Meta-Critic Reinforcement Learning for IOS-Assisted Multi-User Communications in Dynamic Environments

Qinpei Luo*, Boya Di*, Zhu Han+
* State Key Laboratory of Advanced Optical Communication Systems and Networks, School of Electronics, Peking University
+ Electrical and Computer Engineering Department, University of Houston, TX, USA
{luoqinpei, diboya}@pku.edu.cn, zhan2@uh.edu

2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring) | 979-8-3503-1114-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/VTC2023-Spring57618.2023.10199513

***Abstract*—Capable of reflecting and refracting the incident signals on both sides simultaneously, the intelligent omnidirectional surface (IOS) has recently been proposed as a promising solution to enhance the capacity of wireless networks. However, the large number of IOS elements brings a heavy burden to the beamforming scheme design, especially for applications that require a fast response to varying environments. In this paper, aiming to maximize the sum rate of an IOS-aided multi-user system via IOS-enabled beamforming design that can rapidly adapt to dynamic channel states and user mobility, we develop a novel meta-critic reinforcement learning framework where a meta-critic network recognizes the environment change and automatically re-trains of the learning model. A stochastic Explore and Reload procedure is tailored to reduce the high-dimensional action space problem. Simulation results show the proposed scheme can converge to a higher sum rate more rapidly compared to the benchmark methods in dynamic settings. The robustness of our scheme against different IOS sizes is also verified.**

***Index Terms*—Intelligent omni-surface, Beamforming, Meta-learning, Dynamic environments.**

I. Introduction

Recently, the development of metasurfaces has provided an efficient method to improve network coverage as well as spectrum efficiency. One typical metasurface, intelligent omni-surface (IOS), has attracted great attention owing to its ability of simultaneous signal reflection and refraction towards target directions [1]. By designing the phase shifts of the IOS elements, users on both sides of the IOS can be served, thereby supporting omnidirectional communications.

Though the IOS can significantly enhance the data rate of multi-user systems, it also induces a heavy computational burden since the vast number of IOS elements renders the hybrid beamforming scheme non-trivial where the phase shifts of all IOS elements should be optimized. This may degrade the system performance, especially in a dynamic environment that requires real-time updates of IOS phase shift configuration. It is thus important to develop an efficient beamforming scheme to tackle numerous IOS elements adapting to the varying channel information, users' positions, etc.

Recently, machine learning-based beamforming schemes have been explored toward this target. Especially, reinforcement learning (RL) is often used since it can well depict the interaction process between the surface and the environment rather than merely collecting datasets as in supervised machine learning methods. In [2], the deep reinforcement learning method was introduced to improve the sum rate of a MISO system. The work [3] adopted a similar method to maximize energy efficiency. One mostly recent work [4] introduced multi-agent reinforcement learning to improve the sum rate with multiple intelligent surfaces collaboratively.

However, most existing RL-related works either considered a small-sized metasurface with no more than 32 elements or a simplified case where the channel information and locations of users are almost static. In this case, every time the environment changes, the RL network needs to be retrained to update the IOS beamformer, which brings inevitably huge overhead. Moreover, the meta-surface deployed in practice usually consists of hundreds or thousands of elements, making it even harder for the beamforming scheme to update. Against this background, we aim to develop an efficient beamforming scheme to address practical concerns: (1)*How to adapt to the dynamic case where the channel information and user positions vary with time?* (2)*How to deal with the numerous phase shift variables brought by a large-scale IOS in this case?*

In this paper, we propose a meta-critic deep deterministic policy gradient algorithm (MC-DDPG) to solve the above issues. We design a meta-critic and pre-train it to collaboratively learn from multiple tasks and then apply the meta-critic to the newly coming task. A stochastic Explore and Reload procedure is introduced to avoid the solution deviating from the optimal point due to high-dimensional action space. Simulation results show that our MC-DDPG is able to converge to a solution close to the optimization faster when environment changes, compared to the iterative optimization and traditional RL methods with no meta-critic introduced. Besides, we also test our algorithm under different numbers of IOS elements, which proves surfaces with more elements can improve the performance of communication systems.

The rest of this paper is organized as follows. Section II and Section III present the system model and problem formulation. In Section IV, we propose our MC-DDPG algorithm to solve the defined problem. Simulation results are shown in Section V. Finally, we draw the conclusions in Section VI.

This work was supported in part by the National Key R&D Project of China under Grant No. 2022YFB2902800, National Natural Science Foundation Grant 62271012 and 62227809, and Beijing Natural Science Foundation under Grant 4222005.

II. System Model

In this section, we first describe the IOS-assisted multi-user communication system and then present the detailed IOS and channel models.

