# Towards Intelligent and Adaptive Multi-Agent Slicing in Maritime Networks

Sahar Ammar[1], Wiem Abderrahim[2], and Basem Shihada[1]

**Abstract**—Integrated aerial-terrestrial maritime networks represent communication-constrained environments that often suffer from limited adaptability and interoperability. To overcome these limitations, we design an intelligent network slicing framework leveraging Open Radio Access Network (O-RAN) features. We propose an asymmetric-state hierarchical multi-agent reinforcement learning algorithm to achieve adaptive quality of service-aware maritime connectivity. By hierarchically structuring discrete and continuous agents, the algorithm handles the Virtual Network Function (VNF) orchestration actions, the RAN slicing actions, and their interdependency. The two agents are coordinated via a novel inter-agent communication module based on various proposed approaches designed to study the communication between the agents. The results demonstrate that the proposed solution outperforms single-agent benchmarks, achieving approximately 65% higher reward than the discrete baseline and over three-fold increase the reward of the continuous benchmark. It also highlights the key role of the inter-agent communication module in improving the agents' coordination and enhancing the overall performance.

**Index Terms**—Maritime communications, Network Slicing, Multi-Agent Reinforcement Learning (MARL), Open Radio Access Network (O-RAN), Unmanned Aerial Vehicle (UAV).

## 1 Introduction

Over the past few decades, maritime activities have evolved from traditional fisheries and aquaculture to encompass marine tourism, ocean exploration, and emerging data-intensive applications. This evolution necessitates ubiquitous and reliable connectivity capable of meeting heterogeneous Quality of Service (QoS) requirements across diverse use cases such as maritime entertainment, autonomous vessels, marine robotics, and real-time monitoring. For instance, maritime communications systems are expected to offer high data rate connectivity for video streaming users simultaneously with low-delay links for robotic users. However, maritime communications face multiple intrinsic challenges such as large sea regions, scattered end-users and sparse infrastructure [1]. Conventional maritime networks rely on on-shore base stations that offer basic services with limited reach. Although satellite communications have been adopted to extend maritime coverage, these systems often exhibit high latency and require large antennas. They also incur substantial operational costs that cannot be supported by small vessels and low-budget users in remote regions [1]. As maritime communications progressively evolves toward the 6G era, unmanned aerial vehicles (UAVs) have emerged as a promising solution since they are cost-effective, adaptable, and easy to deploy [1]–[5]. UAVs can be used as aerial relays that connect maritime vessels and on-shore base stations to extend the footprint and enhance the communication reliability across wide sea regions.

Nonetheless, the resulting maritime aerial–terrestrial architectures are heterogeneous and often constrained by proprietary, closed implementations, limiting interoperability and scalability. Consequently, efficient maritime network slicing mechanisms are required to support diverse maritime applications. To address these limitations, the Open Radio Access Network (O-RAN) paradigm can be leveraged offering a suitable foundation for network slicing. Through its principles of virtualization, openness, and intelligence, O-RAN enables the deployment and management of multiple network slices over shared infrastructure, supporting maritime services with heterogeneous QoS requirements. In particular, RAN virtualization facilitates the deployment and orchestration of Virtual Network Functions (VNFs) in a flexible and slice-oriented ecosystem, while the disaggregation of RAN functionalities enables multi-vendor interoperability [6]–[9]. Moreover, the O-RAN architecture incorporates software-defined control through the Non-Real-Time (Non-RT) and Near-Real-Time (Near-RT) RAN Intelligent Controllers (RICs). These controllers provide native support for data-driven and AI-assisted decision-making, promoting intelligent and dynamic VNF orchestration and RAN resource management. In addition, the O-RAN open interfaces (e.g., A1, E2, and O1) provide standardized links that facilitate information exchange and data collection for the training and inference of DRL-based network management algorithms. In this context, we design an AI-enabled network slicing framework, built upon the O-RAN architecture, for VNF orchestration and RAN resource management in 6G integrated aerial-terrestrial maritime networks. The proposed framework employs a hierarchical multi-agent reinforcement learning (RL) algorithm to achieve adaptive, seamless, and QoS-aware maritime connectivity.

*Sahar Ammar and Basem Shihada are with CEMSE Division, King Abdullah University of Science and Technology (KAUST), Thuwal, Makkah Province, Saudi Arabia. Wiem Abderrahim is with University of Carthage, Ecole Supérieure des Communications de Tunis (Sup'Com), LR11TIC05, Réseaux Radio-Mobiles Multimédia (MEDIATRON) Lab and University of Gabes, Ecole Nationale d'Ingénieurs de Gabès (ENIG), Tunisia. E-mail: {sahar.ammar, wiem.abderrahim, basem.shihada}@kaust.edu.sa.*

### 1.1 Related Work

Although network slicing is a well-investigated subject in terrestrial networks [10], [11], its adoption in integrated aerial-terrestrial networks, particularly in maritime networks, is still in its infancy. In fact, the concepts of network slicing are rarely explored in maritime communications. However, few studies can be reported where researchers designed network architectures based on the enabling technologies of slicing namely software defined networking (SDN) and network function virtualization (NFV). For instance, an SDN/NFV-based architecture is proposed for opto-acoustic underwater networks in [12], and for maritime logistics and transport in [13]. Despite these contributions, multiple issues associated with maritime slicing, such as RAN resource slicing and VNF deployment, remain unexamined. Furthermore, since the solutions for these problems are typically developed for terrestrial networks; efforts were dedicated to investigate the application of slicing technologies in aerial-terrestrial networks [14]. For example, the authors of [15] explore radio resource slicing optimization for UAV-based vehicular networks using a Long Short-Term Memory algorithm. In [16], a RAN slicing scheme was developed based on successive convex approximation to optimize the transmit power and the throughput of enhanced Mobile Broadband slice. In addition, researchers in [17] examine the VNF placement problem in vehicular aerial-terrestrial networks using a heuristic method for revenue maximization. Meanwhile, other studies employ RL algorithms for VNF deployment optimization [18]–[20]. For instance, a federated Deep Q-Network (DQN) was employed in [20] to minimize age of information and energy consumption in mobile edge computing-based UAV networks. The authors of [18] design a VNF placement and migration with UAV positioning algorithm based on multi-agent RL. They maximize the energy efficiency and minimize the delay using DQN and Deep Deterministic Policy Gradient (DDPG) to deal with discrete and continuous variables of the optimization problem. Furthermore, the joint RAN slicing and VNF placement problem is considered in [21] for integrated aerial-terrestrial networks. An iterative algorithm was proposed to optimize computing resource utilization efficiency for three 5G slices.

Despite these advances, most existing studies either consider the problems of RAN slicing and VNF deployment in a separate manner, or focus on the issue of VNF placement, neglecting the problem of joint RAN slicing and VNF scaling. To address these research gaps, we studied the problem of joint RAN slicing and VNF deployment, particularly VNF scaling and VNF migration, in our prior work [22]. Specifically, we developed an AI-based maritime-oriented network slicing framework for integrated aerial-terrestrial networks, where we focused on maximizing the energy efficiency under resource and QoS requirements constraints. The formulated problem involved two types of variables i.e. continuous variables for RAN resource slicing and discrete variables for VNF scaling and migration. However, the solution proposed in [22] adopted a single-agent approach where the continuous actions were quantized to unify the action space and enable the use of Deep RL (DRL) algorithms. Although such single-agent method presents a simple solution to the mixed-integer problem, the discretization step of the continuous variables can impact the optimality of the agent causing a degradation in its performance. Additionally, the design of the continuous actions quantization creates a tradeoff between the accuracy of the agent's decisions and the simplicity of its action space. To overcome these issues, we develop, in this work, a multi-agent DRL framework based on Dueling Double Deep Q-Network (D3QN) and Twin Delayed DDPG (TD3) to deal with the discrete and continuous actions, respectively. Although multi-agent RL techniques have been widely applied to network slicing, the impact of inter-agent interactions on the overall algorithmic performance has been rarely investigated [18], [23]–[26]. Therefore, further research is required to develop intelligent, adaptive, and cooperative solutions that jointly optimize RAN slicing and VNF orchestration in integrated aerial–terrestrial maritime networks.

### 1.2 Main Contributions

In this paper, we study the problem of joint RAN slicing and VNF orchestration in integrated aerial-terrestrial maritime networks. We leverage O-RAN features, including virtualization and intelligence, to enable the deployment of network slicing. Such features promote seamless maritime connectivity while meeting the diverse QoS demands of different slices. Additionally, we take into account the properties of maritime integrated networks, including the clustered distribution, the low density of maritime users, and the restricted resources of virtualized network nodes (UAVs and marine buoys). We formulate the optimization problem that maximizes the resource utilization, while satisfying the QoS requirements of the desired slices, i.e. a low-latency slice and a high-throughput slice, as well as the resource constraints. The formulated problem involves two types of variables: continuous variables for RAN resource slicing and discrete variables for VNF orchestration.

To overcome the limitations of single-agent solutions, we design an asymmetric-state hierarchical multi-agent reinforcement learning (ASH-MARL) algorithm for maritime slicing. The proposed framework is based on two specialized agents, namely a D3QN agent that handles the discrete VNF orchestration actions and a TD3 agent that manages the continuous RAN slicing actions. In addition, the two agents are hierarchically structured and coordinated via a novel inter-agent communication module, enabling the framework to capture the dependency between the discrete orchestration decisions and the continuous slicing decisions. This module defines the communication between the two agents in our multi-agent setting and enables their coordination. Specifically, it allows us to investigate the impact of the environment state and the D3QN actions on the TD3 agent, and thereby the performance of the multi-agent algorithm. We examine four communication approaches; (i) an independent approach which is a baseline where the two agents act independently, (ii) an augmentation technique where the environment state is augmented with the D3QN actions to form the TD3 state, (iii) a transformation scheme where the TD3 state is obtained by transforming the environment state using the D3QN actions, and (iv) a hybrid approach that combines both augmentation and transformation methods. The main contributions of this work can be summarized as follows:

- We develop an intelligent network slicing framework for integrated aerial-terrestrial maritime networks based on O-RAN concepts. Considering the properties of such networks, we use RAN virtualization to dynamically scale the VNFs and slice the RAN resources to fulfill the needs of two distinct slices.
- We propose the ASH-MARL algorithm, based on D3QN and TD3, to deal with the discrete and continuous variables and handle their inter-dependency. The framework includes a shared replay buffer based on prioritized experience replay (PER) to improve learning stability and efficiency.
- We design a novel inter-agent communication module to study the communication between the two agents by examining four different approaches that describe the impact of the environment state and the D3QN actions on the TD3 agent, and thereby the performance of the multi-agent framework.
- Our findings demonstrate that the ASH-MARL solution outperforms single-agent benchmarks, and highlights the key role of the designed inter-agent communication module in improving the agents' coordination and enhancing the overall performance.

The rest of this paper is organized as follows; the system model is detailed in Section 2 and the joint VNF orchestration and RAN slicing problem is formulated in Section 3. Section 4 is dedicated for the ASH-MARL framework proposed to tackle the formulated problem. Finally, we present and analyze our results in Section 5 and conclude in Section 6.

