IMTVSim: An Integrated Modular Training and Verification Simulator for Unmanned Underwater Vehicles

Jingzehua Xu[*,+], Guanwen Xie[*,+], Zekai Zhang[*], Tianxiang Xing[‡], Jingjing Wang[†], Yong Ren[‡]
[*]Tsinghua Shenzhen International Graduate School, Tsinghua University, Shenzhen, 518055, China
[†] School of Cyber Science and Technology, Beihang University, Beijing, 100191, China.
[‡]Department of Electronic Engineering, Tsinghua University, Beijing, 100083, China

***Abstract*—As a powerful promoter of Internet of Underwater Things (IoUT), unmanned underwater vehicles (UUVs) are widely used in various IoUT applications such as underwater detection and data collection. However, the variable marine environment makes the development and validation of UUV control algorithm face the challenge of high cost and high risk. In this paper, we propose IMTVSim, an integrated modular training and verification simulator for UUVs that integrates customizable modules such as underwater detection model, ocean current model, and 3D underwater scenario, while providing a reinforcement learning (RL) environment to train UUV intelligence to complete the difficult task. In addition, we introduce the large language model to assist the design of reward functions to realize more efficient RL training in our proposed IMTVSim.**

***Index Terms*—Internet of Underwater Things, unmanned underwater vehicle, simulator, reinforcement learning, large language model.**

I. Introduction

As significant drivers of the Internet of Underwater Things (IoUT), unmanned underwater vehicles (UUVs) are extensively utilized in diverse IoUT applications such as underwater exploration and data collection. Nonetheless, the intricate and unpredictable nature of marine environments poses substantial challenges in terms of high costs and elevated risks for the development and testing of UUV control algorithms [1], [2].

Simulators are widely regarded as safe and dependable tools that can generate various test scenarios and amass substantial test data. This capability enhances the efficiency and cost-effectiveness of the design and validation process for robots, contributing to notable successes in both space-based and terrestrial robotic applications in recent years. For instance, Mo *et al*. [4] created Terra, an autonomous vehicle simulation framework, to facilitate efficient navigation in intricate environments. Similarly, Dai *et al*. [5] developed RFlySim, a simulation platform for various types of unmanned aerial vehicles (UAVs), aimed at boosting UAV development efficiency and ensuring safe testing. However, progress in UUV simulation technology has been slower due to the less appealing nature of underwater scenarios and the challenges in accurately simulating the interactions between the marine environment and underwater vehicles.

Although there exist several underwater simulation platforms [7]–[10], they are generally designed for specific tasks and are not easily adaptable to diverse IoUT applications. The primary challenge lies in their limited intelligence, as it is difficult to train agent behaviors using simulation data to effectively complete complex tasks in oceanic environments. Some notable research has suggested viable solutions to the aforementioned challenges. In [11], the researchers introduced a simulation platform designed to model intervention tasks for underwater vehicles, incorporating newly developed plugins that emulate water environment effects, UUV thrusters, and sensors. This work established a foundation for creating a general and intelligent UUV simulation platform. However, this platform lacks sufficient intelligence for testing complex tasks and is not user-friendly. In recent years, reinforcement learning (RL) has proven successful for complex tasks across various robots, including manipulation [13], navigation [14], planning [15], [16], and interaction [17]. Nonetheless, applying RL algorithms to train UUVs in real underwater environments faces hurdles such as low sampling efficiency, unstable training processes, and safety concerns.

The effectiveness of RL training for agents is heavily influenced by the design of the reward function. Traditionally, reward functions for RL algorithms have been manually crafted based on expert experience. This method is not only time-consuming and labor-intensive but also does not ensure the optimal design of the reward function [18]. Recently, inverse RL [19] and preference learning [20] have become popular alternatives for reward function design. These techniques use human preference feedback to create more suitable reward models. However, both approaches still require considerable human effort and extensive data collection, and they often struggle with poor generalization beyond the training data. Fortunately, the advent of the large language model (LLM) addresses this challenge to some extent. Given that LLM inherently captures human preferences, researchers can simply provide the LLM with environment abstractions and task requirements. Subsequently, LLMs can generate effective reward functions for RL training [21].