A. Scenario Description

As shown in Fig. 1, we show a downlink multi-user MISO wireless communication system with a M-antenna BS serving K users equipped with one antenna. Due to the shadowing effect brought by obstacles like tall buildings and unexpected fading of propagation paths, the Light-of-Sight (LOS) channel between the BS and users is often unstable and suffers from severe fading. To improve the performance of the system, an IOS consisting of N elements is deployed between the users and the BS to reflect or refract the transmit signals towards both sides of it. We consider different dynamic cases where the channel states and locations of users may vary with time.

B. Intelligent Omni-Surface Model

Different from the reflecting type metasurface (also known as RIS), IOS can manipulate electromagnetic waves with a dual function of simultaneously reflecting and refracting signals. Each element of IOS is a sub-wavelength with the size of $\delta_w \times \delta_h$ and is capable of performing 2^b possible amplitude/phase shifts to reflect and refract the incident ray [5]. The dual function of IOS can be expressed by [6]

$$\Delta^t = \sqrt{\frac{1}{1+\epsilon}}, \Delta^r = \sqrt{\frac{\epsilon}{1+\epsilon}}, \tag{1}$$

in which ϵ is the refraction-reflection ratio. Δ^t and Δ^r are the energy spilled for the reflected and refracted signals, respectively. The response Γ_n of the n-th IOS element can be given by

$$\Gamma_n = \Delta^n q_n, \tag{2}$$

$$q_n = \sqrt{G_n F_A F_D \delta_w \delta_h |\gamma_n|^2} e^{-j\theta_n}, \tag{3}$$

in which G_n is the antenna gain of the $n-th$ IOS element. γ_n is refer to the power ratio of the reflective or refractive signal. F_A and F_D are the normalized power radiation patterns of the incident signal and departure signals determined by the angles between the arrival/departure signals and the x-axis.

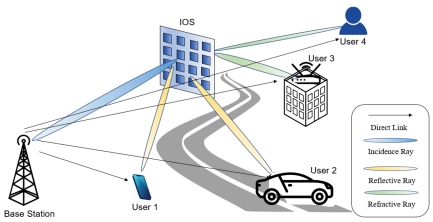

Fig. 1. System model of the IOS-assisted multi-user system

C. Channel Model

We denote the whole channel of the IOS-assisted system from two parts, of which the first is the direct channel between the BS and K users as $\mathbf{H}_{BU} \in \mathbb{C}^{K\times M}$. The two-hop reflective/refractive channel that corresponds to the BS-IOS link and the IOS-user link can be denoted by $\mathbf{H}_{BI} \in \mathbb{C}^{N\times M}$ and $\mathbf{H}_{IU} \in \mathbb{C}^{K\times N}$, respectively. According to Saleh-Valenzuela model [7], the channel matrices can be expressed by

$$\mathbf{H}_{BI} = \sqrt{S_1}\boldsymbol{A}_I \boldsymbol{\Sigma}_{BI} \boldsymbol{D}_B^H, \tag{4}$$

$$\mathbf{H}_{IU,k} = \sqrt{S_{2,k}}\boldsymbol{A}_{IU,k} \boldsymbol{\Sigma}_{IU,k} \boldsymbol{D}_{I,k}^H, \tag{5}$$

$$\mathbf{H}_{BU,k} = \sqrt{S_{3,k}}\boldsymbol{A}_{BU,k} \boldsymbol{\Sigma}_{BU,k} \boldsymbol{D}_{B,k}^H, \tag{6}$$

where $\boldsymbol{D}_B, \boldsymbol{D}_{IU,k}, \boldsymbol{D}_{BU,k}$ and $\boldsymbol{A}_I, \boldsymbol{A}_{IU,k}, \boldsymbol{A}_{BU,k}$ represent the transmit/receive steering matrices. The i-th columns of each $\boldsymbol{D}$ and $\boldsymbol{A}$ are channel steering vectors, it can be expressed by $\boldsymbol{f}(N,\theta) = \frac{1}{\sqrt{N}}[1, e^{j\pi\theta}, \ldots, e^{j(N-1)\pi\theta}]^H$, in which N is the number of antennas and θ is the angle-of-arrival (AoA) or angle-of-departure (AoD). We set the matrices $\boldsymbol{\Sigma}_{BI} = diag(\sqrt{\frac{NN_b}{I_1}}[\lambda_{BI,1}, \ldots, \lambda_{BI,I_1}])$, $\boldsymbol{\Sigma}_{IU} = diag(\sqrt{\frac{NN_u}{I_2}}[\lambda_{IU,1}, \ldots, \lambda_{IU,I_2}])$ and $\boldsymbol{\Sigma}_{BU} = diag(\sqrt{\frac{N_bN_u}{I_3}}[\lambda_{BU,1}, \ldots, \lambda_{BU,I_3}])$, where I_1, I_2 and I_3 are the numbers of links of each channel. For the i-th link, $\lambda_{BI,i}, \lambda_{IU,i}$ and $\lambda_{BU,i}$ denote the channel gain.

