Deep Meta Advisor-aided Exploration for UAV Trajectory Design in Vehicular Networks

Deep Meta Advisor-aided Exploration for UAV Trajectory Design in Vehicular Networks

Leonardo Spampinato*†, Enrico Testi*†, Chiara Buratti*†, Riccardo Marini*

*WiLab, CNIT, Bologna, Italy
†DEI Department, University of Bologna, Bologna, Italy
Emails: {leonardo.spampinato, enrico.testi, c.buratti}@unibo.it, riccardo.marini@wilab.cnit.it

***Abstract*—Unmanned Aerial Vehicles (UAVs) deployed as aerial base stations (UABS) offer an adaptable solution for enhancing network performance, especially in vehicular networks. A key challenge is optimizing UABS trajectories in these dynamic environments. While Deep Reinforcement Learning (DRL) algorithms show the potential to solve this issue, their outcome wildly depends on the exploration phase carried out at the beginning of training. To address this, we introduce a deep meta advisor that, by applying efficient adaptation across similar tasks, learns an optimal exploration policy using augmented state inputs as additional context. Numerical results show that our approach improves agents learning efficiency across multiple tasks by enhancing the exploration phase, allowing to reach target performance with fewer training episodes compared to existing methods.**

***Index Terms*—UAV, V2X, Meta Learning, Exploration Policy.**

I. Introduction and State of the Art

Unmanned Aerial Vehicles (UAVs) deployed as aerial Base Stations (BSs), namely Unmanned Aerial BSs (UABSs), offer a flexible solution for improving terrestrial network performance through dynamic, on-demand service [1]–[3]. Due to their flexibility, as well as the possibility of offering enhanced Line-of-Sight (LoS) links, UABSs are well-suited to support data-intensive Vehicle-To-Everything (V2X) applications, such as advanced driving assistance and extended sensing [4], by collecting data from vehicles in motions. A critical challenge lies in designing algorithms that efficiently optimize the trajectory of UABS to maximize service toward users, which is usually solved by employing Deep Reinforcement Learning (DRL) [3]. In this context, an agent autonomously gathers information about the problem at hand and learns how to solve it. However, the efficiency of the trained model is strictly related to the environment with which the agent interacts, and performance quickly deteriorates when acting in an environment with even slightly different dynamics. Meta learning algorithms were introduced for agents to "learn how to learn", improving their capability to adapt to new tasks quickly [5]. These methodologies can affect the agent in different ways, for example, they can optimize model initialization [6] or enhance exploration [7]. Regarding the problem of UABS trajectory optimization, only a few works have investigated meta learning methodologies to improve agent adaptability to multiple tasks. In [8], authors use meta-Reinforcement Learning (RL) algorithms to tune the learning hyperparameters to enhance trajectory planning for collecting data from fixed on-ground users. However, simplified trajectory planning is considered where maps are subdivided into clusters. This cannot be applied to vehicular networks due to the need for finely-grained control of UABS trajectory. In [9] the use of implicit reward is exploited to enhance exploration of UAV for target rescue and obstacle avoidance. The methodology is not applicable in real-time online training and introduces a computation overhead that makes complex simulations unfeasible. Initial model optimization is proposed in [10], [11], empowering UAV to learn an initial trajectory model that can be quickly improved to locate and serve users, even though a huge number of data is needed for training. In this work, we introduce a new *deep meta advisor* [7] which learns an optimal exploration policy to enhance individual UABSs training. Differently from the aforementioned work, however, we further improve performances by leveraging task-specific *context* during exploration by providing the advisor with augmented input states used exclusively for exploration. The paper structure is as follows: Section II introduces the system model. Section III details the learning algorithm and the proposed meta-advisor. Section IV presents the numerical results. Conclusions are drawn in Section V.