Based on the above analysis, this paper developed IMTVSim, a simulator for UUVs dedicated in the UPEG task based on ROS and Gazebo, aiming to fill the gap in the field

[+] These authors contribute equally to this work.

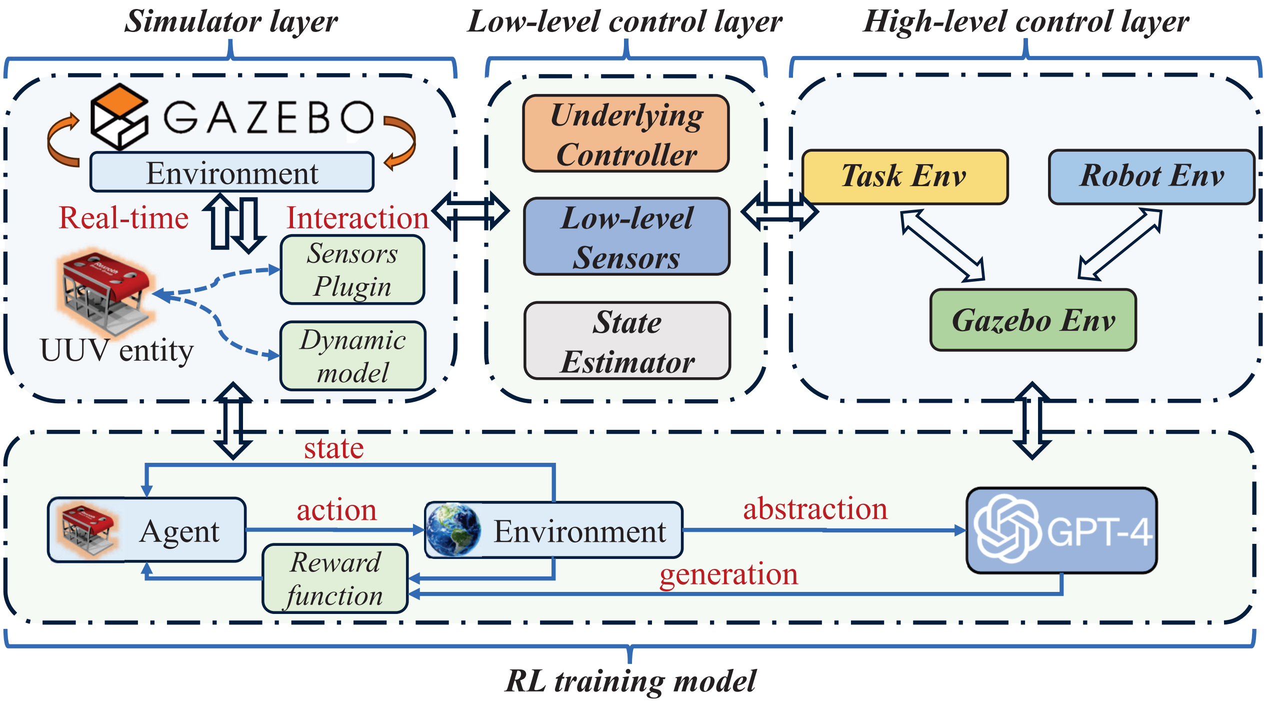

Fig. 1. Illustration of the framework of IMTVSim, which is mainly divided into simulator layer, low-level control layer, high-level control layer, and RL training model (including a LLM interface such as GPT-4 model for reward functions design).

of underwater robot simulation and the UPEG task. Our main contributions can be summarized as follows:

- To the best of our knowledge, this is the first UUV simulator that combines customizable modules such as UUV models, underwater detection model, ocean current model, and 3D underwater scenario, while providing an RL environment to train UUV intelligence to complete the difficult task.
- Given the complexity of the ocean environment and to progressively improve the practicability of the proposed IMTVSim, we use the GPT-4 in LLMs. Just by giving the environment abstractions and task requirements, LLMs can subsequently generate effective reward functions for RL training.

