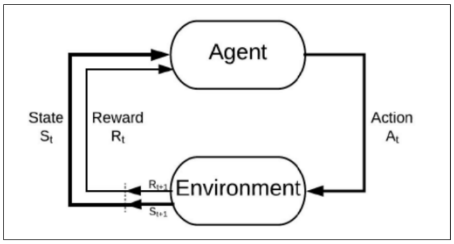

FIGURE 1. Reinforcement learning framework [28].

a) Reward Rt: This is a numerical value that is received by the agent for taking action to move from one state to another in the environment [29].

b) Policy π: This defines the agent's behavior at a given time.

c) Discount factor (rate) γ: This factor is used in the future cumulative reward (return) equation to define the return for infinite series [30].

d) Agent: The agent's goal is to obtain a policy π that maps states to actions optimizing any long-run reinforcement measure. At a given time t, the agent observes the environment, and using some policy, it decides how to take action [31]. Then, it selects an action from a set of available actions. After that, the environment replies by a reward or a penalty and moves to a new state S_{t+1}. The agent tries to take actions that achieve the highest total reward (value or return). As a function of history, the agent can select any action and modify the following policy. It repeats until it reaches the optimal policy.

e) The Bellman equation is useful for reinforcement learning to obtain the value of the current state by knowing the value of the next state. This information assists in finding the optimal Q-function q^* and therefore finding π^*, where Q(s, a) is the expected return starting from state s by selecting action a following π, R_t is the total expected reward for following π in state s selecting action a, and $\max_a Q(s', a')$ is the maximum expected discounted return for any s' selecting action a'.

D. Q-LEARNING

Q-learning is an off-policy learner that learns the optimal policy value without the use of the agent's action. The process of updating the Q-values repeatedly for every state-action pair with the use of the Bellman equation until the equation becomes the optimal Q-function is known as value iteration. The Q-values are stored in the Q-table. To choose the right action, a balance must be struck between exploration and exploitation. The epsilon greedy strategy is used to find that balance. When $\epsilon = 1$, the action taken depends only on the exploration, and as new episodes come, ϵ decreases. ϵ is usually updated when an episode finishes. A random number is generated between 0 and 1 to determine whether the agent chooses exploitation or exploration at each time step [32]. A reward function is derived from the Bellman equation. After several iterations, this function converges to the optimal Q-function (Q^*) [33].

$$\boldsymbol{Q_{t+1}}(s_t, a_t) = \boldsymbol{Q_t}(s_t, a_t) + \kappa(r_{t+1} + \gamma \max_a \boldsymbol{Q_t}(s_{t+1}, a) - \boldsymbol{Q_t}(s_t, a_t)) \quad (5)$$

where κ is the learning rate. It represents a value between 0 and 1 that defines the extent to which the new Q-value overrides the old Q-value. The agent adapts quickly to the new Q-value if the learning rate is high.

The process of the Q-learning algorithm is iterative. The basic steps that it follows begin by creating the Q-table. The learning rate, discount rate, number of episodes, number of steps in each episode, and epsilon must be initialized to a specific value. Then, an action is chosen based on what was explained in the previous sections. After that, the reward is calculated, and the Q-table is updated using the Q-function. The algorithm repeats the steps except for creating the Q-table until the episode finishes.

IV. METHODOLOGY

One of the most critical challenges in UWSNs is how to collect and route the sensed data from the distributed sensors to the sink node. This protocol suggests efficient routing that considers UWSN environmental characteristics, such as power limitations and latency. This can be achieved following a machine learning method that applies reinforcement learning-based (Q-learning) routing that reduces the network latency and energy consumption of UWSNs. The BRP-ML method aims to route the sensed data from the source node to the sink node using a route that consumes less energy and less delay, which is the route that has the maximum Q-value. The design of the proposed routing protocol consists of four phases: initialization phase, discovery phase, clustering phase, and data forwarding phase.

A. INITIALIZATION PHASE

In this phase, the nodes are deployed in their configured locations, and the necessary initializations are set, such as the routing tables, Q-tables, node locations, and initial node energy.

B. DISCOVERY PHASE

In this phase, each node sends broadcast messages to other nodes to update their tables. This information exchange is useful for the clustering phase and the data forwarding phase. Each node that receives a broadcast message calculates the distance between the sending node and itself.

C. CLUSTERING PHASE

In the clustering phase, the optimal number of clusters is chosen, and then the nodes are clustered using a suggested clustering algorithm. Cluster head and edge node selection are performed in this phase. Again, a broadcast message must be used to inform the other nodes about the clustering results. Although this could be considered communication overhead, it is necessary for dynamic topologies. This information exchange could decrease unnecessary data forwarding and balance the energy consumption among nodes.

D. DATA FORWARDING PHASE

After the clustering process, each node knows its cluster ID and its type, whether it is a normal node, cluster head, or edge node. When a normal node has sensed data, it transmits the sensed data to the cluster head, and then the cluster head transmits them to one of the other cluster heads to receive a reward. The reward is calculated using the equations discussed in section IV, and the Q-values in the Q-table are updated. Based on the updates, the node chooses a route using a policy. The void area mechanism (VAM) is applied in this phase to address the void area problem. When the cluster head has collected all data from its cluster's nodes, it can transmit them to another cluster head, edge node, or sink node. It chooses the node that has the highest Q-value. The edge node assists the sink node in reducing the load on the cluster head by making the cluster head transmit the data to a closer node, decreasing the total energy consumption at the cluster head. Furthermore, the edge node can transmit the data to other cluster heads, edge nodes, or the sink node. Figure 2 shows the protocol overview.

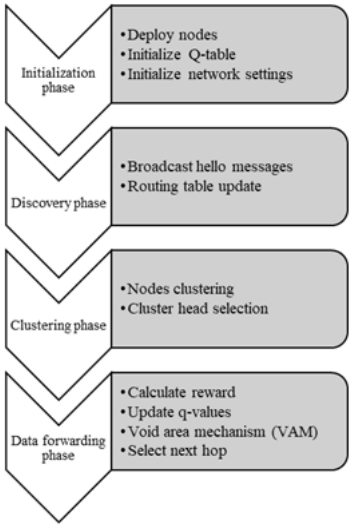

FIGURE 2. Protocol overview.

1) VOID AREA MECHANISM (VAM)

The void area is the area without nodes or the area in which the nodes have drained their batteries. When a node has a packet to send, there are no neighboring nodes that can receive the packets. To solve this problem, a mechanism is suggested named VAM. Assume node A has a packet to transmit, and there are no neighboring nodes between the sink and node A. Some other routes could handle the data through nodes. By using VAM, node A transmits the data to the node with the highest Q-value and starts a timer and waits for an ACK message from the chosen node. Assume that node A chooses node C, and it waits for an ACK message from node C. If node A receives an ACK message from node C, no action is needed since it means that the path to the sink node can go through node C and there is no intermediate void area. Moreover, if the timer has expired and no ACK message has returned to node A, then the Q-value of node C will be reduced, and an alternative path will be chosen to deliver the packet. Because the nodes with low Q-values have a low probability of being selected, the routes that bypass the void areas are excluded in this mechanism. Figure 3 is an illustration of the discussed case.