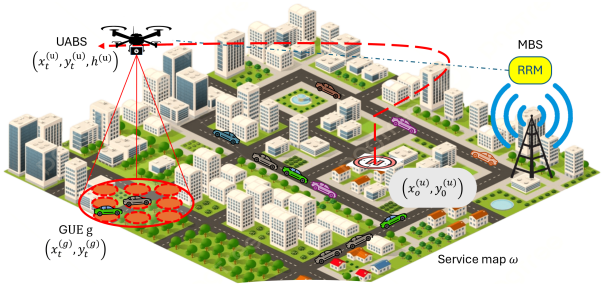

Fig. 1. Scenario representation assuming B=9 beams in 3x3 grid.

II. System Model

A. Scenario and Application

We address an urban scenario where Ground User Equipments (GUEs) $g \in \mathcal{G}$ engage in a V2X application. To enhance the network performance in these areas, a UABS is deployed, flying at speed $v^{(\mathrm{u})}$ at fixed altitude $h^{(\mathrm{u})}$. The UABS maintains

a backhaul link and collaborates with the existing terrestrial infrastructure, which includes a Macro Base Station (MBS) located at a fixed position. We consider a flight mission of duration T, divided into a sequence of discrete timesteps $t = 0, 1, \ldots, T-1$. At each timestep, the current positions of the UABS and GUEs are represented as $(x_t^{(\mathrm{u})}, y_t^{(\mathrm{u})})$, and $(x_t^{(g)}, y_t^{(g)})$, $\forall g \in \mathcal{G}$, respectively. During each timestep, GUEs attempt to transmit Cooperative Awareness Messages (CAMs) with a fixed payload size D. These transmissions occur within service windows of fixed duration T_s. To monitor successful packet transmission in each service window, a priority term $p_t^{(g)}$ is assigned to each GUE and is reset at the beginning of a new window. Users successfully upload packets, i.e., they are *served*, based on the output of a Radio Resource Management (RRM) algorithm, proposed in [12], accounting for all the traffic generated at timestep t for both the MBS, and the UABS. The algorithm maximizes the number of served users, weighted by priorities $p_t^{(g)}$, with the objective of improving service continuity in line with the application requirements. The network state of a user g at timestep t is represented by the tuple $(\psi_t^{(g,\mathrm{m})}, \psi_t^{(g,\mathrm{u})}, p_t^{(g)})$, where $\psi_t^{(g,\mathrm{m})} \in \{0,1\}$ is 1 if g has been served in t by the MBS, $\psi_t^{(g,\mathrm{u})} \in \{0,1\}$ is 1 is it has been served by the UABS, and $p_t^{(g)}$ its current priority. It holds that $\psi_t^{(g,\mathrm{m})} + \psi_t^{(g,\mathrm{u})} = \psi_t^{(g)} \leq 1$. Priority term $p_t^{(g)}$ evolves as $p_{t+1}^{(g)} = p_t^{(g)} + 1$ i.i.f $\psi_t^{(g)} = 1$. Working at mmWave frequency, operating at carrier frequency f_c, the UABS uses beamforming to create B circular beams arranged in a grid layout on the ground. A simplified representation of the scenario considered is reported in Fig. 1.

B. Markov Decision Process Model

Typically, to address a problem using DRL algorithms, a Markov Decision Process (MDP) is formulated to provide a mathematical framework for episodic decision-making. Furthermore, problems exhibiting structural similarities can be labeled as tasks τ_i, all belonging to a task set $\mathcal{T}$.