II. Design and Development of IMTVSim

In this section, we first introduce the overall framework of IMTVSim, and then describe the underwater detection model, ocean current model, followed by the construction of 3D underwater scenario in IMTVSim.

A. Overall Framework of IMTVSim

Fig. 1 illustrates the overarching structure of IMTVSim, which is primarily segmented into four layers: the simulator layer, the low-level control layer, the high-level control layer, and the RL training model, including an interface with the LLM for reward functions design. The simulator layer, built on Gazebo, is tasked with creating the UUV simulation entity and the virtual environment. The UUV entity encompasses dynamic models and sensor plugins. The low-level control layer handles essential operations such as state estimation and base-level controllers. Integrating Gazebo with ROS, the high-level control layer is composed of the Gazebo-Environment class (GazeboEnv), the Robot-Environment class (RobotEnv), and the Task-Environment class (TaskEnv). This layer also interfaces with the RL training model and facilitates the use of the LLM to design reward functions by giving environment abstraction and task requirements.

B. Underwater Detection Model

UUVs employ sonar to scan the environment within a restricted range, enabling them to detect nearby obstacles and monitor targets. This detection process can be consistently modeled using the active sonar equation [22], as follows

$$EM = SL - 2TL(f, d) + TS - NL(f) + DI - DT. \quad (1)$$

All parameters in Eq. (1) are measured in dB. In this context, SL denotes the source level, TL represents transmission loss, TS is the target strength associated with the reflection area of the target, NL stands for the environmental noise level, and DI indicates the directionality index. Additionally, DT and EM correspond to the detection threshold and the echo margin of the active sonar, respectively. Moreover, the transmission loss TL is a function of the detection range d and the central acoustic frequency f, which is given by

$$TL = 20\log(d) + d \times a(f) \times 10^{-3}, \tag{2a}$$

$$a(f) = 0.11\frac{f^2}{1+f^2} + 44\frac{f^2}{4100+f^2} + 2.75\times 10^{-4} f^2 + 0.003, \tag{2b}$$

where $a(f)$ is the attenuation coefficient of sound wave in water. When the frequency f is given, the maximum detection radius r_c of the AUV is

$$r_c = \arg\max_d \{EM(d) \geq 0\}. \tag{3}$$

C. Ocean Current Model

The motion of UUV needs to take into account the influence of ocean turbulent environment. We use two-dimensional Navier-Stokes equations [22], [23] to model the ocean turbulent environment as

$$\frac{\partial \varpi}{\partial t} + (\boldsymbol{\mathcal{V}}_c \nabla)\, \varpi = \zeta \Delta \varpi, \tag{4}$$

where $\boldsymbol{\mathcal{V}}_c = (\mathcal{V}_x, \mathcal{V}_y)$ represents the velocity of the current field, while ϖ and ζ denote the vorticity of the current and the viscosity of the fluid, respectively. To streamline the Navier-Stokes equations, the numerical model of the ocean current is expressed through the superposition of several viscous vortex functions, as described below

$$\mathcal{V}_x(\boldsymbol{p}_i(t)) = -\Gamma \cdot \frac{y - y_0}{2\pi \left\|\boldsymbol{p}_i(t) - \boldsymbol{p}_0\right\|_2^2} \cdot \left(1 - e^{-\frac{\|\boldsymbol{p}_i(t) - \boldsymbol{p}_0\|_2^2}{\delta^2}}\right), \tag{5a}$$