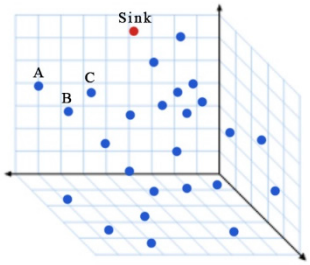

FIGURE 3. Illustration of the network architecture.

2) STRUCTURE OF DATA PACKETS

Three types of data packets are used in the BRP-ML algorithm. Each type is used for a different purpose. These are intended to facilitate the information exchange among nodes and properly achieve both the clustering and routing processes. The three types of packets are broadcast packets, data packets, and ACK packets. The broadcast packet contains the packet ID, node type (cluster head, edge node, normal node), timestamp, source node ID, cluster-ID, residual energy, and max Q-value. The broadcast packet is used to exchange information among nodes and maintain the updated information. The broadcast is among all live nodes, and it is performed periodically. Each time the clustering process is executed, broadcasts must be exchanged. The data packet includes the ID, type, timestamp, previous node ID, previous node max Q-value, previous node residual energy, destination node ID, and data. The previous node information

TABLE 3. Structures of packets.

Broadcast packet	Data packet	ACK packet
ID	ID	ID
Type	Type	Type
Time stamp	Time stamp	Time stamp
Source Node ID	Previous node ID	Source node ID
Cluster ID	Destination node ID	Destination node ID
Residual energy	Data	Message ID
Max Q-value		

in data packets is the information of the previous hop chosen. The unicast data packet is used to transfer the sensed data to the sink node using a multihop path. Having different types of packets benefits the routing protocol. To accomplish the clustering process, only broadcast packets are used. Then, for the routing process, both types of packets are used. Finally, the ACK packets are used to ensure that the data packets reach the destination. The ACK packets can be sent from any node in the network. Each time a node transmits a data packet, it waits for an ACK packet from the destination node to guarantee that the data packet is received. The average round trip time is calculated and used as an ACK expiry time. Table 3 summarizes the details of the packet structures used.

3) CLUSTERING PROCESS

The clustering process is the method used to divide any data points or population into groups such that the points in the same group have similar traits. In UWSNs, the communication interference is known to be high, and by using clustering, the interference can be reduced. Furthermore, the energy consumption is more balanced, which extends the network lifetime. The network is divided into groups or clusters, where each cluster has a cluster head. The sensors communicate only with their cluster head. Then, the cluster head transfers the aggregated data to the sink node either by a multihop path or a single hop. This process enhances the performance of UWSNs [34].

K-means is a partitioning-based algorithm. It splits the nodes based on centroids where the similarity among clusters depends on the node closeness to the cluster centroid [35]. It is a common algorithm because it is easy to implement and has low computational complexity and low memory consumption [36].

K-means++ is an enhanced version of the K-means algorithm that addresses the poor initialization problem [8]. It is a simple algorithm that can provide more accurate results than the standard K-means.

BRP-ML uses an adaptive clustering technique to adapt to network changes. In the network, the sensor nodes are clustered, where each cluster has a cluster head and an edge node. The cluster head is responsible for aggregating the data packets from the other nodes in the cluster and transmitting them to the sink using a multihop path. The edge node assists the cluster head in aggregating and transmitting the data. By using clustering, most nodes transmit the data using short hops, which consumes less energy and extends the network lifetime. Clustering in UWSNs is usually performed in three steps: define the number of clusters (k), perform the clustering process, and assign the cluster head and edge node. The clustering process is the procedure of grouping the nodes into k clusters. This process can be repeated when the number of alive nodes decreases. As mentioned, K-means++ is applied to form clusters that are modified to adapt to the nature of the underwater environment. In the underwater three-dimensional coordinate system, each node has a location vector (x, y, z). Assume that node A has coordinates (x_1, y_1, z_1) and node B has coordinates (x_2, y_2, z_2). Then, the distance between two nodes A and B in a 3D system is

$$D_{AB} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (6)$$

a: PREREQUISITES AND ASSUMPTIONS

1. The shapes of the clusters are not important to our algorithm, which means that it does not depend on a particular cluster feature.
2. Since MI is used as a communication method in this work, the received magnetic field strength (RMFS)-based localization technique is applied. It utilizes a tri-directional antenna to increase the estimation accuracy [37]. Therefore, each node can obtain its three-dimensional coordinates.
3. The location of each node is configured before the deployment, and the nodes can adjust their locations following the configuration.
4. Sensor nodes are deployed randomly in a 3D coordinate system.
5. All sensor nodes have the same initial limited energy, except the sink node, which has an unlimited power supply.
6. Three types of nodes are used: normal nodes that sense and transmit data to a cluster head using a single hop; a cluster head, which is responsible for aggregating the data from normal nodes and sending data to the sink node using a multihop path; and the edge node, which is used as a cluster head assistant.
7. Each cluster has one cluster head, one edge node, and multiple normal nodes.

Figure 4 shows an illustration of the network architecture used, where the deployed nodes are clustered, and each cluster has a cluster head, edge node, and normal node.

b: CHOOSING THE OPTIMAL NUMBER OF CLUSTERS (K)

Neither the K-means nor K-means++ algorithm specifies the number of clusters (k), which should be assigned prior to the clustering process. It is important to choose the optimal number of clusters since it affects the network lifetime. Having many clusters could increase the communication overhead, and having few clusters causes the formation of large clusters, which increases the power consumption. Therefore, BRP-ML uses a combination of two methods to

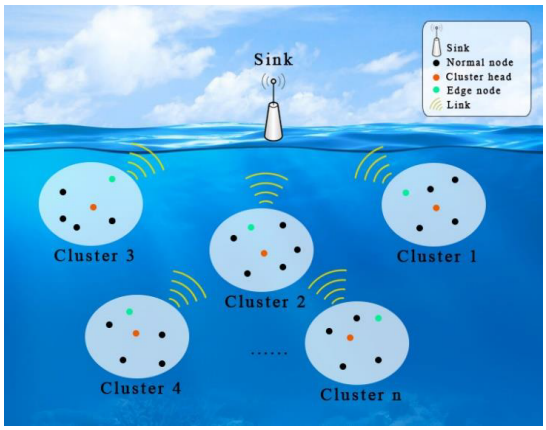

FIGURE 4. Illustration of the network architecture.

obtain the accurate optimal value of k. The first method is the silhouette method that calculates a silhouette coefficient, which shows the goodness of the clustering algorithm, performance wise.

$$\text{Silhouette coefficient} = \frac{d2 - d1}{\max(d1, d2)} \tag{7}$$

where d2 is the mean distance of the nearest cluster, and d1 is the mean intracluster distance for each data point. After selecting random values of k, the silhouette coefficient is calculated for each of them. Thus, k with a higher coefficient is the optimal value [38]. Although this method gives an obvious numeric outcome, its time complexity is very high if calculated for each possible k, and with high-dimensional problems, it takes more time to converge. The other method is the elbow method. The elbow method calculates the within-cluster sum of squared errors (WSS) for each k. WSS is the mean of the squared distance between each data point and the closest cluster center. WSS can be visualized as a curve, and the optimal value of k is the elbow of the curve [39]. This method is faster than the silhouette method, but sometimes the elbow method can be ambiguous and uncertain.