Let us denote by $\mathcal{S}$ the *state space*, representing all possible states s_t that the UABS can observe at time step t. The state at time t, $s_t = (x_t^{(\mathrm{u})}, y_t^{(\mathrm{u})}, t, \mathbf{b}_t^{(\mathrm{u})})$, consists of the current 2D position of the UABS, the time of the flight, and the *per-beam information* vector. The j-th entry of $\mathbf{b}_t^{(\mathrm{u})}$, $b_{t,j}^{(\mathrm{u})}$, is the cumulative priority $p_t^{(g)}$ of GUEs covered by the j-th UABS beam on the ground. s_{0,τ_i} denotes the *initial state*, with a take-off position $(x_0^{(\mathrm{u})}, y_0^{(\mathrm{u})})$. We denote by $\mathcal{A}$ the *action space*, defining the potential movements directions of the UABS, $\{0, \leftarrow, \uparrow, \rightarrow, \downarrow, \nwarrow, \nearrow, \searrow, \swarrow\}$, with 0 indicating hovering. Thus, the UABS trajectory is defined by a sequence of actions $[a_0, \ldots, a_{T-1}]$. $P_{\tau_i}(s,a)$ is the *transition function*, that describes the task-specific dynamics, mapping each state-action pair into the probability of transitioning to a subsequent state s'. $R_{\tau_i}(s,a,s')$ is the *reward function*, assigning a scalar value based on how favorable an action a is in state s, given the resulting transition to s', within task τ_i. Each task $\tau_i \in \mathcal{T}$ is then characterized by the MDP tuple $(\mathcal{S}, \mathcal{A}, P_{\tau_i}, R_{\tau_i}, s_{0,\tau_i})$. In the following, each task τ_i is defined by a specified take-off position, denoted as $(x_0^{(\mathrm{u})}, y_0^{(\mathrm{u})})_{\tau_i}$, and an associated service map ω_{τ_i}. Each service map ω_{τ_i} is characterized by a unique street layout and road traffic distribution. The reward is designed to optimize resource scheduling by favoring trajectories that lead the UABS satisfying more users:

$$r_t = \sum_{g \in \mathcal{G}} \psi_t^{(g,\mathrm{u})} p_t^{(g)} \tag{1}$$

representing the weighted sum of GUEs currently served by the UABS. Such a reward function reward encourages the UABS to maximize the number of users satisfied and their priority throughout its flight.

III. Deep Meta Advisor

This work aims to develop an approach that optimizes agents' exploration behavior, accelerating the DRL algorithm's convergence toward effective policies for each task $\tau_i \in \mathcal{T}$. To achieve this, we introduce a *deep meta-advisor*, a superagent that observes interactions across tasks in parallel. This broader view allows it to learn a shared exploration strategy that all agents can adopt early in training.

A. 3DQN Algorithm

For solving task τ_i, an agent is trained using the Double Dueling Deep Q Network (3DQN) algorithm, which aims to estimate Q-values of the policy π_i, $Q_{\pi_i}(s,a) = \mathbb{E}_{\pi_i}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \mid s_0 = s, a_0 = a\right]$. They represent the expected cumulative reward an agent can achieve by selecting action a in state s and subsequently following policy π_i. The policy π_i dictates which actions to take in each state based on the dynamics of task τ_i. To do so, 3DQN employs two neural networks: the *online* one, with parameters θ_i and the *target* one, with parameters θ_i^-. A dueling network architecture is used enabling agents to more effectively distinguish between the significance of states and impacts of actions, improving learning efficiency [13]. To update the network parameters, each agent collects experiences tuple $\{s_e, a_e, r_e, s'_e\}$ and stores them in its replay buffer $\mathcal{K}_i$ of size $|\mathcal{K}_i|$. Finally, at each training iteration, a batch of k experiences will be drawn from the replay buffer $\mathcal{K}_i$, and the *online* network parameters θ_i will be updated following the Double Deep Q-Learning (DQN) loss function, [14]:

$$\begin{aligned} L(\theta_i) &= \frac{1}{|k|} \sum_{e=1}^{|k|} \left(y_e - Q_{\pi_i}(s_e, a_e \mid \theta_i)\right)^2, \\ \text{where} \quad y_e &= r_e + \gamma Q_{\pi_i}(s'_e, \arg\max_{a'} Q_{\pi_i}(s'_e, a' \mid \theta_i^-)) \end{aligned} \tag{2}$$

with γ the discount factor. Every Y online training step, the *online* parameters are copied to the *target* network, $\theta^- \leftarrow \theta$. 3DQN algorithm demonstrated good learning performance for trajectory design problems, as studied in [3].