For users in the reflective or refractive zone of the IOS, the LoS component of the equivalent end-to-end channel from BS to user k can be given as

$$\mathbf{H}_k^{LoS} = \Delta^u \mathbf{H}_{IU,k} \boldsymbol{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}, k \in \mathcal{K}_u, \tag{7}$$

in which $\boldsymbol{\Theta} = diag([e^{j\theta_1}, \cdots, e^{j\theta_N}])$, $[e^{j\theta_1}, \cdots, e^{j\theta_N}]$ being the phase configuration of IOS elements. $\mathcal{K}_u$ refers to the set of users, while $u \in \{r, t\}$ refers to the reflective and refractive users respectively.

We assume that the equivalent channel of each user follows the Rician distribution, i.e., $\mathbf{H}_k$ consists of both LoS and NLoS components,

$$\mathbf{H}_k = \sqrt{\frac{K}{K+1}}\mathbf{H}_k^{LOS} + \sqrt{\frac{1}{K}}\mathbf{H}_k^{NLOS}, \tag{8}$$

where K is the Rician factor.

Based on the above channel model, we further construct Finite State Markov Channel (FSMC). We choose to fix the LOS component and discretize the NLOS channel $\boldsymbol{H}^{NLOS}$ into L levels, i.e., $\mathcal{H} = \mathbf{H}_1, \ldots, \mathbf{H}_L$. The AoAs and AoDs of NLOS channel on each level are generated randomly. The transition probability matrix is defined as

$$\mathbf{P} = \begin{bmatrix} p_{1,1} & \cdots & p_{1,L} \\ \vdots & \ddots & \vdots \\ p_{L,1} & \cdots & p_{L,L} \end{bmatrix}, \tag{9}$$

where the transition probability $p_{l,l'}$ can be written as

$$p_{l,l'} = Prob[\mathbf{H}_{t+1} = \mathbf{H}_{l'} | \mathbf{H}_t = \mathbf{H}_l], \mathbf{H}_l, \mathbf{H}_{l'} \in \mathcal{H}. \tag{10}$$

The equation above indicates that given the channel state $\mathbf{H}_t = \mathbf{H}_l$ at time slot t, $p_{l,l'}$ refers to the probability of channel state at the next time slot $\mathbf{H}_{t+1}$ transiting from $\mathbf{H}_l$ to $\mathbf{H}_{l'}$. Without loss of generality, we generate $\mathbf{P}$ randomly to depict the time-varying NLoS channel.

III. Problem Formulation

In this section, we will formulate the sum rate maximization problem, and then reformulate it into a Markov decision process (MDP) to develop a reinforcement learning method.

A. Sum Rate Maximization Problem

We consider the sum rate maximization problem in T time slots, each of which has a duration of ΔT. In every time slot, the channel state keeps stable, while between different time slots. The received signal of user k in time slot t can be written as

$$y_{k,t} = (\Delta_k \mathbf{H}_{IU,k} \mathbf{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}) \sum_{j=1}^{K} \boldsymbol{V}_{j,t} m_j + n_{k,t}, \quad (11)$$

where Δ_k could be Δ^r or Δ^t determined by whether the user is receiving a reflective or refractive signal. $\boldsymbol{V}_{j,t}$ refers to the digital beamforming vector from BS to j-th user.m_k denotes the symbol BS sends to user j.$n_{k,t}$ represent Gaussian noise which follows $N(0, \sigma_{k,t}^2)$.The Signal-to-Noise(SNR) of user k can be expressed as

$$\gamma_{k,t} = \frac{|(\Delta \mathbf{H}_{IU,k} \mathbf{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}) \boldsymbol{V}_{k,t} m_k|^2}{|(\Delta \mathbf{H}_{IU,k} \mathbf{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU,k}) \sum_{j=1,\neq k}^{K} \boldsymbol{V}_{j,t} m_j|^2 + \sigma_{k,t}^2}, \quad (12)$$

from which we can express user k's data rate as,

$$R_{k,t} = |\Delta T log(1 + \gamma_{k,t})|. \quad (13)$$

The sum rate maximization problem can be formulated as

$$\begin{aligned} \mathbf{P1}: & \max_{\boldsymbol{V}_t, \mathbf{\Theta}_t} \sum_{t=1}^{T} \sum_{k=1}^{K} R_{k,t}, \\ & s.t. Tr(\boldsymbol{V}_t^H \boldsymbol{V}_t) \leq P_T, t = 1, \cdots, T, \end{aligned} \quad (14)$$

where P_T refers to the total transmission power. Given $\mathbf{\Theta}_t$, through zero-force (ZF) beamforming and water-filling algorithm [8], we can get a sub-optimal solution of $\boldsymbol{V}_t$ directly.