The number of clusters can vary from 2 to (n-1)/2, where n is the number of data points. The elbow method is used first as a decision rule. If the value of k is obvious, the rest of the clustering algorithm will continue to execute. If the value of k is vague, the silhouette method will be applied as a validation method only for the doubtful values of k. Note that the most doubtful value is the largest value at which distortion declines. This step guarantees that k is the optimal value while reducing the time complexity and avoiding having a few clusters to reduce energy consumption. Algorithm 1 shows how the optimal value of k is obtained.

Algorithm 2 shows the steps of the clustering process, which applies K-means++ with modifications to be applied in 3D environments. The algorithm requires the value of k, which is the output of Algorithm 1. The first centroid is selected randomly. The subsequent centroids are selected with probability $P = \frac{D(x')^2}{\sum_{x \in X} D(x)^2}$. D(x), which is the distance between node x and the previously chosen centroid c, and

Algorithm 1 Selecting Optimal k Value

Input: All nodes X = $\{x_1 \ldots, x_n\}$ where n = number of nodes, All nodes X locations
Output: Optimal k value
% Decision rule%
1. For i = 2 to i = (n-1)/2:
   2. Run the K-means algorithm.
   3. Calculate the WSS for each data point to its centroid.
   4. Save the values of distortions.
   5. Array A = doubtful k values.
% Validation %
   6. For each doubtful value in A:
      7. Calculate the silhouette coefficient
8. k = highest silhouette score
9. Return k

D(x′) is the distance between previously chosen centroid c and the new centroid c_{i+1}. After that, the previous steps are repeated until k centroids are reached. Finally, it follows the standard K-means steps that assign each data point to the nearest centroid.

Algorithm 2 Clustering Process

Input: All nodes X = $\{x_1 \ldots, x_n\}$ where n = number of nodes, Value of k.
Output: K clusters
1. A random centroid c_1 is selected where $c_1 \in$ X.
   2. For j = 1 to j = k-1
      3. For i = 1 to i = n
         4. Distance between x_i and previously chosen centroid c_i is calculated using (6)
         5. A new centroid c_{i+1} = x‘∈ X = $\{x_1 \ldots, x_n\}$ is selected with probability P
6. Repeat
   7. For i = 1 to i = n
      8. For j = 1 to j = k
         9. Calculate the distance between x_i and c_j
      10. Assign x_i to the nearest c.
      11. Update the cluster center to the average location.
12. Until convergence

c: CLUSTER HEAD AND EDGE NODE ASSIGNMENT

In BRP-ML, there must be a node that is responsible for aggregating the data from the cluster nodes and transmitting them to the base station by a multihop route. Two factors are considered when assigning cluster heads. First, the cluster head must be the node that has the maximum residual energy because it performs an energy-consuming task. Second, it must be the closest cluster member to all nodes within the cluster. The edge node assignment is selected after the

TABLE 4. Constant notations.

Symbol	Description
ω	Constant cost
β_1, β_2	Weight of costs
α	Delay sensitivity

clustering process. Normally, in each cluster, there is one cluster head and multiple normal nodes. The cluster head collects the data from its cluster members and forwards the data to the sink by multiple hops through other cluster heads. However, in our approach, there is an edge node used to reduce the load on the cluster heads. Therefore, each cluster includes the following node types:

Normal nodes: Sense data and transmit them to the cluster head.

Cluster head: Aggregates the data from normal nodes and transmits them to the sink node through other cluster heads or edge nodes.

Edge node: Receives the data from cluster heads and transmits them to the sink node using a multihop path through edge nodes or other cluster heads.

In the routing phase, the Q-learning algorithm can forward the data packets through either cluster heads or edge nodes. There are two factors for selecting an edge node. First, the edge node must be the node with the maximum residual energy after the cluster head within the cluster. Second, it must be the nearest node among the cluster nodes to the sink. After the clusters are formed, cluster heads and edge nodes are assigned. The nodes send broadcast messages indicating the type of node, status, and cluster number. Usually, the cluster head consumes more energy than normal nodes, which shortens the cluster head lifetime. In some cluster head assignment approaches, a node is assigned to be a cluster head and does not assign another node until the cluster head energy drains. In contrast, our proposal is based on the reselection process approach, where cluster heads change periodically. This means that when a cluster head residual energy becomes less than a certain threshold, another node is selected as the cluster head automatically using the same factors described previously. The new cluster head broadcasts a packet notifying other nodes regarding the changes. Applying this process prolongs the network lifetime. The following rule shows the threshold value E_{Th}. where E_{ave} is the average residual energy of within-cluster nodes: $E_{Th} = E_{ave}$. Algorithm 3 shows the steps of selecting a cluster head:

The edge node selection process is the same as the cluster head selection process except that the edge node selection rule is different, noting that the complexity of both algorithms is O(n).

4) POLICY AND REWARD FUNCTION

In this section, the reward and Q-function are described. Tables 4 and 5 contain the notations used in the following equations.

Algorithm 3 Cluster Head Selection Process

Input: Residual energies of X_n where $X_n = \{x_1 \ldots, x_m\}$ and m = number of nodes within cluster, location of each $x \in X_n$, Maximum Communication Distance (MCD).
Output: Cluster head CH

1. Calculate E_{ave}.
2. For each $x \in X_n$ and $>E_{ave}$:
3. Calculate the distance between x_i and other x's.
4. Calculate CH selection criteria following the rule.
5. CH = the value with the highest selection criteria.
6. If there are more than one node with the same results:
7. Choose CH to be Max (residual energy).
8. Return CH

Algorithm 4 Edge Node Selection Process

Input: Residual energies of X_n where $X_n = \{x_1 \ldots, x_m\}$ and m = number of nodes within cluster, location of each $x \in X_n$, Maximum Communication Distance (MCD), sink node location, CH.
Output: Edge node EN

1. Calculate E_{ave}.
2. For each $x \in X_n$ and!= CH and $> E_{ave}$:
3. Calculate the distance between x_i and the sink node.
4. Calculate EN selection criteria following the rule.
5. EN = Max(A).
6. If there are more than one node with the same results:
7. Choose EN to be Max (residual energy).
8. Return EN

TABLE 5. Variable notations.