B. Exploration Policy

In RL algorithms, each agent must balance the exploration-exploitation tradeoff to train effectively. Early in training,

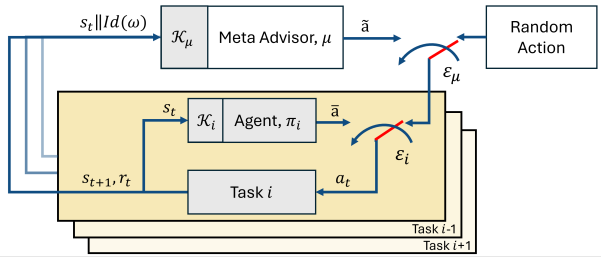

Fig. 2. Graphical representation of the exploration mechanism enabled by the deep meta advisor.

exploration is necessary to understand the environment dynamics, gathering experiences $\{s_e, a_e, r_e, s'_e\}$ that will guide policy development. However, inefficient or overly prolonged exploration can detract from the agent's ability to exploit learned insights later in training, reducing overall performance. Efficient exploration helps the agent quickly gain knowledge, leading to optimal trajectories and maximizing the expected cumulative reward by refining action choices. In this paper, three exploration policies are compared:

ϵ-greedy: this strategy balances exploration and exploitation by controlling the probability of taking a random action. In particular, the agent selects a random action at a given timestep with probability ϵ, whereas with probability $(1-\epsilon_i)$, it selects the action $\hat{a}$ with the highest estimated Q-value, which is an action that exploits the current knowledge of the environment. Throughout a training of N episodes, ϵ is linearly decayed following:

$$\epsilon = \max\left(\epsilon_{\min}, 1 - \frac{(1-\epsilon_{\min})\cdot n}{\epsilon_{\text{frac}}\cdot N}\right) \tag{3}$$

with ϵ_{frac} representing the fraction of episodes over which the decay occurs, $\epsilon_{\min}$ the minimum exploration probability agent will maintain after decaying, and n the current episode.

Deep Meta Advisor: the random action selection of the ϵ-greedy approach can negatively affect navigation by causing unnecessary direction changes, which limit exploration. To counter this, we introduce a *deep meta advisor* that learns via DRL an exploration policy μ for a set of tasks $\mathcal{T}$. Instead of randomly choosing actions during exploration, the advisor estimates the best exploration action $\tilde{a}$ based on its input state $\tilde{s_t}$, equal to agent state s_t. The *meta advisor* trains in parallel with individual agents, storing their experiences in a shared replay buffer. While each agent focuses on learning an optimal exploitation policy π_{τ_i} for the task τ_i it has been assigned to, the advisor learns an exploration policy applicable to all tasks. Without loss of generality, the meta advisor uses the 3DQN algorithm to learn policy parameters θ_μ and θ_μ^-, and an ϵ_μ-greedy policy with $\epsilon_{\text{frac}} << 1$ for its training. The deep meta advisor algorithm is further detailed in Algorithm 1.

Context-aided Deep Meta Advisor: this enhanced meta advisor leverages additional task-related context to learn the exploration policy. Its input state is augmented as $\tilde{s}_t = s_t \parallel \text{Id}(\omega_{\tau_i})$, being Id() a function mapping the service area of a task, ω_{τ_i}, to a one-hot encoded vector. This way, the context-aided deep meta advisor can learn a policy tailored to each encountered map. The proposed scheme is detailed in Algorithm 1 and illustrated in Fig. 2.

Algorithm 1 Agent and Advisor online training

Initialize advisor μ, initialize agents $\pi_i, i = 1, \ldots, |\mathcal{T}|$
for $n = 0, \ldots, N-1$ **do**
  for $\tau_i \in \mathcal{T}$ **do**
    $s_0 \leftarrow s_{0,\tau_i}$
    update ϵ_i according to (3) with $\epsilon_{\text{frac}} = \epsilon_{i,\text{frac}}$
    update ϵ_μ according to (3) with $\epsilon_{\text{frac}} = \epsilon_{\mu,\text{frac}}$
    for $t = 0, \ldots, T-1$ **do**

$$\tilde{s}_t \leftarrow \begin{cases} s_t \parallel Id(\omega_{\tau_i}), & \text{if Co-Adv} \\ s_t, & \text{if Adv} \end{cases}$$

$$a_t \leftarrow \begin{cases} \text{random action}, & \text{with prb. } \epsilon_i \cdot \epsilon_\mu \\ \tilde{a} = \arg\max_a Q_\mu(\tilde{s}_t, a), & \text{with prb. } (1-\epsilon_\mu)\cdot\epsilon_i \\ \bar{a} = \arg\max_a Q_{\pi_i}(s_t, a), & \text{otherwise} \end{cases}$$