$$\mathcal{V}_y(\boldsymbol{p}_i(t)) = -\Gamma \cdot \frac{x - x_0}{2\pi \left\|\boldsymbol{p}_i(t) - \boldsymbol{p}_0\right\|_2^2} \cdot \left(1 - e^{-\frac{\|\boldsymbol{p}_i(t) - \boldsymbol{p}_0\|_2^2}{\delta^2}}\right), \tag{5b}$$

$$\varpi(\boldsymbol{p}_i(t)) = \frac{\Gamma}{\pi \delta^2} \cdot e^{-\frac{\|\boldsymbol{p}_i(t) - \boldsymbol{p}_0\|_2^2}{\delta^2}}, \tag{6}$$

where $\boldsymbol{p}_i(t)$ and $\boldsymbol{p}_0$ represent the current position of UUV i and the coordinate vector of the Lamb vortex center, respectively. $\mathcal{V}_x(\boldsymbol{p}_i(t))$ and $\mathcal{V}_y(\boldsymbol{p}_i(t))$ are the velocities of the ocean current along the X and Y axes perceived by UUV i at position $\boldsymbol{p}_i(t)$ at time t. Furthermore, δ and Γ indicate the radius and strength of the vortex, respectively.

D. Construction of 3D Underwater Scenario

The realism of the virtual ocean environment significantly impacts the simulation's precision, and the primary challenge in creating a 3D underwater scenario lies in accurately modeling the seabed based on real-world terrain data. Our seabed modeling workflow proceeds as follows: initially, we utilize the Anaconda ogr2ogr library to inspect the hierarchical data of the S-57 chart and execute non-visual operations such as format conversion [12]. Subsequently, vector data is transformed into raster data using QGIS or Arcmap software, resulting in a terrain file (.tif). This file is then converted to a height map (.png) with the help of Global Mapper software. To enhance the accuracy of the generated terrain, we adjust the pixel resolution and interpolate any blank areas. The resulting height map is imported into Blender for terrain modeling and the application of realistic textures. The rendered output (.dae file) and texture files (.jpg) are then exported to ROS. Within ROS, these files are combined and the terrain is saved as a .world file using Gazebo. Finally, we manually edit the configuration file to include rigidity parameters, enabling the simulation of accurate collision effects within the Gazebo environment. The visualization process is illustrated in Figs. 2(a) and 2(b).

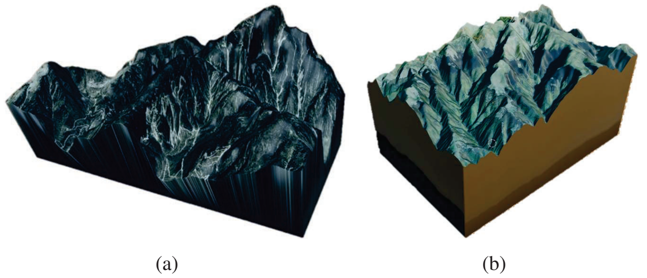

Fig. 2. Seabed terrain construction process via Blender and Gazebo. (a) The seabed terrain modeled by original height map in Blender. (b) The visualization of seabed terrain rendered using Blender.

III. Problem Formulation in IMTVSim

In this section, we first introduce the we first introduce the universal modeling of the task in IMTVSim via Markov decision process (MDP), and then we describe the whole process of reward function design utilizing LLM.

A. Markov Decision Process Modeling

We use RL algorithms to train UUVs for completing corresponding tasks in our proposed IMTVSim. Given a state $\boldsymbol{s}_i$, RL tries to train a UUV to learn a parametric policy π_θ to produce an action $\boldsymbol{a}_i$. The UUV can take this action and transit to the next state $\boldsymbol{s}_{i+1}$ and obtain the corresponding reward r_i. The policy π_θ is learned by finding the optimal parameter θ^* that maximizes the expected total reward

$$J(\theta) = E_{\tau \sim p_\theta(\tau)} \left[\sum_{t=0}^{T} \gamma^t r_t\right], \tag{7}$$

where T is the maximum number of control time steps, γ represents the discounting factor, and τ denotes the sampled trajectory containing a sequence of states and actions.