Symbol	Description
i	Sending node
j	Receiving node
$c(E_i)$,$c(E_j)$	Energy-related cost
$c(D_{ij})$	Delay related cost
E^i_{res} , E^j_{res}	Node residual energy
E^i_{ini} , E^j_{ini}	Node initial energy
t_{delay}	Communication delay between node i and j
$cf.$	Communication failure
T^{cf}_{ij}	Time consumed due to communication failure
P_{fail}	Probability of failed transmission
n_{Lp}	Number of lost packets
n_{tp}	Number of transmitted packets
s_i	State of node i
a_i	Action of node i
$Q(s_i,a_i)$	Action-utility function of node i
$Q^*(S)$	Action with the highest Q-value

In BRP-ML, we assume that node i attempts to forward a packet to node j, where the state of that agent is s_i using action a_i. The transmission reward is $R_{success}$ in the case of

successful transmission is

$$R_{success} = -\omega - \beta_1 \left[c\left(E_i\right) + c\left(E_j\right) + \alpha * c\left(D_{ij}\right)\right] \quad (8)$$

where $0 < \omega < 1$, $0 < \beta_1 < 1$, and $0 < \alpha < 1$.

Our reward function focuses on two main parts: the energy cost and delay cost. The constant cost ω is added due to node communication, which occupies channel bandwidth. Additionally, a delay sensitivity factor is added to balance energy and delay when determining the transmission route. If the delay sensitivity factor is set to one, the selected path considers only the delay. Therefore, the sensitivity factor is the weight given to the delay cost in the equation.

The energy cost function uses the initial node energy and the residual node energy. For each node, the higher the energy, the lower the cost it will be assigned. Therefore, a node with higher residual energy is more likely to be selected. Hence, the energy cost function is

$$c(E_i) = 1 - \frac{E^i_{res}}{E^i_{ini}}, \quad c(E_j) = 1 - \frac{E^j_{res}}{E^j_{ini}} \quad (9)$$

The transmission delay between nodes i and j is used to calculate the delay cost. If that delay between nodes i and j is high (t_{delay}), the delay cost is high. (10) shows the delay cost function:

$$c\left(D_{ij}\right) = 1 - \frac{1}{t_{delay} + 1} \quad (10)$$

In the case of transmission failure, the delay and the energy costs are doubled. Considering that the delay cost is the time consumed due to communication failure T^{cf}_{ij} plus the transmission delay between nodes i and j, the reward function is defined as

$$R_{fail} = -\omega - \beta_2[2 * c\left(E_i\right) + c\left(E_j\right) + \alpha * c\left(T_{ij} + T^{cf}_{ij}\right)] \quad (11)$$

Since the Q-learning functions are based on MDP, the probability of transition from one state to another must be considered to calculate the direct reward function. Therefore, the successful transition probability and the failed transition probability are used to compute the direct reward function. The direct reward is defined as

$$\text{Reward} = (1 - P_{fail})R_{success} + P_{fail}R_{fail} \quad (12)$$

where the probability of failed transmission is the number of lost packets divided by the number of transmissions. Thus, the failed transmission probability is shown in (13). Furthermore, when the sum of the failed transmission probability and successful transmission probability is equal to one, the probability of successful transmission is 1-P_{fail}:

$$P_{fail} = \frac{n_{Lp}}{n_{tp}} \quad (13)$$

Thus, combining (12) and (13), the direct reward can be rewritten as

$$\text{Reward} = \left(1 - \frac{n_{Lp}}{n_{tp}}\right) * R_{success} + \left(\frac{n_{Lp}}{n_{tp}}\right) * R_{fail} \quad (14)$$

After combining (8), (11), and (14), the action utility function is expressed as

$$\begin{aligned} Q\left(s_i, a_i\right) &= \text{Reward} + \gamma \left[\left(1 - P_{fail}\right) * Q*\left(s_i\right) \right. \\ &\quad \left. + P_{fail} * Q*\left(s_j\right)\right] \\ &= \text{Reward} + \gamma \left[\left(1 - \frac{n_{Lp}}{n_{tp}}\right) * Q*\left(s_i\right) \right. \\ &\quad \left. + \left(\frac{n_{Lp}}{n_{tp}}\right) * Q*\left(s_j\right)\right] \end{aligned} \quad (15)$$

5) ROUTING PROCESS

By this phase, the nodes are clustered, and each cluster has a cluster head and an edge node. Data packets are routed to the sink node only through cluster heads or edge nodes. Choosing the next hop depends on the result of the reward function and Q-table. Whenever a normal node has a packet to transmit, it transmits that packet directly to the cluster head. After that, the cluster head aggregates the sensed data from its cluster nodes. The data forwarding algorithm is where the Q-learning process happens. First, it initializes the Q-table where rewards are to be stored. If the node type is normal, then it will transmit the packet. Otherwise, if the node type is a cluster head or edge node, it will make a list of possible hops, where the possible hops are toward other cluster heads or edge nodes within the communication distance. After that, it calculates the reward using the direct reward function (14) and the Q-value using the Q-function (15). Then, a timer starts, which is used for the VAM. Using the results of the Q-function, the next hop is chosen such that its value has the highest Q-value. If the timer terminates and no ACK packet is received by the sending node, the reward of the chosen hop will be lowered, and another hop will be selected. Finally, the Q-table is updated. Algorithm 5 shows the steps of the fourth phase, which is the data forwarding.

Algorithm 5 Data Forwarding Protocol

Input: Node location, Node residual energy, Node cluster ID, Node Type, Sensed data, Maximum Communication Distance (MCD).
Output: Optimal route.

1. Initialize Q-table
2. If (node type is Normal node) then:
3. Transmit the sensed data to the cluster head.
4. Else:
5. For each nonempty node N_i do:
6. For each node not in the same cluster & its type = cluster head or edge node do:
7. Calculate distance between N_i and N_j.
8. Let N_{ij} = neighboring nodes where distance <= MCD.
9. While the next hop is not Sink node do:
10. Calculate the direct reward.
11. Start a timer. // *used for the void mechanism*
12. Calculate the Q-value.
13. Select the next hop N_x with the highest Q-value.
14. If (timer ends and ACK not received):
15. Lower the reward of the chosen hop.
16. Go to step 10.
17. Update the Q-table.
18. $N_i = N_x$*//Update current Node*
19. End

V. SIMULATION AND RESULTS

In this section, simulations are conducted to evaluate the performance of BRP-ML using Sublime Text 3.2.2 on a local PC with an Intel i7 8th generation 3.20 GHz processor, 16 GB of RAM, and the Windows 10 platform. The 3D network figures in this section are plotted using MATLAB R2020b.

A. SIMULATION METRICS

The measured metrics are as follows:

1. Energy efficiency $= \frac{\text{Output energy}}{\text{Input energy}}$
2. Average delay: average time required for data to reach the sink node.
3. Delivery rate $= \frac{\text{Number of delivered packets}}{\text{Total number of transmitted packets}}$
4. Network lifetime: total routing time until the first node expires.
5. Alive node percent: percent of alive nodes among all nodes.
6. Execution time: time it takes to run the algorithm.