      $r_t \sim R_i(s_t, a_t)$
      $s_{t+1} \sim P_i(s_t, a_t)$

$$\tilde{s}_t \leftarrow \begin{cases} s_t \parallel Id(\omega_{\tau_i}), & \text{if Co-Adv} \\ s_t, & \text{if Adv} \end{cases}$$

      $\mathcal{K}_i \leftarrow \{s_t, a_t, r_t, s_{t+1}\}$
      $\mathcal{K}_\mu \leftarrow \{\tilde{s}_t, a_t, r_t, \tilde{s}_{t+1}\}$
      Train θ_i with batch $k \sim \mathcal{K}_i$ according to (2)
  for $j = 0, \ldots, J-1$ **do**
    Train θ_μ with batch $k \sim \mathcal{K}_\mu$ according to (2)

TABLE I
SIMULATION PARAMETERS

$v^{(\text{u})}$	20 m/s	$h^{(\text{u})}$	100 m	ϕ	100°
T	270 s	T_s	10 s	$\lvert\mathcal{G}\rvert$	[90, 180]
f_c	30 GHz	B	9	P_{tx}	14 dBm
G_{tx}	0 dB	G_{rx}	23 dB	P_{n}	-106 dBm
$\epsilon_{\mu,\text{frac}}$	0.2	$\epsilon_{i,\text{frac}}$	0.6	$\epsilon_{\min}$	0.05
$\lvert\mathcal{K}_i\rvert$	5e+4	$\lvert\mathcal{K}_\mu\rvert$	1e+6	k	128
Y	100	γ	0.99	J	2700
D	1 MBit	L	1500 m	W	700 m

IV. Numerical Results

A. Simulation Settings and Performance Metrics

In this section, we report the learning performance of agents trained over N episodes. In the following, the context-aided deep meta advisor is referred to as "Co-Adv", while the simple deep meta advisor is labeled as "Adv". The task set $\mathcal{T}$ is the permutation of all combinations of maps $\{\omega_1, \omega_2\}$, both with common dimensions $L \times W$, and initial positions $\{(0,0), (L,0), (0,W), (L,W), (\frac{L}{2}, \frac{W}{2})\}$. These maps are modeled after real districts in Bologna, Italy.

Realistic movement behavior of GUEs is generated via Simulation of Urban MObility (SUMO) [15], according to street layout and traffic distributions that are map-specific. The Signal-to-Noise Ratio (SNR) in dB, used to estimate the Shannon channel capacity for the RRM algorithm, is computed as $\text{SNR} = P_{\text{tx}} + G_{\text{tx}} + G_{\text{rx}} - PL - P_{\text{n}}$, where P_{tx}, G_{tx}, G_{rx}, and P_{n} represent transmission power, transmitter gain, receiver gain and noise power, respectively. The 3GPP Urban Macro (UMa) channel model, detailed in [16], is adopted for calculating path losses accounting for both LoS and No-LoS

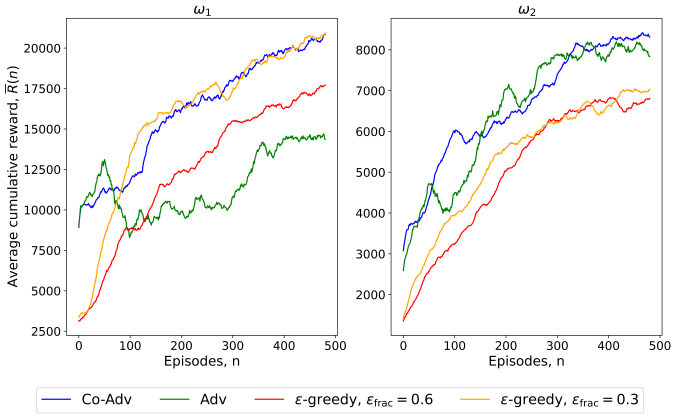

Fig. 3. Average cumulative reward of the proposed methods.