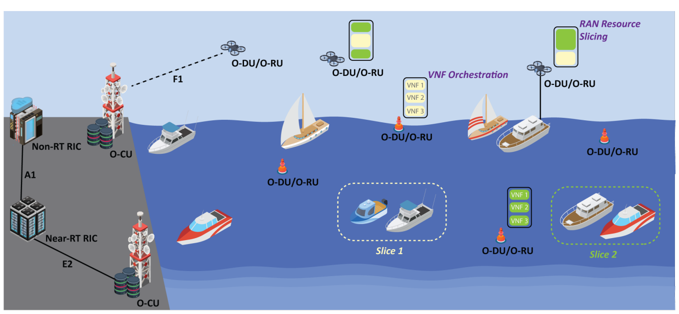



Fig. 1: Illustration of Integrated Aerial- Maritime Network Architecture.

## 2 System Model

The proposed integrated aerial-maritime architecture consists of marine buoys, tethered and non-tethered UAVs acting as RAN nodes, offering connectivity to maritime clustered end-users, as illustrated in Fig.1. The users in the maritime environment typically have a sparse distribution in the large marine area and are simultaneously grouped in small regions i.e. ships and marine vessels, which form dispersed clusters [1]. Thus, we leverage this clustered distribution and model ships as end-users to reduce problem complexity[1]. In addition, such maritime characteristics, namely sparse user distribution and low number of RAN nodes, result in negligible inter-cell interference. We further assume that interference management techniques are implemented to mitigate any residual effects. Moreover, we consider two network slices defined according to their distinct QoS requirements, namely a high-throughput slice (slice 1) and a low-latency slice (slice 2), with a predetermined ship-to-slice association. While the two slices are selected as a representative case study of maritime slicing, the proposed framework can support a larger number of slices, each associated with distinct throughput and latency requirements according to the targeted services.

| Parameter | Description |
|---|---|
| $T$ | Set of time slots |
| $S$ | Set of network slices |
| $U$ | Set of non-tethered UAVs |
| $U_{th}$ | Set of tethered UAVs |
| $B$ | Set of marine buoys |
| $K_s$ | Set of ships belonging to slice $s$ |
| $X_{\kappa_s}(t)$ | Position vector of ship $\kappa_s$ at time slot $t$ |
| $X_{\xi}(t)$ | Position vector of node $\xi \in N = \{U, U_{th}, B\}$ at time slot $t$ |
| $C_{\xi}$ | Maximum CPU capacity at node $\xi \in N$ |
| $P_{\xi}$ | Maximum transmission power at node $\xi \in N$ |
| $B_{\xi}$ | Bandwidth at node $\xi$ |
| $c_i$ | CPU capacity needed for the deployment of one VNF instance $i$ |
| $Q_{P,i,s}$ | CPU capacity required by VNF instance $i$ serving $s$ |
| $Q_{T,i,s}$ | Size of data transmitted by VNF instance $i$ serving $s$ |
| $R_{s,min}$ | Minimum achievable throughput for slice $s$ |
| $L_{s,max}$ | Maximum latency requirement for slice $s$ |
| $\delta_{\xi,i,s}(t)$ | Placement of VNF instance $i$ deployed at node $\xi$ to support slice $s$ at time slot $t$ |
| $\upsilon_{\xi,s}(t)$ | Integer variable indicating the scaling of VNF instances at node $\xi$ serving slice $s$ at time slot $t$ |
| $\Upsilon_{\xi,s}(t)$ | Total number of VNF instances deployed at node $\xi$ serving slice $s$ at time slot $t$ |
| $c_{\xi,i,s}(t)$ | CPU capacity which is allocated to VNF instance $i$ at node $\xi$ serving slice $s$ at time slot $t$ |
| $p_{\xi,i,\kappa_s}(t)$ | Transmission power to be allocated to VNF instance $i$ at node $\xi$ supporting slice $s$ |

TABLE 1: Main parameters.

Furthermore, we incorporate the different components of the O-RAN architecture in the integrated aerial-terrestrial maritime networks [6], [7]. The O-RAN components are connected through standardized open interfaces that facilitate information exchange and control signaling. In fact, the open Radio Unit (O-RU) and Distributed Unit (O-DU) offer low-level functions and they can be both placed in the marine buoys and the UAVs to minimize network delays. Using the F1 interface for its connection to the O-DUs, the open Centralized Unit (O-CU) deals with high-level functionalities. It can run on on-land edge servers with sufficient computing resources. Moreover, the Near-Real-Time RIC (Near-RT RIC) uses the E2 interface to connect to the O-CU/O-DU to exchange network data and control commands. Deployed on cloud/edge servers, the Near-RT RIC is responsible for RAN management and control. It enforces the AI-based policies obtained from the Non-RT

1. We note that this abstraction is commonly adopted in maritime communications [1], [27]–[29].

RIC, through the A1 interface. The Non-RT RIC runs on cloud servers and handles intelligent network optimization, policy management, and RAN analytics by supporting the training of AI models. Fig.1 depicts the main components and interfaces of the O-RAN architecture incorporated in the integrated network. Also, Table 1 gives a summary of the main parameters.

### 2.1 Channel Modeling

The proposed architecture includes two types of channels; Air-to-Sea (AS) channel for the links between UAVs and end-users, and Sea-to-Sea (SS) channel for the links between buoys and end-users. The channel modeling involves the large-scale fading, and the small-scale fading [3], [30]. On the one hand, the path loss, defining the large-scale fading, for the AS channel is given by the Free-space path loss model [30],

$$\eta_{\mathrm{AS}}(t) = 10\alpha_{\mathrm{AS}} \log_{10}\left(\frac{4\pi d_{a,b}(t)}{\lambda}\right) \tag{1}$$

Meanwhile, the two-ray and three-ray models are used for large-scale fading of the SS channel given by [3],

$$\eta_{\mathrm{SS}}(t) = \begin{cases} 10\log_{10}\left(\left(2\eta_0 \sin\Delta_1\right)^2\right), d_{a,b}(t) \leq d_{\mathrm{br}} \\ 10\log_{10}\left(\left(2\eta_0\left(1+\Delta_2\right)\right)^2\right), d_{a,b}(t) \geq d_{\mathrm{br}} \end{cases} \tag{2}$$

where $\eta_0 = \frac{4\pi d_{a,b}(t)}{\lambda}$, $\Delta_1 = \frac{2\pi z_a z_b}{\lambda d_{a,b}(t)}$ $\Delta_2 = 2\sin\left(\frac{2\pi z_a z_b}{\lambda d_{a,b}(t)}\right)\sin\left(\frac{2\pi(z_l - z_a)(z_l - z_b)}{\lambda d_{a,b}(t)}\right)$, and $d_{\mathrm{br}} = \frac{4 z_a z_b}{\lambda}$ represents the boundary distance. Additionally, $z_a$ and $z_b$, $z_l$ denote the heights of the network nodes and the duct layer, and $d_{a,b}(t)$ is the distance between them. Moreover, $\alpha_{\mathrm{ch}}$ is the path loss exponent, and $\lambda = c/f_c$, where $c$ and $f_c$ represent the light velocity and the frequency. Hence, the path gain of the link is defined by:

$$h_{a,b}(t) = 10^{\frac{G_a + G_b - \eta_{\mathrm{ch}}(t)}{10}}, \quad \mathrm{ch} \in \{AS, SS\} \tag{3}$$

where $G_a$ and $G_b$ are the antenna gains of $a$ and $b$. On the other hand, the small-scale fading in the maritime environment is modeled by a Rician distribution, capturing the weak paths produced by the sea surface reflections [1], [2],

$$f_{\zeta_{a,b}(t)}(x) = \frac{x}{\sigma^2}\exp\left(\frac{-\left(x^2 + \mu_{a,b}^2\right)}{2\sigma^2}\right) I_0\left(\frac{x\mu_{a,b}}{\sigma^2}\right) \tag{4}$$

where $\mu_{a,b}^2(t) = P_a(t)\left(\frac{\lambda}{4\pi d_{a,b}(t)}\right)^{\alpha_{\mathrm{ch}}} G_a G_b$ is the average received power of the line-of-sight component, $2\sigma^2$ is the average received power of the multipath component, and $P_a(t)$ denotes the transmit power.

### 2.2 Slice Throughput and Latency

We use the throughput and the latency as slice QoS criteria for the desired slices. First, the throughput $R_{\xi,i,s}(t)$ offered by the VNF instance $i$ at node $\xi$ supporting slice $s$ at time $t$ is expressed as,

$$R_{\xi,i,s}(t) = \sum_{\kappa_s \in K_s} B_\xi log_2(1 + SNR_{\xi,i,\kappa_s}(t)) \tag{5}$$

where $B_\xi$ denotes the bandwidth at $\xi$, $K_s$ is the set of ships belonging to slice $s$, and $SNR_{\xi,i,\kappa_s}(t)$ represents the SNR per ship given by,

$$SNR_{i,\kappa_s}^{\xi}(t) = \frac{h_{\xi,\kappa_s}(t)\zeta_{\xi,\kappa_s}^2(t)p_{\xi,i,\kappa_s}(t)}{B_\xi N_0} \tag{6}$$

where $p_{\xi,i,\kappa_s}(t)$ is the transmission power that is allocated to VNF instance $i$ at $\xi$ serving ship $\kappa_s$ of slice $s$ at time $t$. Additionally, $N_0$ and $\zeta_{\xi,\kappa_s}(t)$ represent the noise power spectral density and the Rician fading factor.

Furthermore, the transmission latency $L_{\xi,i,s}^T(t)$ and the processing latency$L_{\xi,i,s}^P(t)$ of VNF instance $i$ at node $\xi$ supporting slice $s$ at time $t$ are given by,

$$L_{\xi,i,s}^T(t) = \frac{Q_{T,i,s}}{R_{\xi,i,s}(t)}, \quad L_{\xi,i,s}^P(t) = \frac{Q_{P,i,s}}{c_{\xi,i,s}(t) + c_i} \tag{7}$$

where $c_{\xi,i,s}(t)$ is the CPU capacity which is allocated to VNF instance $i$ at node $\xi$ serving slice $s$ at time slot $t$, and $c_i$ is the CPU capacity needed for the deployment of one VNF instance $i$. In addition, $Q_{P,i,s}$ denotes the CPU capacity required for slice $s$ and $Q_{T,i,s}$ is the size of data transmitted by instance $i$ serving $s$. Thus, the total latency $L_{\xi,i,s}(t)$ of VNF instance $i$ at $\xi$ supporting slice $s$ at $t$ is,

$$L_{\xi,i,s}(t) = L_{\xi,i,s}^T(t) + L_{\xi,i,s}^P(t) \tag{8}$$

## 3 Joint VNF Orchestration and RAN Slicing Problem

In this section, we formulate the optimization problem that jointly considers the slicing of RAN resources and the scaling of VNF instances. Hence, the optimization variables consist of the integer scaling variable $\upsilon_{\xi,s}(t)$, which determines the number of VNF instances to be added or removed at network node $\xi$ serving slice $s$ at time slot $t$, the CPU capacity $c_{\xi,i,s}(t)$ and the transmission power $p_{\xi,i,\kappa_s}(t)$ to be allocated to VNF instance $i$ at virtualized node $\xi$ supporting slice $s$.