The process of the RL training can be modeled as a MDP, which can be formulated by a quintuple

$$\mathcal{U} = (\boldsymbol{\mathcal{S}}, \boldsymbol{A}, \mathcal{P}, \boldsymbol{\mathcal{R}}, \gamma), \tag{8}$$

where $\boldsymbol{\mathcal{S}}$, $\boldsymbol{A}$, $\boldsymbol{\mathcal{R}}$ represent state space, action space and reward function, respectively, while $\mathcal{P}$ denotes state transition probability distribution, and $\gamma \in (0, 1)$ is a discount factor.

Given that there are total N UUVs in the environment, so we can respectively represent state space, action space and reward function as follows

$$\boldsymbol{\mathcal{S}} = [\boldsymbol{S}_1, \boldsymbol{S}_2, \cdots, \boldsymbol{S}_{i-1}, \boldsymbol{S}_i], \tag{9}$$

$$\boldsymbol{A} = [\boldsymbol{\mathcal{A}}_1, \boldsymbol{\mathcal{A}}_2, \cdots, \boldsymbol{\mathcal{A}}_{i-1}, \boldsymbol{\mathcal{A}}_i], \tag{10}$$

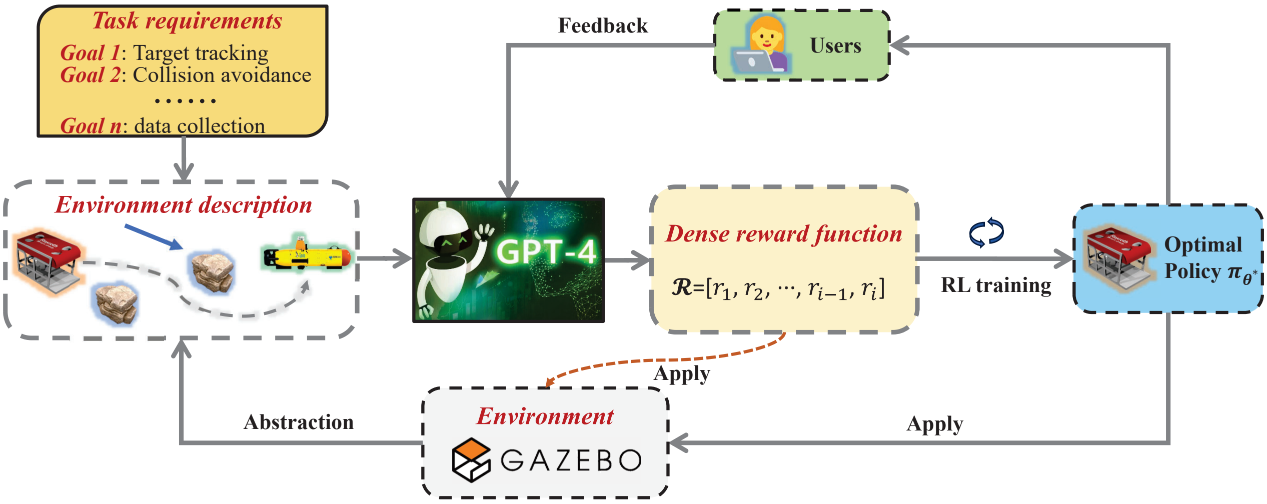

Fig. 3. An overall flow diagram of the reward function design utilizing the LLM such as the GPT-4 model.

$$\boldsymbol{\mathcal{R}} = [r_1, r_2, \cdots, r_{i-1}, r_i], \tag{11}$$

where $\boldsymbol{S}_i$, $\boldsymbol{\mathcal{A}}_i$ and r_i denote the state space, action space and reward function of the ith UUV, respectively.