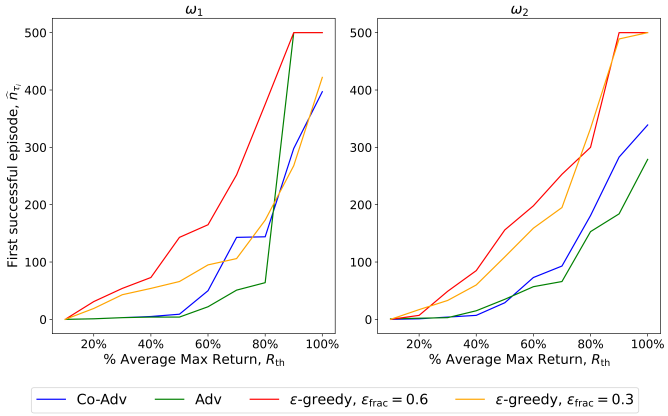

Fig. 4. First successful episode of the proposed methods.

(NLoS) links. The receiving gain of the UABS, G_{rx}, is derived in function of the vertical field of view ϕ and the number of beams B as described in [17]. All the simulation parameters are reported in Table I, where values inside square brackets are map-specific and follow the aforementioned order.

Results are presented in terms of average cumulative reward over a task set:

$$\overline{R}(n) = \frac{1}{|\mathcal{T}|} \sum_{\tau_i \in \mathcal{T}} R_{\tau_i,n} \tag{4}$$

where $R_{\tau_i,n} = \left(\sum_{t=0}^{T} r_t\right)$ is the sum of rewards obtained by an agent solving task τ_i at the n-th training episode. Curves are derived using a moving average with a window length of 50. To better highlight the effectiveness of the proposed approach, we introduce the concept of *first successful episode*. It corresponds to the first episode, within the sequence of N training episodes for task τ_i, whose return $R_{\tau_i,n}$ exceeds a given threshold R_{th}, equivalently:

$$\widehat{n}_{\tau_i} = \arg\min_{n} \{n \mid R_{\tau_i,n} \geq R_{\text{th}}\} \tag{5}$$

Moreover, even if the training is carried out considering the whole set $\mathcal{T}$, results are presented for the two service maps separately.

B. Deep Meta Advisor Performance

In Fig. 3 the average cumulative reward is shown. Notably, our proposed approach is comparable to agents trained with ϵ-greedy and ϵ_{frac}=0.3 when considering tasks in ω_1. However, all individual agents struggle to learn an effective trajectory on ω_2 due to the inefficiency of the ϵ-greedy exploration, which can require careful optimization based on the map considered. On the contrary, the proposed approach Co-Adv, given additional context, simultaneously enhances exploration, thus training, across both maps, showing higher robustness to challenging exploration with fixed hyperparameters. In Fig. 4 the first successful episode, with varying threshold R_{th} based on the best-performing episode, is reported for both service maps. As expected, the number of training episodes needed to reach target performance increases with a higher threshold. However, the trends of the Co-Adv are consistent between the two sets of tasks, whereas the other benchmarks have higher variability. In particular, agents with longer but inefficient exploration (ϵ_{frac}=0.6) cannot reach the highest performances. These results let us conclude that the proposed exploration policy is the most efficient and effective, achieving high returns while also requiring fewer episodes. This means that agents that explore using actions decided by the advisor have improved exploration and subsequent exploitation of their model. It both outperforms a standard random approach like ϵ-greedy, where experiences gathered by agents solving other tasks are not leveraged, and the simpler advisor proposed in [7], which cannot generalize an efficient exploration policy when the number of tasks available is low.