The results of the simulations are compared with the QL-EDR [13] and QELAR [11] routing protocols.

B. SIMULATION CASES

Four simulation cases are executed to test the path changes, energy consumption, and delay on different network sizes in terms of the number of nodes. The nodes are distributed uniformly in a 250 m $\times$ 250 m $\times$ 80 m area. The selected values of α, $\beta 1$, and $\beta 2$ are based on the tests performed in subsection D that achieve the best result. Each simulation case is tested three times. In the first and second runs, the starting locations of nodes are the same. The second run is considered to compare the chosen path, delay, and consumed energy with the first run. The third run is when half the nodes have depleted their energy. The tracked packets are generated from the same node in all runs. In the first simulation case (test-One), a network is deployed in a 3D environment with 120 nodes and one surface sink node where the nodes are labeled with IDs ranging from 0 to 120. The optimal number of clusters for this case using the proposed algorithms is four; therefore, the number of clusters formed in this test is four clusters.

In test-One, node number 70 is selected randomly to tack the generated packets. In the first and second runs, the same path is chosen by the BRP-ML algorithm, and four hops is the length of the path to the sink node. In the third run, the path has changed, but the number of hops has decreased to three. However, the delay decreases by 5%, and the consumed energy increases by 20%. The reason is that the number of hops has decreased, which causes the delay to decrease and the energy to increase.

In the second simulation case (test-Two), a network is deployed in a 3D environment with 150 nodes and one surface sink node where the nodes are labeled with IDs ranging from 0 to 150. The number of clusters formed in this test is three. Node number 70 is also selected by the BRP-ML algorithm randomly as a starting node. Similar to test-One, the same path is chosen in the first and second runs, and four hops is the length of the path to the sink node. In the third run, the number of hops is also four, but the chosen nodes as a path have changed. Both delay and energy have increased. The delay and energy have increased by 8% and 18%, respectively. This is because the number of hops is the same, but the route has a longer distance, which causes the delay and the energy to increase.

In test-Three, the network has 170 nodes with the same starting node number 70. The results of the first and second runs are similar to those of test-Two because the network does not change significantly. In the third run, the path is different from test-Two because the chosen hops have depleted their energy, which causes the algorithm to change the path. Because the hops are less than those in the first and second runs, the delay has decreased by 6%, and the energy consumption has increased by 17%. The energy has increased due to having hops with longer distances.

Finally, in the fourth test, 200 nodes are deployed in the network. In the first and second runs, the delay is less than in all previous tests because the nodes are denser and there are more likely hops to choose from. Additionally, the number of hops is only three. The delay and energy consumption exhibit an inverse relationship, which means the energy consumption increases due to delay reduction. That is why in the third run, the algorithm attempts to balance the delay and energy by choosing a path with more hops to keep that balance. The delay has increased by 9%, and the energy consumption has decreased by 10%.

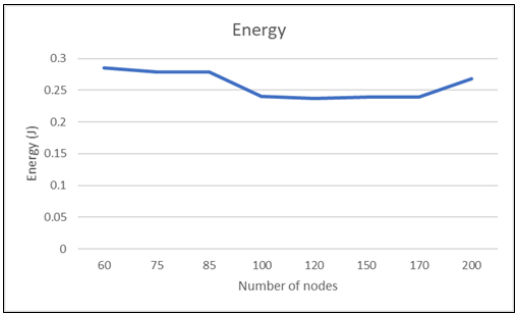

FIGURE 5. Energy consumption comparison for all cases.

Figure 5 demonstrates the effect of network size on energy. The energy consumption is higher for networks with a smaller number of nodes. The energy consumption decreases for networks between 100-170 nodes and then increases again with networks of 200 nodes, noting that all networks have the same area size. Although the energy consumption increases in a 200-node network, the delay is reduced, which achieves the targeted balance. Figure 6 shows the effect of network size on the delay. The delay wavers for networks with a small number of nodes, and it obtains almost the same delay for networks with 100-200 nodes. The delay is higher for small networks because the network area size is the same for all of them, and there are gaps between the nodes that cause long delays.

In conclusion, the outcome performance does not depend on one feature. Both matrices work along with each other, and the algorithm balances them out.

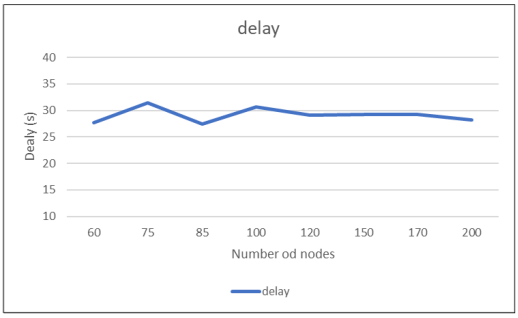

FIGURE 6. Delay comparison for all cases.

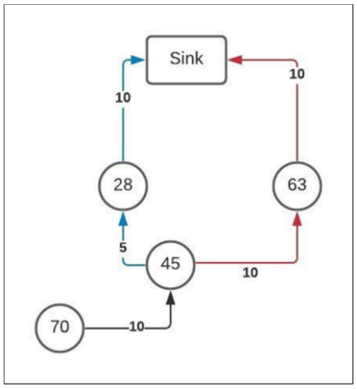

FIGURE 7. Number of hops example.

In conclusion, when two cases have the same number of hops, the delay and energy could have different values because the route distance is not the same. Figure 7 simplifies this conclusion. Assume that node number 70 is the source node and node number 45 is the cluster head, and all nodes' batteries are full. There are two paths available to reach the sink, using either node 63 or node 28. The total distance of the red path is 30 m, and the total distance of the blue path is 25 m. The algorithm chooses the blue path since it costs less than the red path. After a while in the simulation, the battery levels change. According to the algorithm, the red path may be chosen if the energy of node 63 is higher than that at node 28 to avoid draining the node's battery. In both cases, the number of hops is three, but the blue path delay is less than the red path. The BRP-ML algorithm balances the delay and energy consumption.

C. COMPARISONS OF BRP-ML PROTOCOL, QELAR, AND QL-EDR

Table 6 summarizes the simulation parameters used in the comparisons of the BRP-ML protocol, QELAR, and QL-EDR. The underwater network model is built based on known network models, including the principle of node data transmission and the connectivity characteristics of the network. The same model was followed in [11] and [13].

TABLE 6. Simulation parameters.

Parameter	Value
Number of nodes	170
Network area	250 x 250 x 80 m^3
Maximum communication distance	65 m
Initial node energy	2 J
Node position offset	5 m
Number of sink nodes	One sink node located on top of the water surface and in the center.
Packet size	1000 bits

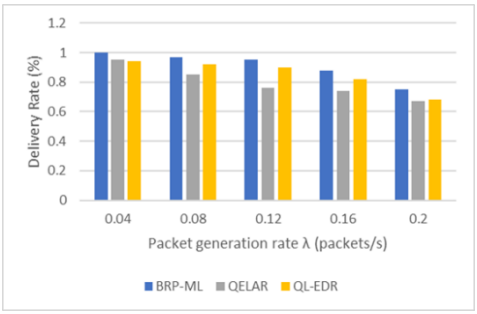

FIGURE 8. Delivery rate comparison.