B. Reward Function Design utilizing LLM

The UUV refines its policy by learning from actions and corresponding rewards, which requires a reward function that effectively balances multiple objectives. To tackle these issues, we employ the GPT-4 model to generate and shape detailed reward function codes based on specified objectives. Once the environment abstraction and optimization objectives are defined, GPT-4 will create the initial dense reward function codes. The detailed reward codes are then incorporated into the RL model in IMTVSim to realize RL training. Nevertheless, due to the sensitivity of RL training, the inherent randomness in LLMs, and potential ambiguities in target descriptions, the initial reward function may not be perfectly aligned with the desired outcomes.

To address this, we implement the trained policy in the environment and refine the reward function design through manual feedback based on observed results. After several iterations of refinement, the reward function becomes well-suited to the training task at hand. And the reward function designed by LLM will be applied to the environment for further RL training, aiming to help the UUV obtain the optimal policy π_θ^*. Fig. 3 illustrates the workflow for designing the reward function using GPT-4.

IV. Conclusion

In this paper, an integrated modular training and verification simulator named IMTVSim for UUVs is developed, with customizable modules while providing a RL environment with the GPT-4 model to assist the design of reward functions, aiming to train UUV intelligence more efficiently to complete complex tasks. Future work will focus on improving the suitability of IMTVSim and real-world environment to address the sim2real challenge.

V. Acknowledgment

This work of Jingjing Wang was partly supported by the National Natural Science Foundation of China under Grant No. 62071268 and No. 62222101, partly supported by the Young Elite Scientist Sponsorship Program by the China Association for Science and Technology under Grant No. 2020QNRC001, and partly supported by the Fundamental Research Funds for the Central Universities. This work of Yong Ren was partly supported by the National Natural Science Foundation of China under Grant 62127801, partly supported by the National Key Research and Development Program of China under Grant 2020YFD0901000. (*Corresponding author: Jingjing Wang*)

References

[1] W. Bessa, M. Dutra and E. Kreuzer, “Dynamic positioning of underwater robotic vehicles with thruster dynamics compensation,” *Int. J. Adv. Robot. Syst.*, vol. 10, no. 9, pp. 1-8, Apr. 2013.

[2] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, “UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation,” in *OCEANS MTS/IEEE Monterey,* Monterey, CA, USA, Sep. 2016, pp. 1-8.

[3] Z. Zhang, W. Mi, J. Du, Z. Wang, W. Wei, Y. Zhang, Y. Yang, and Y. Ren, “Design and implementation of a modular UUV simulation platform,” *Sensors,* vol. 22, no. 20, pp. 8043, Oct. 2022.

[4] Y. Mo, S. Ma, H. Gong, Z. Chen, J. Zhang, and D. Tao, “Terra: A smart and sensible digital twin framework for robust robot deployment in challenging environments,” *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14039-14050, Sep. 2021.

[5] X. Dai, C. Ke, Q. Quan, and K. Y. Cai, “RFlySim: Automatic test platform for UAV autopilot systems with FPGA-based hardware-in-the-loop simulations,” *Aerosp. Sci. Technol.*, vol. 114, Jul. 2021, Art no. 106727.

[6] M. R. Kabir, B. B. Y. Ravi, and S. Ray, “A virtual prototyping platform for exploration of vehicular electronics,” *IEEE Internet Things J.*, vol. 10, no. 18, pp. 16144-16155, Sep. 2023.

[7] L. Hong, X. Wang, D. S. Zhang, M. Zhao, and H. Xu, “Vision-based underwater inspection with portable autonomous underwater vehicle: Development, control, and evaluation,” *IEEE Trans. Intell. Veh.*, vol. 9, no. 1, pp. 2197-2209, Jan. 2024.

[8] Z. Zhang, J. Xu, G. Xie, J. Wang, Z. Han, and Y. Ren, "Environment- and energy-aware auv-assisted data collection for the internet of underwater things," *IEEE Internet Things J.*, vol. 11, no. 15, pp. 26406–26418, 2024.