B. MDP Reformulation

For a fixed digital beamforming scheme such as the zero-forcing [8] or MMSE one, we can rewrite P1 as

$$\mathbf{P2}: \max_{\mathbf{\Theta}_t} \sum_{t=1}^{T} \sum_{k=1}^{K} R_{k,t}. \quad (15)$$

Given the time-varying characteristics of channels, we then reformulate $\mathbf{P2}$ as a MDP with the following components.

1) **Action**: The action in the MDP is the configuration of phases of all IOS elements, defined by

$$a_t = \mathbf{\Theta}_t, \forall \theta_t \in \mathbf{\Theta}_t, \theta_t \in (-\pi, \pi). \quad (16)$$

2) **State**: The state in the MDP refers to the pair of channel states and the IOS phase shift matrix/configuration. The channel state is measured by the equivalent channel between the BS and users, i.e.,

$$\mathbf{H} = diag(\mathbf{\Delta}) \mathbf{H}_{IU} \mathbf{\Theta} \mathbf{H}_{BI} + \mathbf{H}_{BU}, \quad (17)$$

where $\mathbf{\Delta} = [\Delta_1, \cdots, \Delta_K]$. The IOS phase shift configuration in each time slot $t-1$ is given by $\mathbf{\Theta}_{t-1}$, with $\mathbf{\Theta}_0 = [0, \cdots, 0]$ being the initial configuration. Then the state of the MDP can be defined by

$$s_t = \{\mathbf{H}_t, \mathbf{\Theta}_{t-1}\}. \quad (18)$$

3) **Reward**: The reward of the MDP is consistent with the objective value of $\mathbf{P2}$, i.e., the sum rate of all users in time slot t. To avoid the high variation issue brought by the high value of the reward, we multiply the sum rate by a coefficient η, so the reward can be expressed by

$$r_t = \eta \sum_{k=1}^{K} \gamma_{k,t}. \quad (19)$$

Then the accumulated reward at step t is given by $\overline{r_t} = \sum_{t'=t}^{T} \delta^{t'-t} r_{t'}$, where $\delta \in [0, 1]$ is the discount factor. If $\delta = 1$, then maximizing reward is equivalent to the optimization problem $\mathbf{P2}$.

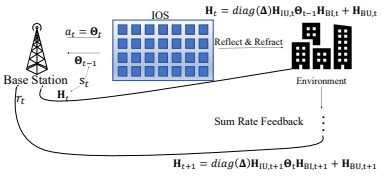

Fig. 2. MDP Process of IOS configuration

The whole process is shown in Fig. 2. At time slot t, the BS acquires the equivalent channel information $\mathbf{H}_t$ from the environment and records the IOS configuration $\mathbf{\Theta}_{t-1}$ at the previous time slot $t-1$.First, the BS uses Zero-Forcing method [8] to give the digital beamforming vector $\boldsymbol{V}_t$. Then it determines the action a_t, i.e., the IOS configuration $\mathbf{\Theta}_t$ at current time slot t, to maximize its own expected reward (19). The BS then transmits the signals to users and obtains the reward of the current time slot t, i.e., the sum rate of all users.

IV. MC-DDPG Algorithm Design

In this section, we first introduce our proposed MC-DDPG framework, and then illustrate its detailed design.

A. MC-DDPG Algorithm Framework

A hierarchical design is shown in Fig. 3a and Fig. 3b, consisting of the meta-learning phase and online learning phase. Unlike the traditional actor-critic framework [9] where each actor is paired with a critic, we design a meta-critic to replace the traditional critics. The experience from all actors will be sent to one meta-critic for updating the Q-value evaluation policy.

1) Task: The blue block refers to the task of RL, which denotes a process of the BS maximizing the sum rates of all users in a fixed number of time slots. For different tasks, the parameters of the BS and IOS are set as the same, while the channel states and locations of users are various.

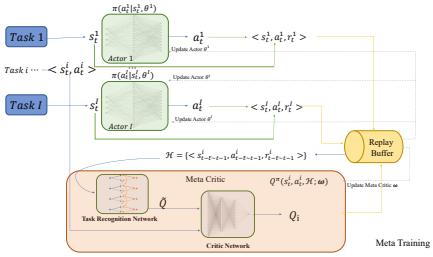

Fig. 3a. Framework of MC-DDPG: Meta Learning Phase

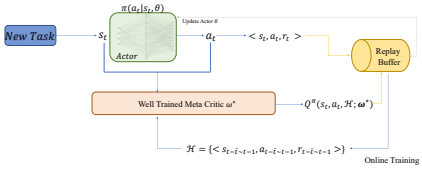

Fig. 3b. Framework of MC-DDPG: Online Learning Phase

2) Actor: The green block in Fig. 3a and Fig. 3b represents the actor of the RL method. It receives the information of state s_t^i from task i in time slot t and outputs correspondent action a_t^i. To better capture the features of continuous action space, we adopt a neural network as the policy of the actor, which will be discussed in detail in Section IV.B.