1) DELIVERY RATE

Figure 8 shows a comparison between BRP-ML, QELAR, and QL-EDR in terms of delivery rate with different packet generation rates (λ). The packet generation rate is on a per node basis where it follows the Poisson process, and the duration between each packet generation follows an exponential distribution. The delivery rate decreases as the packet generation rate increases because there are more packets to transmit, making the node's energy deplete faster. Therefore, the number of packets that reach the sink is reduced. BRP-ML improves the delivery rate by approximately 25% compared with QELAR and 6% compared with QL-EDR. BRP-ML uses the void area mechanism, which increases the delivery rate.

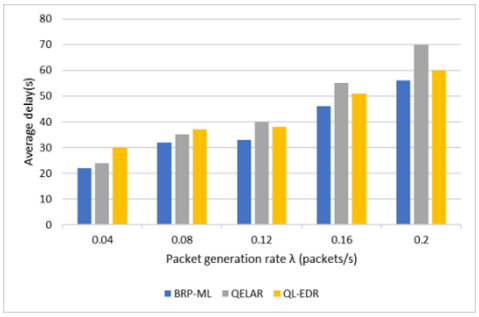

FIGURE 9. Average delay comparison.

2) AVERAGE DELAY

Figure 9 demonstrates a comparison between BRP-ML, QL-EDR, and QELAR in terms of average delay with

different packet generation rates. The average delay increases as the packet generation rate increases. This is because when the number of sent packets increases, the transmission failure increases, and there are more packets to retransmit, which increases the average delay. QELAR has a higher average delay than BRP-ML by 18%, whereas QL-EDR has a 13% higher average delay than BRP-ML. This is because the reward function of BRP-ML considers both the delay and the energy. The difference is not significant because the retransmitted packets take a longer path, which takes more time and increases the delay.

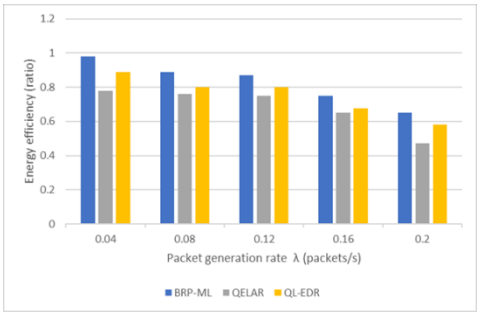

FIGURE 10. Energy efficiency comparison.

3) ENERGY EFFICIENCY

Figure 10 shows a comparison between BRP-ML, QL-EDR, and QELAR in terms of energy efficiency with different packet generation rates. The energy efficiency decreases as the packet generation rate increases. The reason is that the more packets there are to transmit, the more energy is consumed by the nodes, which reduces the network lifetime. BRP-ML outperforms QELAR and QL-EDR by 16% and 9%, respectively. The design of our reward function and approach balances the energy consumption better than QELAR.

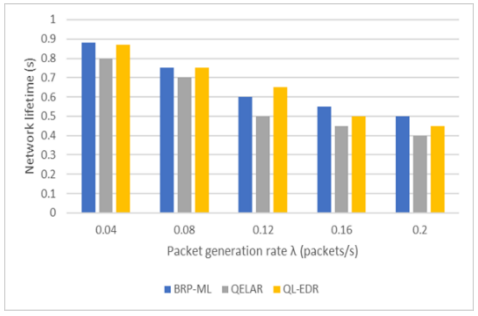

FIGURE 11. Network lifetime comparison.

4) NETWORK LIFETIME

Figure 11 demonstrates the relationship between the normalized network lifetime and packet generation rate. The network lifetime decreases as the packet generation rate increases. The reason is that an increase in packet transmission results in the consumption of more energy, which reduces the network lifetime. The QELAR algorithm achieves the lowest network lifetime, whereas BRP-ML achieves the highest network lifetime among the compared algorithms.

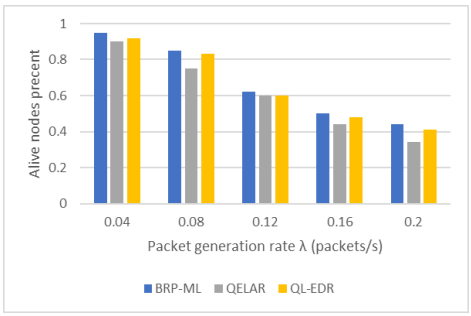

FIGURE 12. Network lifetime comparison.

5) ALIVE NODE PERCENT

Figure 12 shows the alive node percent. The alive node percent decreases as the packet generation rate increases. This is because more packets are transmitted, which drains the nodes' batteries. BRP-ML achieves a better result than the other two algorithms. It has a 14% enhancement compared with QELAR and a 5% enhancement compared with QL-EDR.

6) EXECUTION TIME

To evaluate the energy consumption, the time required for the execution of the algorithm must also be considered. The cost of computing energy E_{run} can be investigated using (16) [14].

$$E_{run} = Pt_{run} \tag{16}$$

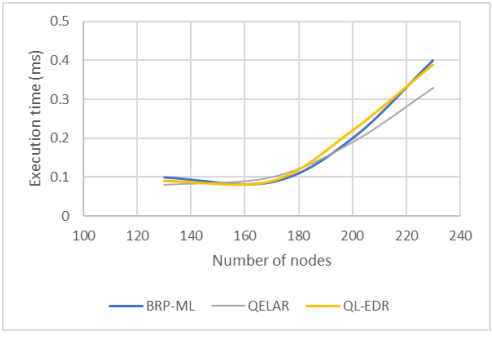

FIGURE 13. Average-case execution time comparison.

where t_{run} is the execution time, and P is the hardware power. The computing energy must be lower than the transmission energy to achieve a longer network life. One of the important factors that affects the learning process is the number of nodes. Because exploitation and exploration consume long computational times, various node-density networks are tested using the same hardware to assess the efficiency of BRP-ML. Figure 13 shows the relation between the number of nodes and the algorithm's average-case execution time.

The algorithms are run 25 times to obtain the average-case execution time. The number of nodes ranges between 130 and 230. When the number of nodes = 130, the fastest approach is QELAR, followed by BRP-ML, which has a similar QL-EDR execution time. When the number of nodes reaches 230, the QL-EDR execution time increases by 23% compared to the case when the number of nodes = 170. In BRP-ML, the execution time increases by 22%, while QELAR increases by 30%. The more time it takes to execute the learning algorithm, the more energy it consumes. Therefore, execution time is a critical factor in designing routing protocols for scalable networks.