[9] O. Álvarez-Tuñón, H. Kanner, L. R. Marnet, H. X. Pham, J. L. F. Sejersen, Y. Brodskiy, and E. Kayacan, "Mimir-UW: A multipurpose synthetic dataset for underwater navigation and inspection," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Detroit, MI, USA, Oct. 2023, pp. 6141-6148.

[10] X. Hou, J. Wang, T. Bai, Y. Deng, Y. Ren, and L. Hanzo, "Environment-Aware AUV Trajectory Design and Resource Management for Multi-Tier Underwater Computing," *IEEE Journal on Selected Areas in Communications.* vol. 41, no. 2, pp. 474-490, Feb. 2023.

[11] M. M. M. Manhães, S. A. Scherer, M. Voss, L. R. Douat, and T. Rauschenbach, "UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation," in *OCEANS MTS/IEEE Monterey,* Monterey, CA, USA, Sep. 2016, pp. 1-8.

[12] Z. Zhang, W. Mi, J. Du, Z. Wang, W. Wei, Y. Zhang, Y. Yang, and Y. Ren, "Design and implementation of a modular UUV simulation platform," *Sensors,* vol. 22, no. 20, pp. 8043, Oct. 2022.

[13] J. Pitz, L. Röstel, L. Sievers, and B. Bäuml, "Dextrous tactile in-hand manipulation using a modular reinforcement learning architecture," in *Proc. IEEE Int. Conf. Robot. Autom.*, London, UK, May-Jun. 2023, pp. 1852-1858.

[14] S. S. Samsani and M. S. Muhammad, "Socially compliant robot navigation in crowded environment by human behavior resemblance using deep reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 6, no. 3, pp. 5223-5230, Jul. 2021.

[15] J. Kumar, C. S. Raut, and N. Patel, "Automated flexible needle trajectory planning for keyhole neurosurgery using reinforcement learning," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Kyoto, Japan, Oct. 2022, pp. 4018-4023.

[16] X. Hou, J. Wang, C. Jiang, Z. Meng, J. Chen, and Y. Ren, "Efficient Federated Learning for Metaverse via Dynamic User Selection, Gradient Quantization and Resource Allocation," *IEEE Journal on Selected Areas in Communications.* vol. 42, no. 4, pp. 850-866, Apr. 2024.

[17] P. Liu, K. Zhang, D. Tateo, S. Jauhri, Z. Hu, J. Peters, and G. Chalvatzaki, "Safe reinforcement learning of dynamic high-dimensional robotic tasks: navigation, manipulation, interaction," in *Proc. IEEE Int. Conf. Robot. Autom.*, London, UK, May-Jun. 2023, pp. 9449-9456.

[18] A. Ng, D. Harada, and S. J. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *International Conference on Machine Learning.* PMLR, 1999, pp. 278-287.

[19] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *National Conference on Artificial Intelligence.* AAAI, 2008, pp. 1433-1438.

[20] K. Lee, L. M. Smith, and P. Abbeel, "Pebble: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training," in *International Conference on Machine Learning.* PMLR, 2021, pp. 6152-6163.

[21] T. Xie, S. Zhao, C. H. Wu, Y. Liu, Q. Luo, V. Zhong, Y. Yang, and T. Yu, "Text2reward: Reward shaping with language models for reinforcement learning," in *International Conference on Learning Representations*. PMLR, 2024, pp. 1–37.

[22] X. Hou, J. Wang, T. Bai, Y. Deng, Y. Ren, and L. Hanzo, "Environment-aware AUV trajectory design and resource management for multi-tier underwater computing," in *IEEE J. Sel. Areas Commun.*, vol. 41, no. 2, pp. 474–490, Feb. 2023.

[23] S. Shuai and M. H. Kasbaoui, "Accelerated decay of a Lamb–Oseen vortex tube laden with inertial particles in Eulerian–Lagrangian simulations," in *J. Fluid Mech.*, vol. 936, 2022.