V. Conclusions

In this paper, a meta learning framework for enhancing the learning efficiency and adaptability of UABSs in vehicular networks was introduced. By training a deep meta advisor, it is possible to improve the exploration policy of multiple agents during their own training phase, effectively guiding them into discovering improved trajectories. Moreover, we proposed the use of augmented input states to provide contextual information across different tasks, enhancing robustness. Numerical results, studied both in terms of training outcome and time to reach target performance, demonstrate the superiority of the proposed framework over classical approaches across a wide set of challenging and peculiar scenarios without further hyperparameters tuning.

VI. Acknowledgment

This work has been carried out in the framework of the CNIT WiLab-Huawei Joint Innovation Center and also supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on “Telecommunications of the Future” (PE00000001 - program “RESTART”, Structural Project 6GWINET). We thank Aman Jassal, Chan Zhou and Malte Schellmann for the very fruitful discussion on this paper.

REFERENCES

[1] 3GPP, "Unmanned Aerial System (UAS) support in 3GPP," *TS 22.125 V17.1.0*, Dec. 2019.

[2] A. Fotouhi, H. Qiang, M. Ding, M. Hassan, L. G. Giordano, A. Garcia-Rodriguez, and J. Yuan, "Survey on UAV Cellular Communications: Practical Aspects, Standardization Advancements, Regulation, and Security Challenges," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3417–3442, 2019.

[3] L. Spampinato, D. Ferretti, C. Buratti, and R. Marini, "Joint Trajectory Design and Radio Resource Management for UAV-aided Vehicular Networks," *IEEE Transactions on Vehicular Technology*, pp. 1–14, 2024.

[4] ETSI, "5G; service requirements for enhanced V2X scenarios," *ETSI TS 22.186 version 16.2.0*, Nov. 2020.

[5] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-Learning in Neural Networks: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5149–5169, 2022.

[6] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *Proc. ICML*, vol. 70, Aug. 2017, pp. 1126–1135.

[7] F. M. Garcia and P. S. Thomas, "A Meta-MDP Approach to Exploration for Lifelong Reinforcement Learning," in *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS '19. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2019, p. 1976–1978.

[8] Y. Hu, M. Chen, W. Saad, H. V. Poor, and S. Cui, "Meta-Reinforcement Learning for Trajectory Design in Wireless UAV Networks," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[9] Z. Wang, W. Gao, G. Li, Z. Wang, and M. Gong, "Path Planning for Unmanned Aerial Vehicle via Off-Policy Reinforcement Learning With Enhanced Exploration," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 8, no. 3, pp. 2625–2639, 2024.

[10] Z. Lu, X. Wang, and M. C. Gursoy, "Trajectory Design for Unmanned Aerial Vehicles via Meta-Reinforcement Learning," in *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2023, pp. 1–6.

[11] R. Marini, S. Park, O. Simeone, and C. Buratti, "Continual Meta-Reinforcement Learning for UAV-Aided Vehicular Wireless Networks," in *ICC 2023*, 2022.

[12] D. Ferretti, S. Mignardi, R. Marini, R. Verdone, and C. Buratti, "QoE and Cost-Aware Resource and Interference Management in Aerial-Terrestrial Networks for Vehicular Applications," *IEEE Transactions on Vehicular Technology*, vol. 73, no. 8, pp. 11 249–11 261, 2024.

[13] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling Network Architectures for Deep Reinforcement Learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, p. 1995–2003.

[14] H. v. Hasselt, A. Guez, and D. Silver, "Deep Reinforcement Learning with Double Q-Learning," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, ser. AAAI'16. AAAI Press, 2016, p. 2094–2100.

[15] P. A. Lopez *et al.*, "Microscopic Traffic Simulation using SUMO," in *Proc. ITSC*, Maui, USA, Nov. 2018, pp. 2575–2582.

[16] 3GPP, "Technical Specification Group Radio Access Network; Study on channel model for frequencies from 0.5 to 100 GHz," *TR 38 901 version 16.1.0*, Dec. 2019.

[17] V. Salvia, *Antenna and Wave Propagation*. Laxmi Publications, 2007.