### 3.1 Constraints

Several constraints should be fulfilled in order to make the appropriate decisions of VNF orchestration and RAN slicing to serve both of the desired slices. These constraints involve communication and computing resources, ensuring that the consumed resources remain within the available capacities at each virtualized node $\xi$, given by,

$$C_1: \sum_{s\in S}\sum_{i\in\{1,\ldots\Upsilon_{\xi,s}(t)\}} \delta_{\xi,i,s}(t)\cdot(c_{\xi,i,s}(t) + c_i) \leq C_\xi \tag{9}$$

$$C_2: \sum_{s\in S}\sum_{i\in\{1,\ldots\Upsilon_{\xi,s}(t)\}}\sum_{\kappa_s\in K_s} \delta_{\xi,i,s}(t)\cdot p_{\xi,i,\kappa_s}(t) \leq P_\xi \tag{10}$$

where $S$ is the number of slices and $\delta_{\xi,i,s}(t)$ indicates the placement of VNF instances and equals to 1 if VNF instance $i$ is deployed at node $\xi$ to support slice $s$ at time slot $t$ and 0 otherwise. Additionally, $\Upsilon_{\xi,s}(t)$ denotes the total number of VNF instances deployed at node $\xi$ serving slice $s$ at time slot $t$ and depends on the scaling variable according to $\Upsilon_{\xi,s}(t) = \Upsilon_{\xi,s}(t-1) + \upsilon_{\xi,s}(t)$. Also, $P_\xi$ and $C_\xi$ represent the maximum transmission power and CPU capacity available at $\xi$. Moreover, slice-level throughput and latency QoS constraints should be satisfied to accommodate the needs of the target slices $s \in S$:

$$C_3 : \sum_{\xi \in N} \sum_{i \in \{1, \dots \Upsilon_{\xi,s}(t)\}} \delta_{\xi,i,s}(t) \cdot R_{\xi,i,s}(t) \geq R_{s,min}, \quad (11)$$

$$C_4 : \sum_{\xi \in N} \sum_{i \in \{1, \dots \Upsilon_{\xi,s}(t)\}} \delta_{\xi,i,s}(t) \cdot L_{\xi,i,s}(t) \leq L_{s,max}, \quad (12)$$

where $N = \{U, U_{th}, B\}$, $U$, $U_{th}$, and $B$ denote the sets of non-tethered UAVs, tethered UAVs and marine buoys. Additionally, $L_{s,max}$ and $R_{s,min}$ represent the maximum latency and the minimal achievable throughput for slice $s$.

### 3.2 Problem Formulation

While satisfying the QoS demands of the two slices, we aim to maximize the resource utilization $\Phi_{\text{RU}}(t)$ of the network, defined as follows,

$$\Phi_{\text{RU}}(t) = \frac{1}{|N|} \sum_{\xi \in N} \omega \cdot \varphi_{\xi}^{c}(t) + (1 - \omega) \cdot \varphi_{\xi}^{p}(t) \quad (13)$$

where $\omega$ is a weight factor balancing between the resource utilization of the computing and transmit power resources. $\varphi_{\xi}^{c}(t)$ and $\varphi_{\xi}^{p}(t)$ denote the resource utilization of the two types of resources at location $\xi$, given by,

$$\varphi_{\xi}^{c}(t) = \frac{\sum_{s \in S} \sum_{i \in \{1, \dots \Upsilon_{\xi,s}(t)\}} \delta_{\xi,i,s}(t) \cdot (c_{\xi,i,s}(t) + c_i)}{C_{\xi}} \quad (14)$$

$$\varphi_{\xi}^{p}(t) = \frac{\sum_{s \in S} \sum_{i \in \{1, \dots \Upsilon_{\xi,s}(t)\}} \sum_{\kappa_s \in K_s} \delta_{\xi,i,s}(t) \cdot p_{\xi,i,\kappa_s}(t)}{P_{\xi}} \quad (15)$$

Therefore, the optimization problem formulated for the joint RAN slicing and VNF orchestration is given by,

$$(\mathbf{P}) : \max_{\upsilon_{\xi,s}(t), c_{\xi,i,s}(t), p_{\xi,i,\kappa_s}(t)} \frac{1}{|T|} \sum_{t \in T} \Phi_{\text{RU}}(t) \quad (16a)$$

$$s.t \quad C_1 - C_4$$

$$\upsilon_{\xi,s}(t) \in \mathbb{Z}, \quad c_{\xi,i,s}(t) \in \mathbb{R}, \quad p_{\xi,i,\kappa_s}(t) \in \mathbb{R} \quad (16b)$$

where $T$ is the set of time slots. This optimization horizon is introduced to capture the temporal dynamics of the maritime network resulting from ship mobility, UAV mobility, and VNF orchestration variations. The formulated optimization problem is NP-hard since it is a mixed-integer non-linear optimization problem with numerous variables and constraints. In addition, the dependency between the discrete and continuous optimization variables further increases the complexity of the problem. In fact, the continuous variables including the CPU capacity $c_{\xi,i,s}(t)$ and the transmission power $p_{\xi,i,\kappa_s}(t)$ depend on the VNF instances placement $\delta_{\xi,i,s}(t)$ and their total number $\Upsilon_{\xi,s}(t)$, which in turn depend on the integer scaling variable $\upsilon_{\xi,s}(t)$. Hence, using traditional optimization techniques is no longer sufficient to offer the required optimality and efficiency. Thus, we design an asymmetric-state hierarchical multi-agent reinforcement learning (ASH-MARL) framework to solve the optimization problem.

## 4 Proposed Asymmetric-State Hierarchical Multi-agent RL Framework

In this section, we outline the asymmetric-state hierarchical multi-agent RL framework proposed to tackle the joint VNF orchestration and RAN slicing problem.

### 4.1 Overview of Reinforcement Learning

RL represents a category of machine learning algorithms that learn through the interaction with an environment, aiming to maximize a cumulative reward [31]. Two main classes of RL algorithms can be distinguished namely value-based approaches (e.g. Q-learning) and policy gradient methods (e.g. REINFORCE). While the former employs value functions to learn a policy, the latter directly optimizes the parameterized function that represents the policy. However, traditional RL methods can not handle dynamic environments and high-dimensional state and action spaces. Hence, DRL was developed by merging sequential learning with deep neural networks (DNNs) [32]. For instance, DQN [32] was proposed to extend the applications of Q-learning to high-dimensional state spaces by utilizing DNNs as value function approximators. Meanwhile, DDPG [33] was introduced based on the actor-critic architecture and the deterministic policy gradient concept for continuous action spaces. Although DQN and DDPG are considered significant advancements in RL compared to traditional methods, they still suffer from several issues. Such issues motivated the development of multiple variants of DQN and DDPG such as Double DQN (DDQN), Dueling Double Deep Q-Network (D3QN), and Twin Delayed DDPG (TD3). In fact, D3QN combines the concepts of double Q-networks and dueling architecture to mitigate the limitations of DQN [34], [35]. On the one hand, the double DQN component addresses the issue of overestimation bias by decoupling action selection from evaluation using two separate Q-networks. This decoupling enhances the accuracy of Q-value estimation and consequently the training stability. On the other hand, the dueling DQN component improves the learning efficiency through the decomposition of the Q-value into a state-value stream estimating the value of being in a state and an advantage stream estimating the quality of an action in that state. By combining these two concepts, D3QN offers enhanced robustness, stability, and efficiency in discrete action spaces. Similarly, TD3 builds on the DDPG algorithm to improve stability for continuous tasks [36]. Specifically, while the actor-critic architecture of DDPG includes an actor network for action selection and a critic network for Q-value estimation, TD3 employs two critic networks to reduce the overestimation bias of DDPG. Additionally, TD3 adopts delayed policy updates for training stabilization, where the frequency of actor updates is lower than that of the critics. TD3 also introduces target policy smoothing, which adds clipped noise to the actions during target Q-values computation to regularize their estimates [37]. These improvements allow TD3 to achieve improved performance, higher training stability and sample efficiency compared to DDPG for continuous actions. As a result, we design our ASH-MARL framework using the D3QN algorithm for discrete VNF scaling decisions, and the TD3 method for the continuous RAN slicing actions.

## 4.2 Markov Decision Process Formulation

We define the Markov Decision Process (MDP) formulation of the optimization problem in (16) as follows:

- The state space: The state representation $s_t$ at timestep $t$ consists of the information on the VNF instances, including their placements $\delta_{\xi,i,s}(t)$, and their total number at each virtualized node $\Upsilon_{\xi,s}(t)$. The state also includes information on the network nodes and the end-users, specifically their position vectors $X_\xi(t)$ and $X_{\kappa_s}(t)$, as well as a binary variable $\iota_{\kappa_s}(t)$ indicating the slice $s$ to which the ship belongs. Therefore, the state can be expressed as:

$$s_t = \begin{bmatrix} \delta_{\xi,i,s}(t),\ i \in \{1, ...\Upsilon_{\xi,s}(t)\}, \xi \in N, s \in S \\ \Upsilon_{\xi,s}(t),\ \xi \in N, s \in S \\ X_\xi(t),\ \xi \in N \\ X_{\kappa_s}(t),\ \kappa_s \in K_s, s \in S \\ \iota_{\kappa_s}(t),\ \kappa_s \in K_s, s \in S \end{bmatrix} \tag{17}$$

- The action space: The action space $\mathcal{A}$ includes both discrete and continuous variables. The discrete actions $a_{d_t}$ correspond to the VNF scaling decisions:

$$a_{d_t} = [\upsilon_{\xi,s}(t),\ \xi \in N,\ s \in S] \tag{18}$$

The continuous actions $a_{c_t}$ correspond to the RAN slicing decisions, i.e. the resource slicing of the transmission power and the CPU capacity:

$$a_{c_t} = \begin{bmatrix} c_{\xi,i,s}(t),\ i \in \{1, ...\Upsilon_{\xi,s}(t)\}, \xi \in N, s \in S \\ p_{\xi,i,\kappa_s}(t),\ i \in \{1, ...\Upsilon_{\xi,s}(t)\}, \xi \in N, \kappa_s \in K_s, s \in S \end{bmatrix} \tag{19}$$

Hence, the DRL agent's actions at each timestep $t$ are $a_t = (a_{d_t}, a_{c_t})$, consisting of a combination of the two types of actions.

- The reward function: The reward function $r_t$ is designed through a weighted penalty approach [38], targeting the resource utilization maximization and the satisfaction of the resource and QoS requirements, as follows,

$$r_t = \Omega_{\text{obj}} \cdot \Phi_{\text{RU}}(t) - \sum_{\vartheta \in \Theta} \Omega_c \cdot max\{0, \vartheta(t) - \vartheta_{max}\} \tag{20}$$

where $\vartheta(t)$ and $\vartheta_{\max}$ denote the constraints when expressed in form of $\vartheta(t) \leq \vartheta_{\max}$, and $\Theta$ represent the set of constraints $C_1 - C_4$. In addition, $\Omega_{\text{obj}}$ and $\Omega_c$ are the weights that allows the agent to balance between the objective function maximization and the constraints satisfaction.