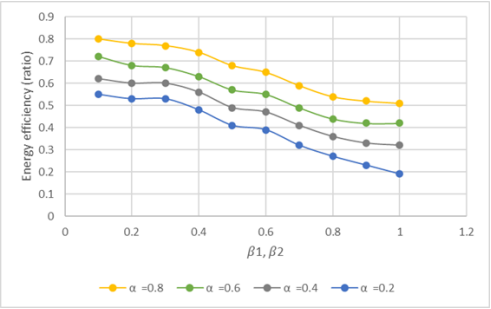

FIGURE 14. Energy efficiency with different parameter values.

D. COMPARISONS OF DIFFERENT PARAMETER VALUES

Different coefficients are tested to investigate the influence on the energy efficiency and delivery rate. Figure 14 shows the energy efficiency with different β coefficients. These coefficients are equivalent to the coefficients of the reward functions $\beta 1 = \beta 2$ with values varying between 0.1 and 1, and the values of α vary between 0.2 and 0.8 with a step of 0.2.

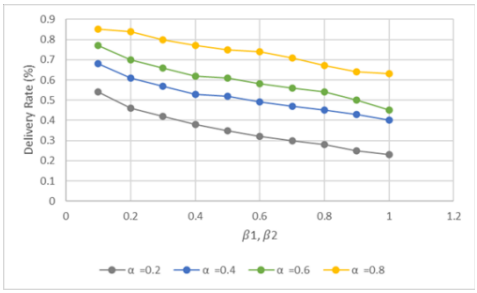

FIGURE 15. Delivery rate with different parameter values.

When $\beta 1$ and $\beta 2$ are set to high values, the route selection decision differentiates based on the energy consumption, while this decision considers the delay when α is assigned high values. The energy efficiency increases as α increases. This is because the node's remaining energy affects the routing with a higher value of ω in the reward function. Thus, nodes that have more energy are more likely to be selected to forward the packet.

Figure 15 illustrates the delivery rate of different coefficients. The delivery rate decreases as α decreases. This is because relying on minimizing delay while choosing the path by the algorithm causes uneven distribution of the energy. When $\beta 1$ and $\beta 2$ are set to high values and $\alpha = 0.8$, the delivery rate drops by 25% compared to low values of $\beta 1$ and $\beta 2$. Therefore, to balance the energy efficiency and delivery rate, choosing α should consider both delay and node residual energy. To achieve a high delivery rate and energy efficiency, the referral values to choose are $\alpha = 0.7$ and $\beta 1 = \beta 2 = 0.6$ to consider both energy consumption and delay.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have presented a machine learning algorithm to address some of the UWSN limitations. We focused on extending the network lifetime by decreasing the delivery delay while balancing energy consumption. A BRP-ML routing protocol was proposed, which is a reinforcement learning (Q-learning) algorithm used to route the sensed data to the surface sink node for further data analysis. The protocol functions in four phases to ensure that each phase is working effectively and is more flexible for any future modifications. BRP-ML uses a clustered network that helps adapt to network changes and reduces communication interference. For the clustering phase, K-means++ is used to divide and group the nodes. To validate the clustering process, the silhouette score was used as an internal validation method to measure the algorithm performance. BRP-ML considers the void area issue, which is a common problem in UWSNs. We presented a VAM to address void regions and increase the delivery rates. We demonstrated that BRP-ML could effectively enhance network performance. Simulation results showed that it could balance the energy consumption and delivery delay while considering the void area. The results showed that BRP-ML increased the delivery rate up to 25% and decreased the average delay up to 18% while achieving energy efficiency up to 16% compared to the QELAR and QL-EDR algorithms.

For future work, we will consider deploying a multi-sink node to study the effect on the packet delivery rates and delay. Recently, some researchers have used AUVs for underwater wireless charging. We will consider deploying such AUVs to charge the nodes before the nodes deplete their energies to prolong the network lifetime and enhance the performance. Additionally, a machine learning algorithm was designed for an AUV to choose the route it takes to charge nodes. The normal head will make a forwarding decision in which it decides whether to forward it to the cluster head or edge node to save time and energy.

REFERENCES

[1] University of Rhode Island's. *The First Practical Uses of Underwater Acoustics: The Early 1900s*. [Online]. Available: https://dosits.org/people-and-sound/history-of-underwater-acoustics/the-first-practical-uses-of-underwater-acoustics-the-early-1900s/

[2] University of Rhode Island's. *Between World War I and World War II: The 1920s and 1930s*. [Online]. Available: https://dosits.org/people-and-sound/history-of-underwater-acoustics/between-world-war-i-and-world-war-ii-the-1920s-and-1930s/

[3] G. Tuna, O. Arkoc, and K. Gulez, "Continuous monitoring of water quality using portable and low-cost approaches," *Int. J. Distrib. Sensor Netw.*, vol. 9, no. 6, Jun. 2013, Art. no. 249598.