3) Meta Critic: The meta-critic can be divided into two parts, a task recognition network, and a critic network. Given a specific task, the task recognition network extracts the history information and generates the task-recognition Q-value, which is sent to the critic network together with the state-action information of this task. The critic network outputs a task-specific Q-value to update the actor networks [10].

Meta Learning Process Description: The meta-learning phase can be described as follows: First, for each task i, its current state s_t^i is fed to actor i so the actor can generate an action a_t^i from its learned policy $\pi(a|s_t^i)$. Task i operates the action and receives the reward r_t^i from the environment, then the state-action-reward information $< s_t^i, a_t^i, r_t^i >$ will be stored in the replay buffer. Meanwhile, the meta critic collects the history information of task i, i.e., $\mathcal{H}_t = \{(s_k, s_{k+1}, a_k, r_k)\}, k \in [t-\bar{t}, t-1]$ from the replay buffer together with the state-action pair $< s_t^i, a_t^i >$ to give a task-specific Q-value for updating the actor networks. The meta-critic is also updated by the trajectories of all tasks in the replay buffer, which will be detailed in Section IV.B.

In the online learning phase, for a newly-coming real-time task, the update of the actor network is the same as that of the meta-learning phase. Differently, the critic is kept static during online learning. We directly use the well-trained meta-critic to evaluate the task and then update the actor.

B. Detailed Design of MC-DDPG

In this subsection, we explain the key parts of our proposed MC-DDPG algorithm, including the tailored design of meta critic along with actors and the Explore and Reload procedure we introduced.

1) Tailored Description of Meta Critic and Actors: Our meta critic is designed based on the TD3 structure which consists of two Q-networks for more accurate Q-value estimation [11]. We use two neural networks (NN) with weights $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2$ to parameterize the meta critic, and each actor of task i is also modeled as an NN policy $\pi(a|s_t^i, \boldsymbol{\theta}^i)$ with weights $\boldsymbol{\theta}^i$.

In the dynamic case, the distribution of trajectories of observed data from different tasks may deviate. We aim to design a critic that is capable of identifying different tasks by collecting the history of time-related series of them. For that purpose, we introduce the Long-Short term Memory (LSTM) networks as *task recognition* network which outputs the task recognition Q-value. Following the interaction process we give in Fig. 2, we define the task i's history as a segment of tuples of state, action, and reward from step u to step v, i.e., $\mathcal{H}_{u\sim v}^i = \{s_u^i, a_u^i, r_u^i, \ldots, s_v^i, a_v^i, r_v^i\}$. For simplicity, we directly use the most recent $\bar{t}$ examples as the input of the task recognition network.

We adopt a four-layer full connected network (FNN) to learn the features from the current state-action pairs and the task recognition Q-value from LSTM. Aggregately, the meta-critic will first recognize the coming task, and then accurately criticize the actor's action in a specific state by giving a task-specific Q-value. We desire to pre-train a meta-critic that is able to instruct the actor to rapidly adapt to the varying environment given any new tasks. Such a meta-critic can be expressed by

$$\tilde{Q}_k = f_{LSTM}(\mathcal{H}^i_{[t-\bar{t},t-1]}; \boldsymbol{\omega}_k^{LSTM}), k = 1, 2, \tag{20}$$

$$Q(s_t^i, a_t^i, \mathcal{H}^i_{[t-\bar{t},t-1]}; \boldsymbol{\omega}_k) = f_{FNN}(s_t^i, a_t^i, \tilde{Q}_k; \boldsymbol{\omega}_k^{FNN}), k = 1, 2, \tag{21}$$

where the $\tilde{Q}$ denotes the task recognition Q-value.

As for the actor, we still use FNN as our policy network. It takes current state s_t as input and outputs deterministic action

$$\pi(a|s_t^i; \boldsymbol{\theta}^i) = f_{FNN}(s_T^i; \boldsymbol{\theta}^i) \tag{22}$$