## 4.3 ASH-MARL-based solution

### 4.3.1 *Outline of the ASH-MARL Framework*

The main components of the proposed ASH-MARL framework, illustrated in Fig.2, are the discrete D3QN agent and the continuous TD3 agent, indicating their action selection mechanism and their target value and loss computations. It also includes the prioritized experience replay (PER) replay buffer and the inter-agent communication module. On the one hand, the D3QN agent, which makes the VNF orchestration decisions, employs two neural networks namely the online Q-network for action selection and the target Q-network for Q-value evaluation [34]. D3QN also uses the dueling architecture to compute the Q-values by decomposing it into state-value and advantage streams, thereby improving learning efficiency [35]. On the other hand, the TD3 agent, which takes the RAN slicing actions, utilizes an actor network that selects actions through a deterministic policy and two critic networks to estimate the Q-values [36]. Unlike DDPG, TD3 adopts delayed updates where the policy (i.e. the actor network) is updated less frequently than the Q-function (i.e. the critic networks). Moreover, the PER replay buffer $\mathcal{D}$ is used to store the interactions between the environment and the ASH-MARL framework. During training, batches of these transitions are sampled using the PER mechanism to update the networks [39]. This sampling approach improves the learning efficiency and stability of the agent compared to the uniform sampling of standard replay buffers. Furthermore, the D3QN and TD3 agents are hierarchically structured and coordinated via the inter-agent communication module. This coordination allows the ASH-MARL algorithm to capture the dependency between the orchestration and slicing actions, while preserving a constant TD3 action space dimension. We note that although the D3QN agent dynamically scales the number of VNFs, the TD3 action space dimensionality remains constant. This condition is necessary since dynamically changing the TD3 action space would be incompatible with standard DRL algorithms, which rely on fixed-dimensional spaces for the actor and critic networks as well as the replay buffer. Instead, the dependency is captured through the hierarchical interaction between the two agents and the proposed inter-agent communication module. Specifically, the module determines how the environment state $s_t$ and the discrete actions $a_{d_t}$ of the D3QN agent impact the state of the TD3 agent $s_t^{TD3}$. We explore four different approaches to describe this interaction:

i) Independent approach $(n = 1)$: The D3QN actions do not impact the TD3 state which remains equal to the environment state $s_t$.
ii) Augmentation approach $(n = 2)$: The environment state is augmented with the D3QN actions and the TD3 state is a concatenation of $s_t$ and $a_{d_t}$.
iii) Transformation approach $(n = 3)$: The D3QN actions are used to transform the environment state. Specifically, the D3QN actions determine the number of VNF instances that should be added or removed at network node $\xi$ to serve slice $s$. Hence, they have a direct effect on the VNF instances placement $\delta_{\xi,i,s}(t)$ and their total number $\Upsilon_{\xi,s}(t)$, which are part of $s_t$. Consequently, the TD3 state is a function $f$ of $s_t$ and $a_{d_t}$, describing this effect.
iv) Hybrid approach $(n = 4)$: The TD3 state is obtained using both augmentation and transformation approaches.

To represent the communication between the discrete and continuous agents, we define the following function $\mathcal{H}_n[s_t, a_{d_t}]$:

$$\mathcal{H}_n[s_t, a_{d_t}] = \begin{cases} s_t, & n = 1 \\ [s_t, a_{d_t}], & n = 2 \\ f(s_t, a_{d_t}), & n = 3 \\ [f(s_t, a_{d_t}), a_{d_t}], & n = 4 \end{cases} \tag{21}$$

where $n$ indicates the type of inter-agent communication.

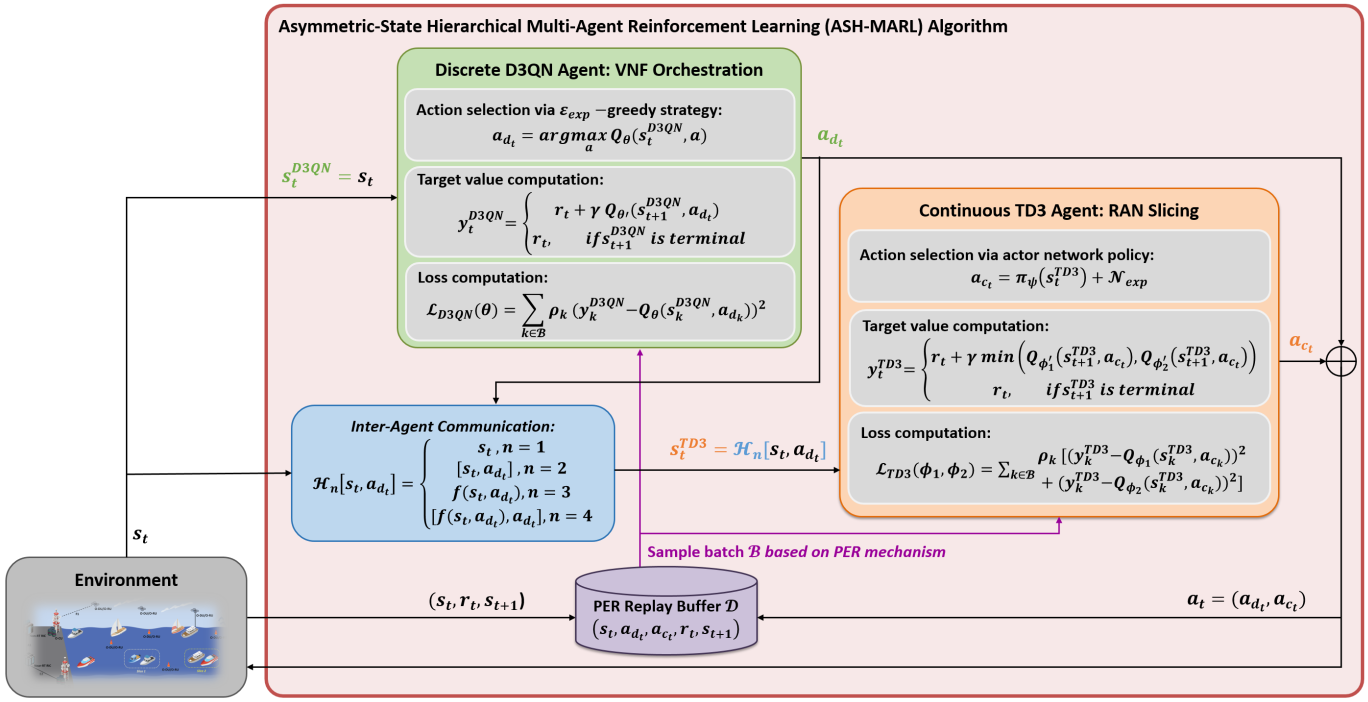



Fig. 2: Asymmetric-State Hierarchical Multi-Agent Reinforcement Learning (ASH-MARL) Framework.

#### *4.3.2 Training Process*

The ASH-MARL algorithm is presented in Algorithm 1. Given the current state $s_t$, the ASH-MARL algorithm chooses an action $a_t$ at each timestep $t$. The selected action is a combination of the discrete and continuous actions chosen by the D3QN and the TD3 agents, respectively. First, the D3QN agent selects the discrete VNF scaling actions $a_{d_t}$ based on a $\varepsilon-$greedy mechanism, given the state $s_t^{D3QN} = s_t$ as follows:

$$a_{d_t} = \begin{cases} \text{random action from } \mathcal{A}, \text{ with probability } \varepsilon_{exp} \\ \arg\max_a Q_\theta(s_t^{D3QN}, a), \text{ with probability } 1 - \varepsilon_{exp} \end{cases} \tag{22}$$

where $\varepsilon_{exp}$ is the exploration rate, and $Q_\theta(s_t^{D3QN}, a)$ denote the Q-values computed using the dueling architecture as follows:

$$Q_\theta(s_t^{D3QN}, a) = V_{\theta,\beta}(s_t^{D3QN}) + (A_{\theta,\alpha}(s_t^{D3QN}, a) - \frac{1}{|\mathcal{A}|}\sum_{a'\in\mathcal{A}} A_{\theta,\alpha}(s_t^{D3QN}, a')) \tag{23}$$

where $A_{\theta,\alpha}(s_t^{D3QN}, a)$ and $V_{\theta,\beta}(s_t^{D3QN})$ represent the advantage and the state-value of action $a$, computed using the online dueling Q-network with two separate sub-networks with parameters $\alpha$ and $\beta$, respectively. Second, the discrete action $a_{d_t}$ and the environment state $s_t$ are used to obtain the state of the TD3 agent based on the chosen type of inter-agent communication $n$, as follows:

$$s_t^{TD3} = \mathcal{H}_n[s_t, a_{d_t}],\ n = 1, 2, 3, 4 \tag{24}$$

The continuous agent selects the continuous RAN slicing actions $a_{c_t}$ according to the actor network policy $\pi_\psi$ as follows:

$$a_{c_t} = \text{clip}\Big(\pi_\psi(s_t^{TD3}) + \mathcal{N}_{exp}, a_c^{Min}, a_c^{Max}\Big) \tag{25}$$

where $\mathcal{N}_{exp}$ denotes the additive Gaussian exploration noise with standard deviation $\sigma_{exp}$, and $a_c^{Min}$ and $a_c^{Max}$ define the range of the continuous action. Then, the combined action $a_t = (a_{d_t}, a_{c_t})$ is executed, and the ASH-MARL algorithm receives from the environment the next state $s_{t+1}$ and a reward $r_t$. These interactions with the environment $(s_t, a_{d_t}, a_{c_t}, r_t, s_{t+1})$ are stored in the replay buffer $\mathcal{D}$, which is shared between the two agents. To update the networks, a mini-batch of transitions is sampled from $\mathcal{D}$ using a PER mechanism. Specifically, the transitions are sampled from the buffer, based on a priority score $p_k^\varrho$, with the following probability:

$$\mathcal{P}(k) = \frac{p_k^\varrho}{\sum_i p_i^\varrho}, \quad \text{where} \quad p_k^\varrho = \delta_k + \varepsilon_{PER} \tag{26}$$

where $\varrho$ controls the degree of prioritization, $\varepsilon_{PER}$ ensures non-zero probabilities, and $\delta_k$ denotes the temporal-difference (TD) error of transition $k$, given by,

$$\delta_k = \max\left(|\delta_k^{D3QN}|, |\delta_k^{TD3}|\right) \tag{27}$$

where $\delta_k^{D3QN}$ and $\delta_k^{TD3}$ are the TD errors of the D3QN and TD3 agents. The TD error of the discrete agent is given by,

$$\delta_k^{D3QN} = y_k^{D3QN} - Q_\theta(s_k^{D3QN}, a_k) \tag{28}$$

where $Q_\theta(s_k^{D3QN}, a_k)$ are the current Q-values estimated by the online dueling Q-network and $y_k^{D3QN}$ are the corresponding target values given by,

$$y_k^{D3QN} = \begin{cases} r_k + \gamma\, Q_{\theta'}(s_{k+1}^{D3QN}, \tilde{a}_{d_k}) \\ r_k, \quad \text{if } s_{k+1} \text{ is terminal} \end{cases} \tag{29}$$

where $Q_{\theta'}(s_{k+1}^{D3QN}, \tilde{a}_{d_k})$ are the Q-values computed using the target dueling Q-network and $\gamma$ denotes the discount