[4] A. Faustine, A. N. Mvuma, H. J. Mongi, M. C. Gabriel, A. J. Tenge, and S. B. Kucel, "Wireless sensor networks for water quality monitoring and control within lake Victoria basin: Prototype development," *Wireless Sensor Netw.*, vol. 6, no. 12, pp. 281–290, 2014.
[5] J. Heidemann, M. Stojanovic, and M. Zorzi, "Underwater sensor networks: Applications, advances and challenges," *Philos. Trans. Roy. SoC. London A, Math. Phys. Sci.*, vol. 370, no. 1958, pp. 158–175, Jan. 2012.
[6] M. Waldmeyer, H.-P. Tan, and W. K. G. Seah, "Multi-stage AUV-aided localization for underwater wireless sensor networks," in *Proc. IEEE Workshops Int. Conf. Adv. Inf. Netw. Appl.*, Singapore, Mar. 2011, pp. 908–913.
[7] K. M. Awan, P. A. Shah, K. Iqbal, S. Gillani, W. Ahmad, and Y. Nam, "Underwater wireless sensor networks: A review of recent issues and challenges," *Wireless Commun. Mobile Comput.*, vol. 2019, Jan. 2019, Art. no. 6470359.
[8] I. F. Akyildiz, P. Wang, and Z. Sun, "Realizing underwater communication through magnetic induction," *IEEE Commun. Mag.*, vol. 53, no. 11, pp. 42–48, Nov. 2015.
[9] N. Li, J.-F. Martínez, J. M. M. Chaus, and M. Eckert, "A survey on underwater acoustic sensor network routing protocols," *Sensors*, vol. 16, no. 3, p. 414, Mar. 2016.
[10] Towards Data Science. (Nov. 15, 2009). *A Beginners Guide to Q-Learning*. Accessed: May 12, 2020. [Online]. Available: https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c
[11] T. Hu and Y. Fei, "QELAR: A Q-learning-based energy-efficient and lifetime-aware routing protocol for underwater sensor networks," in *Proc. IEEE Int. Perform., Comput. Commun. Conf.*, Austin, TX, USA, Dec. 2008, pp. 247–255.
[12] Z. Jin, Y. Ma, Y. Su, S. Li, and X. Fu, "A Q-learning-based delay-aware routing algorithm to extend the lifetime of underwater sensor networks," *Sensors*, vol. 17, no. 7, p. 1660, Jul. 2017.
[13] S. Wang and Y. Shin, "Efficient routing protocol based on reinforcement learning for magnetic induction underwater sensor networks," *IEEE Access*, vol. 7, pp. 82027–82037, 2019.
[14] X. Li, X. Hu, W. Li, and H. Hu, "A multi-agent reinforcement learning routing protocol for underwater optical sensor networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–7.
[15] N. Javaid, T. Hafeez, Z. Wadud, N. Alrajeh, M. S. Alabed, and N. Guizani, "Establishing a cooperation-based and void node avoiding energy-efficient underwater WSN for a cloud," *IEEE Access*, vol. 5, pp. 11582–11593, 2017.
[16] Y. Su, R. Fan, X. Fu, and Z. Jin, "DQELR: An adaptive deep Q-network-based energy- and latency-aware routing protocol design for underwater acoustic sensor networks," *IEEE Access*, vol. 7, pp. 9091–9104, 2019.
[17] Z. Fang, J. Wang, C. Jiang, B. Zhang, C. Qin, and Y. Ren, "QLACO: Q-learning aided ant colony routing protocol for underwater acoustic sensor networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Seoul, South Korea, May 2020, pp. 1–6.
[18] M. Khalid, F. Ahmad, M. Arshad, W. Khalid, N. Ahmad, and Y. Cao, "E2MR: Energy efficient multipath routing protocol for underwater wireless sensor networks," *IET Netw.*, vol. 8, no. 5, pp. 321–382, 2019.
[19] Y. Li, S. Wang, C. Jin, Y. Zhang, and T. Jiang, "A survey of underwater magnetic induction communications: Fundamental issues, recent advances, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 3, pp. 2466–2487, 3rd Quart., 2019.
[20] J. I. Agbinya, "A magneto-inductive link budget for wireless power transfer and inductive communication systems," *Prog. Electromagn. Res. C*, vol. 37, pp. 15–28, 2013.
[21] L. Yan, D. Wei, M. Pan, and J. Chen, "Downhole wireless communication using magnetic induction technique," United States Nat. Committee URSI Nat. Radio Sci. Meeting (USNC-URSI NRSM), Boulder, CO, USA, Tech. Rep., 2018.
[22] Z. Sun and I. F. Akyildiz, "Magnetic induction communications for wireless underground sensor networks," *IEEE Trans. Antennas Propag.*, vol. 58, no. 7, pp. 2426–2435, Jul. 2010.
[23] M. Zorzi, P. Casari, N. Baldo, and A. F. Harris, "Energy-efficient routing schemes for underwater acoustic networks," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 9, pp. 1754–1766, Dec. 2008.
[24] W. Guo and W. Zhang, "A survey on intelligent routing protocols in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 38, pp. 185–201, Feb. 2014.
[25] M. Adnan, L. Yang, T. Ahmad, and Y. Tao, "An unequally clustered multi-hop routing protocol based on fuzzy logic for wireless sensor networks," *IEEE Access*, vol. 9, pp. 38531–38545, 2021.
[26] M. A. Mazaideh and J. Levendovszky, "A multi-hop routing algorithm for WSNs based on compressive sensing and multiple objective genetic algorithm," *J. Commun. Netw.*, vol. 23, no. 2, pp. 138–147, Apr. 2021.
[27] J. Bell, "What is machine learning," in *Machine Learning: Hands-On for Developers and Technical Professionals*. Hoboken, NJ, USA: Wiley, 2020, pp. 1–2.
[28] C. Shyalika. (Nov. 15, 2009). *A Beginners Guide to Q-Learning*. Towards Data Science. Accessed: May 12, 2020. [Online]. Available: https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c
[29] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. London, U.K.: MIT Press, 2015.
[30] J. Greaves. (Nov. 9, 2017). *Understanding RL: The Bellman Equations*. Accessed: May 14, 2020. [Online]. Available: https://joshgreaves.com/reinforcement-learning/understanding-rl-the-bellman-equations/
[31] G. Abhijit, "Reinforcement learning: A tutorial survey and recent advances," *INFORMS J. Comput.*, vol. 21, pp. 178–192, May 2009.
[32] A. Parkinson. (Dec. 2, 2019). *The Epsilon-Greedy Algorithm for Reinforcement Learning*. Accessed: Jun. 1, 2020. [Online]. Available: https://medium.com/analytics-vidhya/the-epsilon-greedy-algorithm-for-reinforcement-learning-5fe6f96dc870
[33] L. Matignon, G. Laurent, and N. L. Fort, "Reward function and initial values: Better choices for accelerated goal-directed reinforcement learning," in *Proc. Int. Conf. Artif. Neural Netw.*, vol. 1, 2006, pp. 840–849.
[34] Z. Ying and S. Hongliang, "A clustered routing protocol for underwater wireless sensor networks," in *Proc. 34th Chin. Control Conf. (CCC)*, Hangzhou, China, Jul. 2015, pp. 7665–7670.
[35] H. Ismkhan, "Ik-means−+: An iterative clustering algorithm based on an enhanced version of the k-means," *Pattern Recognit.*, vol. 79, pp. 402–413, Jul. 2018.
[36] L. Morissette and S. Chartier, "The k-means clustering technique: General considerations and implementation in mathematica," *Tuts. Quant. Methods Psychol.*, vol. 9, no. 1, pp. 15–24, Feb. 2013.
[37] Scikit-Learn. *Sklearn Metrics Silhouette_Score*. Accessed: Oct. 5, 2020. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html
[38] K. Mahendru, (Jun. 17, 2019). *How to Determine the Optimal K for K-Means*. Accessed: Oct. 6, 2020. [Online]. Available: https://medium.com/analytics-vidhya/how-to-determine-the-optimal-k-for-k-means708505d204eb#:~:text=End%20Notes,for%20finding%20the%20optimal%20K
[39] Z. Rahman, F. Hashim, M. F. A. Rasid, and M. Othman, "Totally opportunistic routing algorithm (TORA) for underwater wireless sensor network," *PLoS ONE*, vol. 13, no. 6, Jun. 2018, Art. no. e0197087.

L. ALSALMAN received the B.E. degree in computer engineering from Kuwait University, in 2015, where she is currently pursuing the M.S. degree with the Department of Computer Engineering. Her research interests include designing routing protocols for sensor networks and reinforcement learning.

E. ALOTAIBI received the B.S. (Hons.) and M.S. degrees in computer engineering from Kuwait University, Kuwait, in 1999 and 2004, respectively, and the Ph.D. degree in computer science from the University of California at Davis, Davis, USA, in 2010. She is currently an Assistant Professor with the Department of Computer Engineering, Kuwait University. Her research interests include cross-layer design, routing, and topology control over wireless mesh networks (WMNs). Recently, she has been working on routing and scheduling of software-defined networks (SDNs), in addition to underwater wireless sensor networks (UWSNs).