2) Loss Function: To train the policy and value networks, we first define the loss functions of the actor and critic networks, respectively, then minimize them based on the back propagation method. For the meta critic, we use temporal difference (TD) error of all tasks as the loss function

$$L(\boldsymbol{\omega}_k) = \frac{1}{I}\sum_{i=1}^{I} \mathbb{E}_{\pi(\boldsymbol{\theta}^i)}[Q(s_t^i, a_t^i, \mathcal{H}^i_{[t-\bar{t},t-1]}; \boldsymbol{\omega}_k) - (r_t + \gamma min_k Q(s_{t+1}^i, a_{t+1}^i, \mathcal{H}^i_{[t-\bar{t}+1,t]}; \boldsymbol{\omega}_k))]^2, k = 1, 2, \tag{23}$$

where I refers to the number of tasks. TD error in (23) represents the similarity between the estimated Q-value of two critic networks and target Q-value. As for the actor of each

task, the loss function can be represented by the negative Q-value

$$J(\boldsymbol{\theta}^i) = \mathbb{E}_{\pi(\boldsymbol{\theta}^i)}[-Q(s_t^i, a_t^i, \mathcal{H}_{[t-\bar{t},t-1]}^i; \boldsymbol{\omega}_1)], \tag{24}$$

which is equivalent to maximizing the expected accumulated reward (19). The parameters update can be expressed as

$$\boldsymbol{\omega}_{t+1} = \boldsymbol{\omega}_t - \rho \nabla_{\boldsymbol{\omega}} L(\boldsymbol{\omega}), \tag{25}$$

$$\boldsymbol{\theta}_{t+1}^i = \boldsymbol{\theta}_t^i - \rho \nabla_{\boldsymbol{\theta}^i} L(\boldsymbol{\theta}^i), \tag{26}$$

3) Explore & Reload procedure: Traditional environments with continuous action space have relatively low dimensions, however, when it comes to a high dimensional problem as we proposed in Section III.C, fast convergence may become a difficult problem. When the dimension of action space is too high, it is often hard to find a good policy to schedule the learning rate. Thus, we design a stochastic Explore and Reload procedure and set two thresholds Th_{reward} and Th_{eps}. We introduce noise to enhance the exploration of agent

$$a_t = \pi(a|s_t) + e, \tag{27}$$

where $e \sim \mathcal{N}(0, \epsilon)$. Before training, we initialize ϵ as ϵ_0. Then in each training episode, we record the maximum reward $reward_{max}$ and the corresponding model of policy π^{max}. For the sake of convergence, we set ϵ to be exponentially decaying as the number of episodes grows until convergence. If the current reward declines beyond Th_{reward} compared to $reward_{max}$, i.e.,

$$reward_{max} - reward_{current} > Th_{reward} \tag{28}$$

or $reward_{max}$ has not been updated for Th_{eps} episodes, the actor reloads the best recorded model π^{max} and resets the exploration noise $\epsilon = \epsilon_0$ to restart the exploration.

C. Algorithm Description

We summarize the MC-DDPG algorithm in Alg.1. At the beginning of meta training, we initialize the network parameters of actors and the meta critic with replay buffers allocated to each actor. For each episode, we select I learning tasks and initialize them (Line 5). In one task at each step, we select an action and add exploration noise on it before sending it to IOS with a reward and the next state fed back (Line 8-9). Then we sample recent $\bar{t}$ examples and store them in replay buffer with state, next state, action, and reward (Line 11). Line 13 is the process of updating the meta critic, while Lines 15-17 aim to update the parameters of the actor policy network of task i and target networks, where d denotes the actor update interval. Lines 18-22 describe the model reloading process. The output of the meta training is a well-trained meta critic $\boldsymbol{\omega}^\star$ (Line 23).

In the online learning stage, the agent directly uses well-trained meta critic $\boldsymbol{\omega}^\star$ to estimate Q-value, and thus only the actor needs to be trained. In line 27, we initialize the policy network and replay buffer. The actor determines the agent's action and stores the tuple $< s_t, s_{t+1}, a_t, r_t >$ in the buffer as shown in Lines 31-34. The actor is then updated and reloaded (Lines 36-40). The output of online learning is the trained policy of actor $\boldsymbol{\theta}^\star$ (Line 41).