**Algorithm 1** ASH-MARL Algorithm for Joint VNF Orchestration and RAN Slicing

**Input** ASH-MARL hyperparameters: discount factor $\gamma$, batch size $\mathcal{B}$, PER parameters $\varrho$, $\varepsilon_{PER}$, and $\nu$, type of inter-agent communication $n$, and number of episodes $M$
  D3QN hyperparameters: exploration rate $\varepsilon_{exp}$, learning rate $\epsilon_{D3QN}$, and Polyak factor $\tau_{D3QN}$
  TD3 hyperparameters: exploration noise $\mathcal{N}_{exp}$, target policy noise $\mathcal{N}_{targ}$, target policy clipping parameter $\mathcal{N}$, learning rates $\epsilon_{TD3}^{critic}$, and $\epsilon_{TD3}^{actor}$, policy delay frequency $F_{TD3}$, and Polyak factor $\tau_{TD3}$
**Initialize** the discrete D3QN agent:
  The online dueling Q-network $Q_\theta$ with weights $\theta$
  The target network $Q_{\theta'}$ with weights $\theta' \leftarrow \theta$
**Initialize** the continuous TD3 agent:
  The actor network $\pi_\psi$ with weights $\psi$
  The critic networks $Q_{\phi_1}$, $Q_{\phi_2}$ with weights $\phi_1$, $\phi_2$
  The target networks $\pi_{\psi'}$, $Q_{\phi'_1}$, and $Q_{\phi'_2}$ with weights $\pi_{\psi'} \leftarrow \pi_\psi$, $\phi'_1 \leftarrow \phi_1$, and $\phi'_2 \leftarrow \phi_2$
**for** ep = 1 to $M$ **do**
  Observe initial state $s_1$
  **for** t = 1 to $T$ **do**
    ***Discrete action selection using D3QN agent***
    Select action $a_{d_t}$ using (22)
    ***Inter-agent communication via*** $\mathcal{H}_n[s_t, a_{d_t}]$
    Compute the TD3 agent state $s_t^{TD3}$ using (21)
    ***Continuous action selection using TD3 agent***
    Select action $a_{c_t}$ using (25)
    ***Interaction with the environment***
    Execute the actions $(a_{d_t}, a_{c_t})$ in the environment.
    Observe reward $r_t$ and next state $s_{t+1}$
    Store transition $(s_t, a_{d_t}, a_{c_t}, r_t, s_{t+1})$ in $\mathcal{D}$
    ***Sampling based on PER mechanism***
    Sample a mini-batch $\mathcal{B}$ of transitions from $\mathcal{D}$ with probabilities $\mathcal{P}(k), k \in \mathcal{B}$
    Compute the IS weights $\rho_k$ using (35)
    ***Discrete D3QN agent update***
    **for** $k \in \mathcal{B}$ **do**
      Select the target action $\tilde{a}_{d_k}$ using (30)
      Compute the target value $y_k^{D3QN}$ using (29)
      Compute the TD error $\delta_k^{D3QN}$ using (28)
    **end for**
    Compute the loss $\mathcal{L}_{D3QN}(\theta)$ using (34)
    Update the online dueling network $Q_\theta$ by minimizing the loss function $\mathcal{L}_{D3QN}(\theta)$ using gradient descent:

$$\theta \leftarrow \theta - \epsilon_{D3QN} \ \nabla_\theta \mathcal{L}_{D3QN}(\theta)$$

    Update the target network $Q_{\theta'}$, using Polyak updates:

$$\theta' \leftarrow \tau_{D3QN}\theta + (1 - \tau_{D3QN})\theta$$

    ***Continuous TD3 agent update***
    **for** $k \in \mathcal{B}$ **do**
      Compute the TD3 agent state $s_k^{TD3}$ using (21)
      Select the target action $\tilde{a}_{c_k}$ using (33)
      Compute the target value $y_k^{TD3}$ using (32)
      Compute the TD error $\delta_k^{TD3}$ using (31)
    **end for**
    Compute the critic loss $\mathcal{L}_{TD3}(\phi_1, \phi_2)$ using (37)
    Update the critic networks $Q_{\phi_1}$ and $Q_{\phi_2}$ by minimizing the loss function $\mathcal{L}_{TD3}(\phi_1, \phi_2)$ using gradient descent:

$$\phi_j \leftarrow \phi_j - \epsilon_{TD3}^{critic} \ \nabla_{\phi_j} \mathcal{L}_{TD3}(\phi_1, \phi_2), \ j = 1, 2$$

    **if** $t \mod F_{TD3} = 0$ **then**
      Compute the actor objective $\mathcal{J}_{TD3}(\psi)$ using (38)
      Update the actor network $\pi_\psi$ by maximizing the policy objective $\mathcal{J}_{TD3}(\psi)$ using gradient ascent:

$$\psi \leftarrow \psi + \epsilon_{TD3}^{actor} \ \nabla_\psi \mathcal{J}_{TD3}(\psi)$$

      Update the target networks $Q_{\phi'_1}$, $Q_{\phi'_2}$, and $\pi'_\psi$, using Polyak updates:

$$\phi'_j \leftarrow \tau_{TD3}\phi_j + (1 - \tau_{TD3})\phi'_j, \ j = 1, 2$$

$$\pi_{\psi'} \leftarrow \tau_{TD3}\pi_\psi + (1 - \tau_{TD3})\pi_{\psi'}$$

    **end if**
    ***Update the PER priorities***
    **for** $k \in \mathcal{B}$ **do**
      Compute the TD error $\delta_k$ using (27)
      Compute the new priorities $p_k^\varrho$ using (26)
    **end for**
  **end for**
**end for**

factor. Also, $\tilde{a}_{d_k}$ is the target discrete action selected according to the greedy policy using the online Q-network, given by,

$$\tilde{a}_{d_k} = \arg\max_a Q_\theta(s_{k+1}^{D3QN}, a) \tag{30}$$

Moreover, the TD error of the continuous agent is,

$$\delta_k^{TD3} = y_k^{TD3} - \frac{1}{2}\left(Q_{\phi_1}(s_k^{TD3}, a_k) + Q_{\phi_1}(s_k^{TD3}, a_k)\right) \tag{31}$$

where $Q_{\phi_1}(s_k^{TD3}, a_k)$ and $Q_{\phi_1}(s_k^{TD3}, a_k)$ denote the current Q-values estimated by the first and second critic networks of the TD3 agent, and $y_k^{TD3}$ are the target values given by,

$$y_k^{TD3} = \begin{cases} r_k + \gamma \min\left(Q_{\phi'_1}(s_{k+1}^{TD3}, \tilde{a}_{c_k}), Q_{\phi'_2}(s_{k+1}^{TD3}, \tilde{a}_{c_k})\right) \\ r_k, \quad \text{if } s_{k+1} \text{ is terminal} \end{cases} \tag{32}$$

where $Q_{\phi'_1}(s_{k+1}^{TD3}, \tilde{a}_{c_k})$ and $Q_{\phi'_2}(s_{k+1}^{TD3}, \tilde{a}_{c_k})$ are the Q-values estimated by the two target critic networks. Additionally, $\tilde{a}_{c_k}$ is the target continuous action selected according to the target actor network policy and given by,

$$\tilde{a}_{c_k} = \text{clip}\big(\pi_{\psi'}(s_{k+1}) + \text{clip}\big(\mathcal{N}_{targ}, -\mathcal{N}, \mathcal{N}\big), a_c^{Min}, a_c^{Max}\big) \tag{33}$$

where $\mathcal{N}_{targ}$ is the target policy smoothing Gaussian noise with standard deviation $\sigma_{targ}$, and $\mathcal{N}$ is the target noise clipping parameter.

After sampling a mini-batch $\mathcal{B}$ of transitions, the agents update their networks by optimizing their loss and objective functions. On the one hand, the D3QN agent updates its online dueling Q-network by minimizing the following loss function:

$$\mathcal{L}_{D3QN}(\theta) = \sum_{k \in \mathcal{B}} \rho_k \left(y_k^{D3QN} - Q_\theta(s_k^{D3QN}, a_{d_k})\right)^2 \tag{34}$$

| Parameter | Value |
|---|---|
| Path loss exponents $\alpha_{AS}$, $\alpha_{SS}$ | 2.2, 2 |
| Carrier frequency $f_c$ | 2 GHz |
| Noise power spectral density $N_0$ | $-124$ dBm/Hz |
| UAVs antenna gain $G_u$ | 20 dB |
| Buoys and ships antenna gain $G_b$, $G_{\kappa_s}$ | 10 dB |
| Discount factor $\gamma$ | 0.99 |
| Batch size $\mathcal{B}$ | 200 |
| Replay Buffer size $\mathcal{D}$ | 400000 |
| PER parameters $\varrho$, $\varepsilon_{PER}$, $\nu$ | 0.6, $10^{-6}$, 0.4 |
| D3QN learning rate $\epsilon_{D3QN}$ | 0.0001 |
| D3QN Polyak factor $\tau_{D3QN}$ | 0.005 |
| TD3 exploration noise $\mathcal{N}_{exp}$ | 0.22 |
| TD3 target policy noise $\mathcal{N}_{targ}$ | 0.15 |
| TD3 target policy clipping parameter $\mathcal{N}$ | 0.2 |
| TD3 learning rates $\epsilon_{TD3}^{critic}$, $\epsilon_{TD3}^{actor}$ | 0.001, 0.00002 |
| TD3 policy delay frequency $F_{TD3}$ | 3 |
| TD3 Polyak factor $\tau_{TD3}$ | 0.002 |

TABLE 2: Simulation Parameters [30], [40], [41].

where $\rho_k$ are the importance sampling (IS) weights introduced by the PER sampling mechanism and expressed as,

$$\rho_k = \left(\frac{1}{\mathcal{B}\,\mathcal{P}(k)}\right)^{\nu} \tag{35}$$

where $\nu$ is the importance sampling exponent. Then, a soft update strategy is used to update the target network $Q_{\theta'}$ using the online network of the D3QN agent. In this work, instead of the hard updates ($\theta' \leftarrow \theta$), we adopt the Polyak updates, typically used by DDPG and TD3 algorithms, to adjust the target parameters:

$$\theta' \leftarrow \tau_{D3QN}\theta + (1-\tau_{D3QN})\theta \tag{36}$$

where $\tau_{D3QN}$ denotes the Polyak factor of D3QN agent. These soft updates offer smooth update of the target network weights, which enhances training stability. On the other hand, the TD3 agent updates its two critic networks by minimizing the loss function given by,

$$\begin{aligned}\mathcal{L}_{TD3}(\phi_1,\phi_2) = \sum_{k\in\mathcal{B}} \rho_k[(y_k^{TD3} - Q_{\phi_1}(s_k^{TD3}, a_{c_k}))^2 \\ +(y_k^{TD3} - Q_{\phi_2}(s_k^{TD3}, a_{c_k}))^2]\end{aligned} \tag{37}$$

Additionally, the actor network of the TD3 agent is updated by maximizing the deterministic policy gradient objective expressed as,

$$\mathcal{J}_{TD3}(\psi) = \sum_{k\in\mathcal{B}} Q_{\phi_1}(s_k, \pi_\psi(s_k)) \tag{38}$$

Using delayed Polyak updates, the actor network $\pi_\psi$ and the target networks $Q_{\phi_1'}$, $Q_{\phi_2'}$, and $\pi_\psi'$ of the TD3 agent are updated less frequently than the critic networks $Q_{\phi_1}$ and $Q_{\phi_2}$ with policy delay frequency $F_{TD3}$ and Polyak factor $\tau_{TD3}$.

## 5 Simulation Results

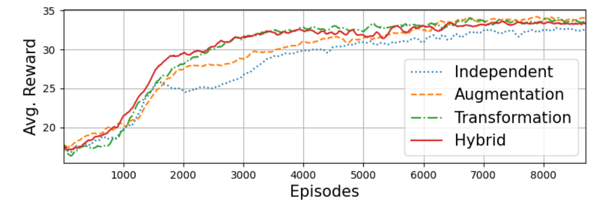
This section presents the numerical results that evaluate the proposed network slicing framework for maritime networks. To conduct our simulations, we assume that the considered maritime region spans over an area of 25 $Km^2$ and includes 6 marine buoys available for connectivity services. The UAVs are assumed to fly at the altitudes of $z_\xi = 115m$ for $\xi \in U$, and $z_\xi = 112m$ for $\xi \in U_{th}$. Meanwhile, the marine buoys and the ships are assumed to be at the same sea level ($z_{\kappa_s} = z_\xi = 2m$, $\xi \in B$). In addition, the maximum transmission power $P_\xi^{\text{transmit}}$, the maximum CPU capacity $C_\xi^{\text{total}}$, and the available bandwidth $B_\xi$ at node $\xi$ are 30 dBm, $10^9$ cycles/s, 2 MHz for the marine buoys [4]. The data sizes needed to serve the low-latency slice and the high-throughput slice are 2 Kbits and 2 Mbits. The CPU capacity $c_i$ needed for the deployment of one VNF instance $i$ is $10^8$ cycles/s and the capacity $Q_{P,i,s}$ required by the VNF instance $i$ to serve slice $s$ is $10^7$ cycles/s. Moreover, by conducting numerous experiments, we fine-tune the parameters of the designed ASH-MARL algorithm. On the one hand, the network architecture of the D3QN agent is composed of two layers with 256 neurons. On the other hand, both actor and critic networks of the TD3 include two layers of 512 neurons. These architectures provide sufficient representation capacity to capture the high-dimensional state and action spaces of the considered orchestration and slicing problem while maintaining reasonable computational complexity. Furthermore, we tune the different hyper-parameters including the discount factor, the batch size and the PER parameters. In addition, we employ an adaptive exploration rate $\varepsilon_{exp}$ for the D3QN agent where the initial rate and the decay steps are $0.9$ and $150000$, respectively. The simulation parameters for the integrated network environment and the ASH-MARL algorithm are summarized in Table 2.

### 5.1 Convergence Performance

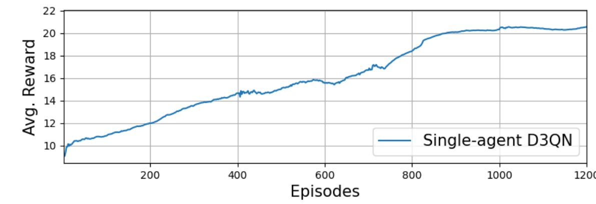
The convergence performance of the proposed ASH-MARL algorithm is illustrated in Fig.3 (a). We observe the average training reward for the different inter-agent communication methods, namely the independent, the augmentation, the transformation, and the hybrid approaches. The four methods show stable convergence achieving average rewards between 32 and 34 after about 8000 episodes. Despite slower initial learning, the augmentation approach reaches the highest reward. This observation indicates that providing TD3 with an augmented state representation improves the agents coordination and the ASH-MARL long-term policy quality. Meanwhile, the transformation method presents a faster convergence with similar maximum reward, benefiting from the compact transformation of the environment's state using the D3QN actions. Moreover, although the hybrid approach converges to a slightly lower reward, it exhibits the fastest early learning indicating that the combination of the augmentation and transformation techniques enables more efficient policy updates and cooperative learning. Lastly, the independent baseline converges to a comparable reward, but with significantly slower learning reflecting the limited coordination between the agents in the absence of explicit inter-agent interactions. Therefore, the convergence performance of these approaches indicates that enriching the TD3 agent state representation with the D3QN actions enhances the learning efficiency of the proposed ASH-MARL framework. These findings thereby highlights the key role of the inter-agent communication module in improving the agents' coordination.

In addition, we compare our solution with two single-agent benchmarks to evaluate the benefits of the hierarchical multi-agent algorithm. In particular, we consider a single-agent D3QN baseline with fully discrete actions and a single-agent TD3 baseline with fully continuous actions. Moreover, we include a metaheuristic benchmark based on the genetic algorithm (GA), which is widely adopted as a benchmark for evaluating network slicing and VNF orchestration solutions in the literature. We examine their average reward depicted in Fig.3 (b), Fig.3 (c) and Fig.3 (d). We observe that both single-agent benchmarks present a faster convergence speed compared with the multi-agent algorithm, requiring only 1200 episodes. This observation is expected since a single-agent method deals with a simpler learning problem and needs to learn only one policy with no coordination overhead. Additionally, the reward of the single-agent system is directly attributed to its own actions, giving it a straightforward credit assignment and thereby accelerating its learning process. In contrast, the ASH-MARL algorithm deals with a complex learning problem that requires the coordination of two policies. This coordination introduces an overhead that delays the convergence, especially with the hierarchical learning structure. The shared reward also contributes to the slow learning of the multi-agent system by complicating the credit assignments of the agents. Nonetheless, the ASH-MARL algorithm outperforms all considered benchmarks. In particular, the four ASH-MARL variants achieve average rewards between 32 and 34, compared with around 20 for both single-agent D3QN and GA and about 10 for the single-agent TD3. Thus, ASH-MARL achieves about 65% higher reward than the D3QN and GA benchmarks and over three times the reward of the TD3 baseline. In fact, the proposed multi-agent framework decomposes the complex joint RAN slicing and VNF orchestration problem into two sub-problems. Each agent then specializes in one part of the task, which reduces the complexity of the state-action space for each agent, enhances their exploration efficiency and leads to superior overall performance.

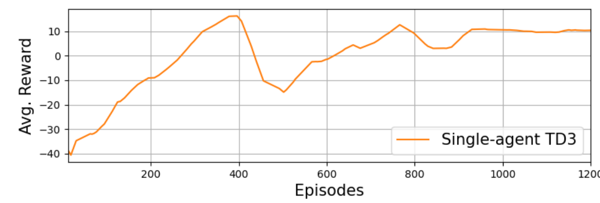
After understanding the convergence performance of the ASH-MARL framework, we examine its communication performance. We conduct extensive experiments and vary multiple settings of the integrated aerial-terrestrial maritime network. Specifically, we investigate three key performance indicators (KPIs) including the resource utilization, the throughput of slice 1 (the high-throughput slice) and the latency of slice 2 (low-latency slice). We note that we ensure equal total computing and communication resources for the different scenarios to provide a fair comparison and derive precise insights.



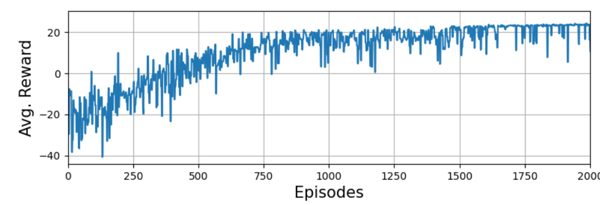

(a) ASH-MARL Algorithm




(b) Single-agent D3QN Benchmark




(c) Single-agent TD3 Benchmark




(d) Metaheuristic Benchmark

Fig. 3: Average reward over episodes for the ASH-MARL algorithm and the single-agent benchmarks.

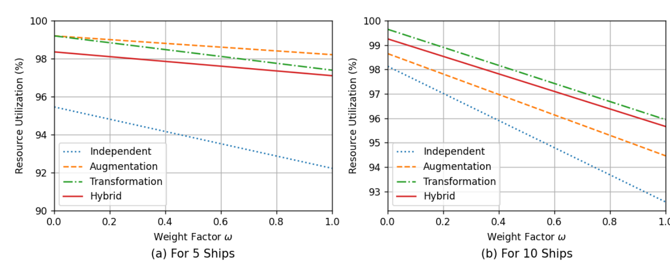



Fig. 4: Impact of Resource Weight Factor $\omega$ on Resource Utilization.

### 5.2 Impact of Resource Weight Factor

We begin by studying the effect of the weight factor $\omega$ that balances between the two resource utilization components defined in (13). Fig.4 depicts the variation of the overall resource utilization when $\omega$ increases from 0 to 1, for the four inter-agent communication approaches and considering the cases of 5 and 10 ships. We observe that the resource utilization is a decreasing function of $\omega$, which suggests that the transmit power tend to be more heavily utilized than the CPU resources. Additionally, the overall resource utilization remains above 92% across different scenarios. This observation indicates that the intelligent network slicing framework allows the virtualized network nodes to efficiently exploit both resources to satisfy the QoS requirements of the two slices. We note that we set $\omega$ to $0.5$ which balances between the two types of resources, for the rest of the results section to ensure accurate insights.

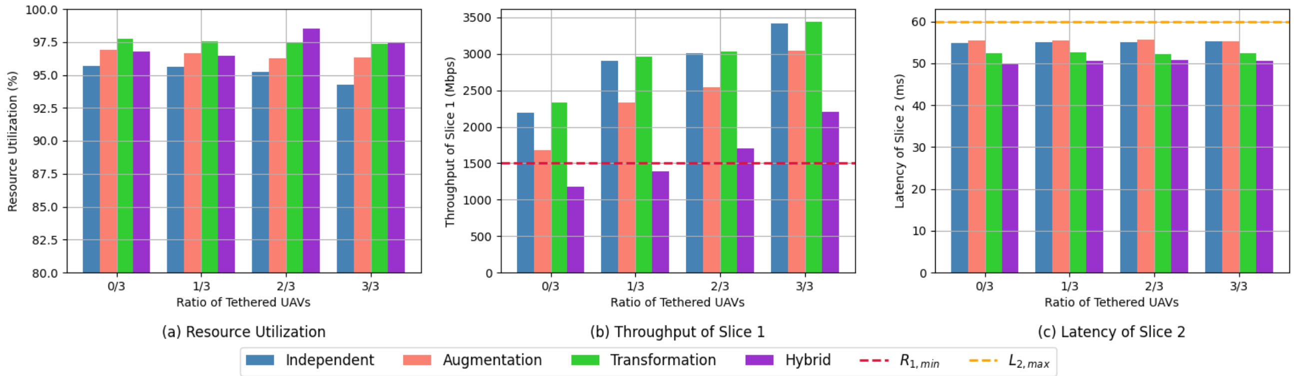



Fig. 5: Impact of Aerial Network Setup on Network Performance.

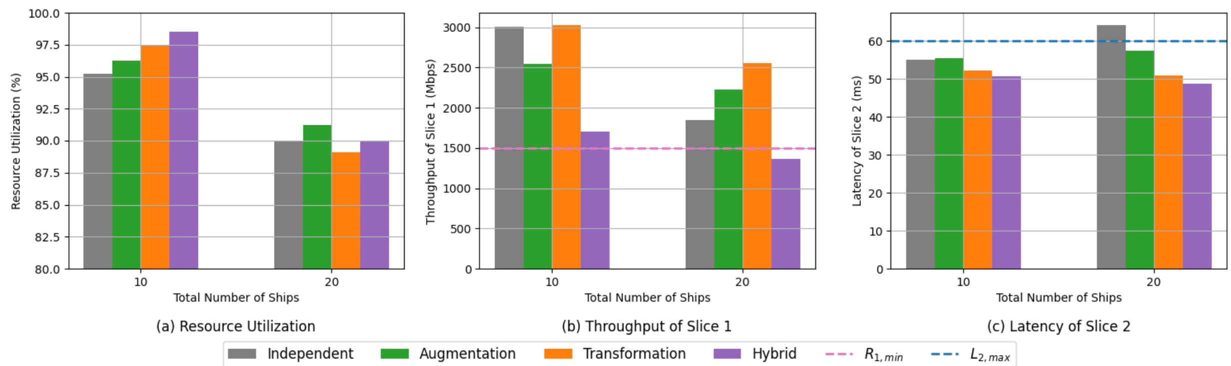



Fig. 6: Impact of Total Number of Ships on Network Performance.