Algorithm 1 MC-DDPG Algorithm

1: **Meta Training:**
2: **input:** Multiple task samples.
3: **Initialize:** (For each task i) Critic Networks $Q_{\boldsymbol{\omega}_1}, Q_{\boldsymbol{\omega}_2}$, and actor-network π_{θ^i} with parameters $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \boldsymbol{\theta}^i$; Target Networks $\boldsymbol{\omega}_1' \leftarrow \boldsymbol{\omega}_1, \boldsymbol{\omega}_2' \leftarrow \boldsymbol{\omega}_2, \boldsymbol{\theta}^{i\prime} \leftarrow \boldsymbol{\theta}^i$;Replay Buffer $\mathcal{R}^i$;
4: **for** eps in range(*MaxEpisode*) **do**
5: Sample I tasks and initialize states $s_0^1, \cdots, s_0^I$.
6: **for** t in range(*MaxStep*) **do**
7: **for** each task i **do**
8: Select action by the actor policy with exploration noise (27).
9: Get reward r_t^i and next state s_{t+1}^i.
10: Select recent $\bar{t}$ samples to form history.
11: Store transition tuple $(s_t^i, s_{t+1}^i, a_t^i, r_t^i, \mathcal{H}_t^i)$ into $\mathcal{R}^i$.
12: Sample a batch of N transitions $(s^i, s^{i\prime}, a^i, r^i, \mathcal{H}^i)$ from $\mathcal{R}^i$.
13: Update Meta Critic by (23) and (25)
14: **if** t mod d **then**
15: Update $\boldsymbol{\theta}^i$ by (24) and (26)
16: $\boldsymbol{\omega}_k' \leftarrow \tau \boldsymbol{\omega}_k + (1-\tau)\boldsymbol{\omega}_k'$.
17: $\boldsymbol{\theta}^{i\prime} \leftarrow \tau \boldsymbol{\theta}^i + (1-\tau)\boldsymbol{\theta}^{i\prime}$.
18: **for** each task i **do**
19: **if** current reward $>$ max reward **then**
20: Store policy model and update the max reward.
21: **if** conditions defined in IV-B3 are satisfied **then**
22: Reload policy model and exploration noise.
23: **Output:** Well-trained meta critic $\boldsymbol{\omega}^\star$.
24: ———————————————————————
25: **Online Training:**
26: **input:**A new task;Well-trained meta critic $\boldsymbol{\omega}^\star$.
27: **Initialize:** Policy network $\boldsymbol{\theta}_0$; Replay Buffer $\mathcal{R}$;
28: **for** eps in range(*MaxEpisode*) **do**
29: **Initialize** system state s_0.
30: **for** t in range(*MaxStep*) **do**
31: Select action from the policy with exploration noise by (27).
32: Get reward r_t and next state s_{t+1}.
33: Sample recently $\bar{t}$ samples to form history.
34: Store transition tuple $(s_t, s_{t+1}, a_t, r_t, \mathcal{H}_t)$ into $\mathcal{R}$.
35: Sample a batch of N transitions $(s, s', a, r, \mathcal{H})$ from $\mathcal{R}$.
36: Update $\boldsymbol{\theta}$ by (24) and (26)
37: **if** current reward $>$ max reward **then**
38: Store policy model and update the max reward.
39: **if** conditions defined in IV-B3 are satisfied **then**
40: Reload policy model and exploration noise.
41: **Output:** The trained policy of actor $\boldsymbol{\theta}^\star$.

Remark: The MC-DDPG algorithm is guaranteed to converge. This is because we keep decaying exploration noise and learning rate, with model reloading introduced.

V. Simulation Results

In this section, we evaluate the proposed MC-DDPG approach in dynamic settings. The performance of MC-DDPG is compared to two benchmark algorithms. [8].

TABLE I
Simulation Parameters

Parameter	Value
Total number of users K	4
Number of antennas on BS M	8
Number of IOS elements	{4,16,64,100}
Time limitation T	20 time slots
Carrier frequency f_c	5.9 GHz
Noise power spectral density	-93 dBm/Hz
Learning rate ρ	$1e-5$
Discount Factor γ	0.99

A. Simulation Setup

Major parameters of the simulation are summed in Table I. We assume that the task is updated every 300 episodes, each episode consists of 20 time slots. We compare our MC-DDPG with two benchmarks, in each of which for any new task, the whole algorithm needs to be initialized and performed again.

1) Twin Delayed Deep Deterministic Policy Gradient (TD3), which is a state-of-the-art RL algorithm [11] without the meta critic.
2) Zero-Force Exhausting (ZF Exhaust), where the digital beamforming is based on the ZF method, and the IOS phase shift optimization is performed based on the Exhaustion Attack method with the phase shifts of IOS elements discretized.

B. Sum Rate Performance in Dynamic Settings

In Fig. 4, we evaluate how the performance of the proposed algorithm varies with the channel states. Specifically, we update the transition probability matrix in (9) every 300 episodes, i.e., the task is also updated periodically.

As shown in Fig. 4, the proposed MC-DDPG converges within 50 episodes to provide a better performance within 50 episodes and achieves a higher sum rate compared to the two benchmarks. This shows that the proposed scheme can efficiently adapt to rapid environment changes. Fig. 4 also reveals that there is a trade-off between the convergence speed of MC-DDPG and the achievable sum rate.

In Fig. 5, we evaluate the performance of the proposed scheme when mobile users move rapidly. The proposed MC-DDPG can rapidly converge to a higher sum rate compared to the benchmarks. Specifically, since the MC-DDPG can converge within 100 episodes and we set each user's position changes several meters every second, we remark that it can support the user mobility at a 36 km/h level. The MC-DDPG also shows a more stable convergence performance compared to ZF-Exhaust since the ZF-Exhaust is subject to random initialization of IOS phase shifts.