### 5.3 Impact of Aerial Network Setup

Fig.5 illustrates the impact of the aerial segment architecture of the integrated network. We vary the ratio of tethered-UAVs while maintaining a constant total UAV number to establish a fair comparison. First, as shown in Fig.5 (a), we observe that the four inter-agent communication approaches maintain high resource utilization that is greater than 92% across different UAV deployments. The augmentation, transformation and hybrid techniques offer improved resource utilization compared to the independent method. Specifically, the transformation approach is consistently more resource-efficient, while the hybrid case requires the deployment of at least 2 tethered-UAVs. Moreover, we examine the impact of the tethered UAV ratio on the QoS of the two slices. On the one hand, the throughput of slice 1, depicted in Fig.5 (b), significantly improves when the number of the tethered UAVs increases. This observation is anticipated because of the reduced distance between the UAVs and the ships. While the independent, augmentation, and transformation methods offer a data rate greater than $R_{1,min}$ for all UAV settings, the hybrid case can only reach the minimum throughput requirement when at least two 2 tethered-UAVs are employed. On the other hand, the QoS requirement $L_{2,max}$ in terms of slice 2 latency is satisfied in all cases, as illustrated in Fig.5 (c). In particular, the hybrid approach provides the lowest delays compared to the other approaches, while the independent and augmentation methods have the highest latencies. Overall, the transformation technique demonstrates a superior performance across the three KPIs. This finding is due to the state representation, where the D3QN actions are directly incorporated into the TD3 state through the transformation function. As a result, the TD3 agent receives a state representation that already captures the impact of the orchestration decisions on the environment state while preserving the original state dimensionality. This compact representation facilitates the learning of the dependency between the orchestration and slicing decisions, leading to more effective slicing policies. The augmentation approach exhibits similar trends but achieves slightly lower KPI values, which is due to the concatenation of the environment state and the D3QN actions, where the TD3 agent observes them as separate inputs within its state representation. Consequently, the TD3 agent is required to infer the dependency between the orchestration and slicing decisions from a higher-dimensional state representation, increasing the complexity of the learning process. Meanwhile, the hybrid scheme provides high resource utilization with minimal latencies, however, it struggles to satisfy the data rate requirement. This finding can be explained by the TD3 state constructed through the combination of the transformed state with the D3QN actions. Although this provides additional information, it introduces a degree of redundancy and increases the TD3 state space dimensionality, which further complicates the learning task. Finally, although the independent method fulfills the QoS needs of the two slices, it is less resource-efficient than the other approaches due to the absence of explicit inter-agent communication, which limits their coordination capability.

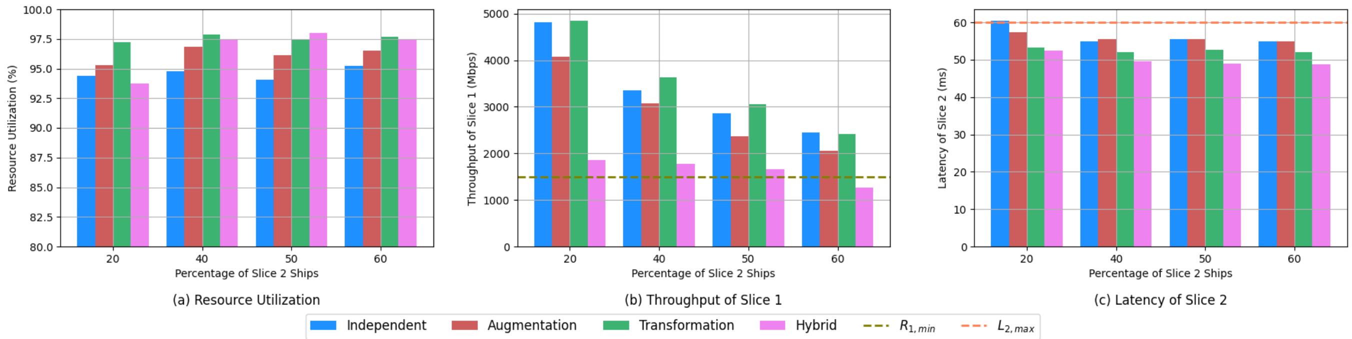



Fig. 7: Impact of Ship Distribution Across Slices on Network Performance.

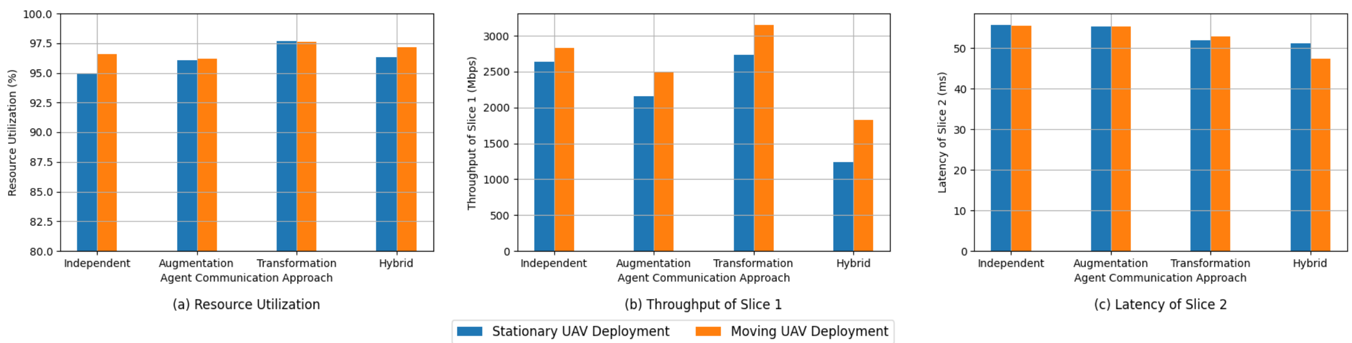



Fig. 8: Impact of Non-tethered UAV Deployment Strategy on Network Performance.

### 5.4 Impact of Network Scalability

We examine the impact of the network scalability on the network performance by varying the total number of ships, as illustrated in Fig.6. For all inter-agent communication approaches, the resource utilization decreases as the number of ships increases from 10 to 20 ships, with a reduction ranging between 5% and 8%, as shown in Fig.6 (a). This general performance deterioration is expected since the rise in number of ships results into more complex high-dimensional environment and larger state-action spaces. We also note a degradation in the throughput performance of slice 2, depicted in Fig.6 (b), which is expected as the integrated maritime network employs equal resources to support additional ships. While the data rates of the independent and hybrid approach are significantly affected by the increased number of ships, the transformation and augmentation techniques maintain high throughput values. Moreover, we observe a similar trend in the latency of slice 1, illustrated in Fig.6 (c), where it marginally rises with network scalability for the augmentation, transformation and hybrid techniques, whereas the increase is substantial for the independent case. This observation indicates that the inter-agent communication between TD3 and D3QN, through the asymmetric-state design, enhances the scalability capabilities of the ASH-MARL algorithm. While this analysis focuses on ship-level scalability by varying their number, the proposed framework can be extended to support maritime IoT slices with massive connectivity requirements [1], [42].

### 5.5 Impact of Ship Distribution Across Slices

We analyze the impact of ship distribution across slices on the KPIs, by varying the percentage of slice 2 ships within a total of 10 ships, as depicted in Fig.7. We note that the resource utilization remains higher than 92% across the different ship distributions, as illustrated in Fig.7 (a). The independent approach is consistently less resource-efficient compared to the other approaches, whereas the transformation method exhibits the highest performance. In addition, Fig.7 (b) shows that the throughput of slice 1 decreases as the percentage of slice 2 ships increases. This observation is anticipated since a larger number of slice 2 ships corresponds to fewer slice 1 ships. The different inter-agent communication approaches maintain a data rate above the $R_{1,min}$ requirement, with the transformation scheme offering the best throughput performance. Furthermore, the latency of slice 2 is demonstrated in Fig.7 (c) where a minimized delay is ensured across the ship distributions. While the hybrid approach achieves the shortest latencies, the independent method present the longest latencies surpassing the maximum requirement $L_{1,max}$ when 20% of the ships belong to slice 2. This finding further emphasizes the benefits of the information exchanged between the TD3 and D3QN agents through the inter-agent communication module.

### 5.6 Impact of Non-tethered UAV Deployment Strategy

We investigate the impact of non-tethered UAV deployment on the network performance, as illustrated in Fig.8. Specifically, we employ two UAV deployment strategies including a stationary UAV scheme where the UAV is hovering at a strategic position at the center of the maritime area, and a moving UAV strategy where the non-tethered UAV is traveling at a specific altitude along the same direction as the ships movement. These strategies are adopted to capture the effect of non-tethered UAV dynamics on the proposed framework, while focusing on the optimization of RAN slicing and VNF orchestration[2]. Fig.8 (a) shows the resource utilization for the four inter-agent communication methods. We observe that the UAV movement slightly enhances the resource utilization, indicating that the ASH-MARL algorithm effectively adapts to the additional environment dynamics introduced by the UAV mobility. Moreover, the effect of the UAV deployment strategy on the throughput performance is illustrated in Fig.8 (b). Moving the non-tethered UAVs along the ships direction leads to a noticeable improvement in slice 1 throughput. This observation is expected since the distance between the UAVs and the ships is reduced. Furthermore, the latency of slice 2 maintain similar values across the two deployment schemes for the independent, augmentation, and transformation techniques, as shown in Fig.8 (c). In contrast, the hybrid approach exhibits a reduction in delay when the moving UAVs strategy is employed, which indicates that the combined transformation and augmentation of the TD3 state enables the ASH-MARL algorithm to better exploit the proximity of nearby UAVs.

# 6 Conclusion

In this work, we designed an intelligent network slicing framework for VNF orchestration and RAN resource slicing to support two different slices in integrated aerial-terrestrial maritime networks. We developed ASH-MARL based on D3QN and TD3 to handle the discrete and continuous actions and deal with their interdependency. Additionally, we introduced an inter-agent communication module that describes the communication between the two agents in our multi-agent setting. This module allowed us to examine four different approaches, namely the augmentation, transformation, hybrid, and independent approaches, that describe the impact of the environment state and the D3QN actions on the TD3 agent. Our findings indicate that the transformation technique exhibits a superior overall performance across the three KPIs, including the resource utilization, the throughput of slice 1 and the latency of slice 2, owing to its direct and compact inter-agent communication. Furthermore, the comparison of ASH-MARL algorithm with two single-agent approaches shows that the former outperforms these latter benchmarks. As future work, we intend to extend our AI-native network slicing framework to accommodate three-layered network architectures consisting of terrestrial, aerial and satellite segments. This extension would provide seamless connectivity for different types of ships including both large cargo vessels and small boats, and support diversified maritime applications with tailored QoS requirements. In addition, we plan to investigate different slicing problem formulations that jointly optimize multiple performance objectives, including resource utilization, throughput, latency, fairness, and service robustness. Lastly, the integration of additional slices, including maritime IoT slices and mission-critical slices, represents another extension of this work, enabling the proposed framework to support massive connectivity and reliability requirements in addition to the currently considered high-throughput and low-latency slices.

2. We note that more advanced UAV mobility strategies, including UAV trajectory optimization, have been investigated in our previous work [22].

# References


[1] F. S. Alqurashi, A. Trichili, N. Saeed, B. S. Ooi, and M.-S. Alouini, "Maritime communications: A survey on enabling technologies, opportunities, and challenges," *IEEE Internet of Things Journal*, vol. 10, no. 4, pp. 3525–3547, 2023.

[2] S. Ammar, O. Amin, and B. Shihada, "Tethered UAV-based communications for under-connected near-shore maritime areas," in *IEEE BlackSeaCom*, 2024, pp. 42–47.

[3] J. Wang, H. Zhou, Y. Li, Q. Sun, Y. Wu, S. Jin, T. Q. Quek, and C. Xu, "Wireless channel models for maritime communications," *IEEE Access*, vol. 6, pp. 68 070–68 087, 2018.

[4] M. Dai, C. Dou, Y. Wu, L. Qian, R. Lu, and T. Q. Quek, "Multi-UAV aided multi-access edge computing in marine communication networks: A joint system-welfare and energy-efficient design," *IEEE Transactions on Communications*, 2024.

[5] J. Lu, S. Ammar, O. Amin, and B. Shihada, "Tethered UAV-assisted networks for ubiquitous near-shore maritime connectivity," in *IEEE MECOM*, 2025.

[6] C.-L. I and S. Katti, "O-RAN: Towards an Open and Smart RAN," *O-RAN Alliance White Paper*, no. WP-2018, October 2018.

[7] B. Agarwal, R. Irmer, D. Lister, and G.-M. Muntean, "Open ran for 6g networks: Architecture, use cases and open issues," *IEEE Communications Surveys & Tutorials*, 2025.

[8] M. Wani, M. Kretschmer, B. Schröder, A. Grebe, and M. Rademacher, "Open ran: A concise overview," *IEEE Open Journal of the Communications Society*, vol. 6, pp. 13–28, 2024.

[9] K. Alam, M. A. Habibi, M. Tammen, D. Krummacker, W. Saad, M. Di Renzo, T. Melodia, X. Costa-Pérez, M. Debbah, A. Dutta *et al.*, "A comprehensive tutorial and survey of o-ran: Exploring slicing-aware architecture, deployment options, use cases, and challenges," *IEEE Communications Surveys & Tutorials*, 2025.

[10] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Computer Networks*, vol. 167, p. 106984, 2020, doi=10.1016/j.comnet.2019.106984.

[11] W. Rafique, J. R. Barai, A. O. Fapojuwo, and D. Krishnamurthy, "A survey on beyond 5g network slicing for smart cities applications," *IEEE Communications Surveys & Tutorials*, vol. 27, no. 1, pp. 595–628, 2024.

[12] A. Celik, N. Saeed, B. Shihada, T. Y. Al-Naffouri, and M.-S. Alouini, "A software-defined opto-acoustic network architecture for internet of underwater things," *IEEE Communications Magazine*, vol. 58, no. 4, pp. 88–94, 2020.

[13] M. Gharbaoui, B. Martini, S. Noto, A. L. Ruscelli, P. Pagano, and P. Castoldi, "Experimenting SDN/NFV solutions for flexible maritime transport & logistics (T&L) services," in *IEEE NFV-SDN*, 2023, pp. 27–33.

[14] S. Ammar, C. Pong Lau, and B. Shihada, "An in-depth survey on virtualization technologies in 6G integrated terrestrial and non-terrestrial networks," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 3690–3734, 2024.

[15] Y. H. Xu, J. H. Li, W. Zhou, and C. Chen, "Learning-empowered resource allocation for air slicing in UAV-assisted cellular V2X communications," *IEEE Systems Journal*, vol. 17, pp. 1008–1011, 3 2023.

[16] P. Yang, X. Xi, T. Q. Quek, J. Chen, X. Cao, and D. Wu, "Repeatedly energy-efficient and fair service coverage: UAV slicing," *IEEE Global Communications Conference*, 12 2020.

[17] G. Wang, S. Zhou, S. Zhang, Z. Niu, and X. Shen, "SFC-based service provisioning for reconfigurable space-air-ground integrated networks," *IEEE Journal on Selected Areas in Communications*, vol. 38, pp. 1478–1489, 7 2020.

[18] M. Pourghasemian, M. R. Abedi, S. S. Hosseini, N. Mokari, M. R. Javan, and E. A. Jorswieck, "AI-based mobility-aware energy efficient resource allocation and trajectory design for NFV enabled aerial networks," *IEEE Transactions on Green Communications and Networking*, vol. 7, pp. 281–297, 3 2023.

[19] Q. He and J. Liang, "Online joint optimization of virtual network function deployment and trajectory planning for virtualized service provision in multiple-unmanned-aerial-vehicle mobile-edge networks," *Electronics*, vol. 13, 3 2024.

[20] M. Akbari, A. Syed, W. S. Kennedy, and M. Erol-Kantarci, "AoI-aware energy-efficient SFC in UAV-aided smart agriculture using asynchronous federated learning," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 1222–1242, 2024.

[21] Y. Peng and B. Di, "Joint VNF deployment and resource allocation in integrated terrestrial-aerial access networks enabled by network slicing," in *IEEE EUC*, 2022, pp. 74–80.

[22] S. Ammar, W. Abderrahim, and B. Shihada, "Maritime-oriented network slicing in o-ran integrated aerial-terrestrial networks," *IEEE Transactions of Mobile Computing*, 2025.

[23] Y. Cui, H. Shi, R. Wang, P. He, D. Wu, and X. Huang, "Multi-agent reinforcement learning for slicing resource allocation in vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 2, pp. 2005–2016, 2023.

[24] B. Lu, Y. Wu, L. Qian, S. Zhou, H. Zhang, and R. Lu, "Multi-agent drl-based two-timescale resource allocation for network slicing in v2x communications," *IEEE Transactions on Network and Service Management*, 2024.

[25] M. Asim Ejaz, G. Wu, and T. Iqbal, "Dynamic and efficient resource allocation for 5g end-to-end network slicing: A multi-agent deep reinforcement learning approach," *International Journal of Communication Systems*, vol. 37, no. 17, p. e5916, 2024.

[26] N. Ghafouri, J. S. Vardakas, K. Ramantas, and C. Verikoukis, "A multi-level deep rl-based network slicing and resource management for o-ran-based 6g cell-free networks," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 11, pp. 17 472–17 484, 2024.

[27] X. Fang, W. Feng, Y. Wang, Y. Chen, N. Ge, Z. Ding, and H. Zhu, "Noma-based hybrid satellite-uav-terrestrial networks for 6g maritime coverage," *IEEE Transactions on Wireless Communications*, vol. 22, no. 1, pp. 138–152, 2022.

[28] Y. Xu, "Quality of service provisions for maritime communications based on cellular networks," *IEEE Access*, vol. 5, pp. 23 881–23 890, 2017.

[29] T. Wei, W. Feng, Y. Chen, C.-X. Wang, N. Ge, and J. Lu, "Hybrid satellite-terrestrial communication networks for the maritime internet of things: Key technologies, opportunities, and challenges," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8910–8934, 2021.

[30] A. A. Khuwaja, Y. Chen, N. Zhao, M.-S. Alouini, and P. Dobbins, "A survey of channel modeling for UAV communications," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2804–2821, 2018.

[31] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction, 2nd edition*. MIT press Cambridge, 2018.

[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[33] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[34] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.

[35] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1995–2003.

[36] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[37] O. A. S. up. Twin delayed ddpg. [Online]. Available: https://spinningup.openai.com/en/latest/algorithms/td3.html

[38] Y. Liu, A. Halev, and X. Liu, "Policy learning with constraints in model-free reinforcement learning: A survey," in *IJCAI*, 2021.

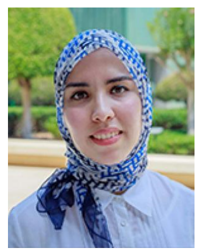
[39] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," *arXiv preprint arXiv:1511.05952*, 2015.

[40] D. W. Matolak and R. Sun, "Air–ground channel characterization for unmanned aircraft systems—part I: Methods, measurements, and models for over-water settings," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 26–44, 2016.

[41] J. Xu, M. A. Kishk, and M.-S. Alouini, "Space-air-ground-sea integrated networks: Modeling and coverage analysis," *IEEE Trans. Wireless Commun.*, vol. 22, no. 9, pp. 6298–6313, 2023.

[42] C. Dutriez, O. S. Oubbati, C. Gueguen, and A. Rachedi, "Adaptive and scalable cluster head management for mmtc using sampling-based agglomerative clustering and localized reclustering," in *GLOBECOM 2025-2025 IEEE Global Communications Conference*. IEEE, 2025, pp. 5223–5228.


**Sahar Ammar** received her Diplôme d'ingénieur from Ecole Polytechnique de Tunisie, Tunisia, in 2020 and her M.Sc. degree in electrical and computer engineering in 2022 from King Abdullah University of Science and Technology (KAUST), Saudi Arabia. She is currently pursuing her Ph.D. degree in electrical and computer engineering with the Networking Lab at KAUST. Her research interests include next-generation wireless networks, network virtualization technologies, and optical wireless communications.



**Wiem Abderrahim** (S'14 - M'18) received her Doctoral Degree in Information and Communication Technologies from the Higher School of Communications of Tunis (Sup'Com), Carthage University, Tunisia in 2017. In 2019, she joined King Abdullah University of Science and Technology (KAUST),Thuwal, Saudi Arabia as post-doctoral fellow within the Computer, Electrical and MathematicalSciences and Engineering (CEMSE) Division. Since 2023, she holds the position of assistant professor at Ecole Nationale d'Ingénieurs de Gabès (ENIG) and she is a research fellow at the MEDIATRON Lab within Sup'Com, Carthage University, Tunisia.

**Basem Shihada** (M'04-SM'12, IEEE) obtained his Ph.D. in Computer Science from the University of Waterloo, Canada, in 2007. Shortly after completing his studies, Prof. Shihada joined King Abdullah University of Science and Technology as a Founding Faculty member in 2008. His expertise lies in developing cutting-edge wireless systems, where he has made groundbreaking contributions across various domains, including intelligent wireless systems, wireless underwater systems, molecular communication systems, and non-terrestrial systems. Prof. Shihada's notable achievements are the creation and successful demonstration of Aqua-Fi, the world's first underwater Wi-Fi. Sun-Fi, the world first communication via building glass, and communication via breath. Prof. Shihada's work has been recognized with several best paper awards at renowned conferences within his field. His invaluable contributions have also been published in prestigious scientific journals such as Nature Electronics and many IEEE Transactions. In 2023, he become an area editor and received the exemplary editor award from the IEEE Communications Letter journal.