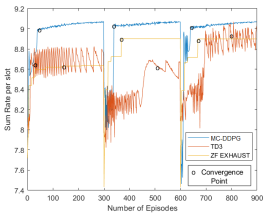

Fig. 4. Sum rate performance with respect to the varying channel states

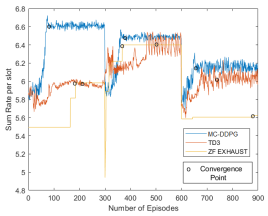

Fig. 5. Sum rate performance with respect to User's Loactions

C. Influence of the Number of IOS Elements

Fig. 6 shows the sum rate varying with the number of IOS elements N. We observe that the proposed MC-DDPG achieves a higher sum rate as N increases. Besides, the convergence speed maintains the same when N changes, implying the effectiveness of the proposed meta-critic method against the large-scale IOS-assisted communication system.

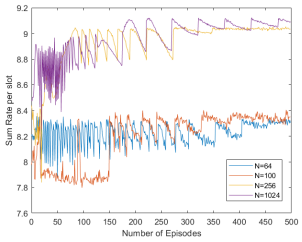

Fig. 6. Sum rate v.s. the number of IOS elements

VI. Conclusion

In this paper, we considered an IOS-assisted communication system in dynamic environments. We proposed a meta-critic RL scheme for sum rate maximization given the limited CSI. By designing and training a meta critic, the proposed scheme can adapt to the dynamic environment changes such as the CSI and user positions. Simulation results verify that the MC-DDPG algorithm achieves a faster convergence speed and a higher sum rate compared to the benchmarks. Two conclusions can be drawn below. First, the designed meta-critic significantly enhances the robustness of the IOS-assisted multi-user communications against user mobility and the dynamic CSI. Second, there exists a trade-off between the convergence speed and the achievable sum rate of MC-DDPG.

References

[1] S. Zhang, H. Zhang, B. Di, Y. Tan, M. Di Renzo, Z. Han, H. V. Poor, and L. Song, "Intelligent omni-surfaces: Ubiquitous wireless transmission by reflective-refractive metasurfaces," *IEEE Trans. Wireless Commun.*, vol. 21, no. 1, pp. 219–233, Jan. 2021.

[2] C. Huang, R. Mo, and C. Yuen, "Reconfigurable intelligent surface assisted multiuser miso systems exploiting deep reinforcement learning," *IEEE J. Selected Areas Commun.*, vol. 38, no. 8, pp. 1839–1850, Aug. 2020.

[3] G. Lee, M. Jung, A. T. Z. Kasgari, W. Saad, and M. Bennis, "Deep reinforcement learning for energy-efficient networking with reconfigurable intelligent surfaces," in *IEEE Int. Conf. Commun. (ICC)*, Jun. 2020.

[4] J. Zhang, J. Li, Y. Zhang, Q. Wu, X. Wu, F. Shu, S. Jin, and W. Chen, "Collaborative intelligent reflecting surface networks with multi-agent reinforcement learning," *IEEE J. Selected Topics Signal Process.*, vol. 16, no. 3, pp. 532–545, Mar. 2022.

[5] S. Zeng, H. Zhang, B. Di, Y. Tan, Z. Han, H. V. Poor, and L. Song, "Reconfigurable intelligent surfaces in 6G: Reflective, transmissive, or both?" *IEEE Commun. Lett.*, vol. 25, no. 6, pp. 2063–2067, Jun. 2021.

[6] H. Zhang and B. Di, "Intelligent omni-surfaces: Simultaneous refraction and reflection for full-dimensional wireless communications," *IEEE Commun. Surveys & Tutorials*, vol. 24, no. 4, Apr. 2022.

[7] R. W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave mimo systems," *IEEE J. Selected Topics Signal Process.*, vol. 10, no. 3, pp. 436–453, 2016.

[8] B. Di, H. Zhang, L. Song, Y. Li, Z. Han, and H. V. Poor, "Hybrid beamforming for reconfigurable intelligent surface based multi-user communications: Achievable rates with limited discrete phase shifts," *IEEE J. Selected Areas Commun.*, vol. 38, no. 8, pp. 1809–1822, Aug. 2020.

[9] A. G. Barto, R. S. Sutton, and C. W. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Sys. Man Cyber.*, no. 5, pp. 834–846, 1983.

[10] F. Sung, L. Zhang, T. Xiang, T. Hospedales, and Y. Yang, "Learning to learn: Meta-critic networks for sample efficient learning," *arXiv preprint arXiv:1706.09529*, 2017.

[11] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Int. Conf. Machine Learning (ICML)*, pp. 1587–1596, Jul. 2018.